O‘ZBEKISTON RESPUBLIKASI
RAQAMLI TEXNOLOGIYALAR VAZIRLIGI

TOSHKENT AXBOROT TEXNOLOGIYALARI UNIVERSITETI

B.Sh. Maxkamov, X.N. Zaynidinov, J.N. Nurmurodov

SUN’IY INTELLEKT ASOSLARI

Muxammad al-Xorazmiy nomidagi
Toshkent axborot texnologiyalari universitetining
2024-yil 27-iyundagi 749-01-sonli buyrug‘iga asosan
darslik sifatida tavsiya etilgan

Toshkent – 2024

UO‘K: 004.2
KBK: 32.973
Z 66
Maxkamov B.Sh., Zaynidinov X.N., Nurmurodov J.N. “Sun’iy intellekt asoslari” / Toshkent axborot texnologiyalari universiteti. – 2024 yil 264 b.

Mazkur darslik sun’iy intellekt va uning asosiy tushunchalarini o‘rganishga bag‘ishlangan. Darslikda sun’iy intellekt nazariyasining asosiy tushunchalari keltirilgan. Klassik va zamonaviy sun’iy intellekt texnologiyalari, shuningdek, ularning matematik modellari va dasturiy algoritmlari keltirib o‘tilgan.

Ushbu darslik universitetning barcha bakalavr yo‘nalishlari uchun mo‘ljallangan.

Taqrizchilar:

Kobulov A.V.	t.f.d. Mirzo Ulug‘bek nomidagi O‘zbekiston Milliy universiteti “Axborot xavfsizligi” kafedrasi professori.
Nazirova E.Sh.	t.f.d.Toshkent Axborot Texnologiyalari Universiteti “Televizion texnologiyalar” fakulteti dekani professor.

ISBN: 978-9943-5805-5-8

© X.N. Zaynidinov va boshqalar, 2024
© «JAVOHIR ILM NASHR», 2024

MUNDARIJA

Kirish ……5

I BOB. SUN'IY INTELLEKTGA KIRISH

1.1. Sun'iy intellektga kirish ……...15

1.2. Sun'iy intellekt imkoniyatlari va turlari ……..…. 29

II BOB. MA'LUMOTLARNING INTELLEKTUAL TAHLILI, AGENTLAR VA EKSPERT TIZIMLARI

2.1. Intellektual agent.....41

2.1.1. Intellektual agent turlari……..45

2.2. Ma'lumotlarning intellektual tahlili.....……51

2.2.1. Ma'lumotlar intellektual tahlilining rivojlanish bosqichlari.....51

2.2.2. Ma'lumotlarni qidirish algoritmlari (Data search).....81

2.3. Bilimlar bazasi.....94

2.3.1. Bilimlarni ajratib olish……99

2.3.2. Bilimlarni taqdim etish modellari……...104

2.3.3. Noaniqlikni bilim modellari……108

2.4. Ekspert tizimlari.....115

III BOB. MASHINALI O'QITISH

3.1. Mashinali o'qitishga kirish 127

3.1.1. Mashinali o'qitish turlari.....129

3.1.2. Xatoliklarni hisoblash.....135

3.1.3. Gradiyent pastlash.....140

3.2. O'qituvchili o'qitish algoritmlari.....144

3.2.1. Chiziqli regressiya.....144

3.2.2. Polinomial regressiya.....152

3.2.3. Logistik regressiya 154

3.2.4. K-NN algoritmi.....160

3.2.5. Support vektor mashinasi (SVM)……..166

3.2.6. Regulyarizatsiya.....171

3.3. O'qituvchisiz o'qitish algoritmlari.....177

3.3.1. K-Means Clustering.....177

IV BOB. NEYRON TARMOQLAR

4.1. Sun'iy neyron tarmoqlarga kirish.....185

4.1.1. Eng sodda perseptron.....188

4.2. Chuqur o'qitish algoritmlari 191

4.2.1. Ko'p qatlamli perseptron.....194

4.2.2. ReLU funksiyasi.....195

4.2.3. Softmax funksiyasi.....198

4.2.4. Oldinga va ortga siljish algoritmi.....200

4.2.5. CNN arxitekturasi va ishlash strukturasi 204

4.2.6. RNN arxitekturasi va ishlash strukturasi.....215

4.3. Tabiiy tilga qayta ishlov berish 223

4.3.1. Morfologik tahlil.....225

4.3.2. Sintaktik tahlil 231

4.3.3. Semantik tahlil.....239

Xulosa..... 258

Adabiyotlar..... 260

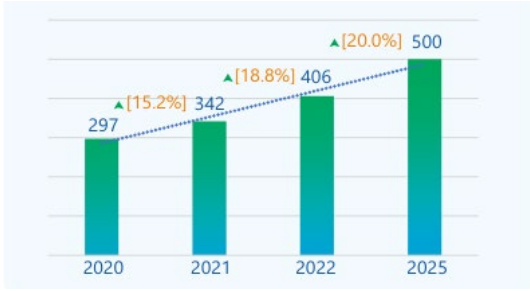


KIRISH

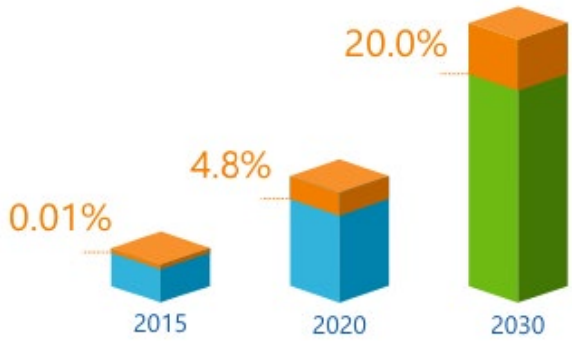
Ilm-fan va axborot-kommunikatsiya texnologiyalari jadal taraqqiy etib borayotgan bugungi kunda dunyoning rivojlangan mamlakatlarida davlat va jamiyat boshqaruvi, iqtisodiyot, sanoat, ijtimoiy himoya, ta’lim, tibbiyot, qishloq xo‘jaligi, mudofaa, xavfsizlik, turizm va boshqa sohalarda zamonaviy axborot texnologiyalari va sun’iy intellekt imkoniyatlaridan keng foydalanish urfga kirmoqda. O‘zbekistonda ham, axborotlashtirish va raqamli iqtisodiyotni rivojlantirish orqali 2030-yilga qadar innovatsion taraqqiy etgan yetakchi davlatlar qatoridan o‘rin egallashi ustuvor vazifa sifatida belgilangan, shunga asosan 2021-2022-o‘quv yilidan boshlab 5ta OTMda grant asosida “Sun’iy intellekt” yo‘nalishi bo‘yicha kadrlar tayyorlash boshlandi. O‘zbekistonda sun’iy intellekt va shu kabi texnologiyalarni joriy etish orqali 2025-yilga borib mamlakatning ikkita asosiy sanoatida 430 mingdan ortiq yangi ish o‘rinlari yaratilishi kutilmoqda. Shuningdek, sun’iy intellekt murakkab tahlillar va katta ma’lumotlar bilan ishlovchi dasturlar va texnologiyalarni o‘z ichiga olib, mantiqli izchil mulohaza qilish hamda tavsiya berish qobiliyatiga ega “*aqlli”* texnologiya hisoblanadi. Mutaxassislar tomonidan sun’iy intellektga to‘rtinchi sanoat inqilobining asosi sifatida qaralmoqda. Rivojlangan mamlakatlarda ijtimoiy sohada, xususan, sog‘liqni saqlash, ta’lim, bandlik, ijtimoiy himoya va boshqa yo‘nalishlarda sun’iy intellekt va zamonaviy axborot texnologiyalari imkoniyatlaridan foydalanish keng tus olmoqda. Shu qatorda, sun’iy intellekt asosida ishlovchi kasbiy-malakaviy moslik bo‘yicha dasturning O‘zbekiston

sharoitida qo‘llanilishi norasmiy sektorda band bo‘lgan, ayniqsa, bir martalik ish bozorlaridagi ish qidiruvchilarning bandligini ta’minlashga va ularni ijtimoiy himoya bilan qamrab olishga imkon beradi. Dastlabki bosqichda sun’iy intellekt texnologiyalarini mobil qurilmalardan foydalanish mumkin bo‘lgan dasturlar orqali taklif qilib, norasmiy sektordagilarni kunlik va mavsumiy ishlarga jalb qilish, ularning bandligini ta’minlash mumkin. Sun’iy intellekt texnologiyalaridan foydalanishni kengaytirish, raqamli ma’lumotlarni yig‘ish, saqlash va qayta ishlash tizimini takomillashtirish maqsadida hozirda yurtimizda ushbu sohada malakali kadrlarni tayyorlash, mazkur yo‘nalishdagi ilmiy-loyihalarni qo‘llab-quvvatlash bo‘yicha qator ishlar amalga oshirilmoqda.

Sun’iy intellektning dunyoda rivojlanish tendensiyasi. Bugungi kunda sun’iy intellekt butun dunyoda rivojlanishda davom etmoqda.



Sun’iy intellekt bozori 2025-yilga kelib 500 milliard dollargacha o‘sishi kutilmoqda. Buning natijasida esa Jahon yalpi ichki mahsulotiga ham ta’sir o‘tkazmoqda.

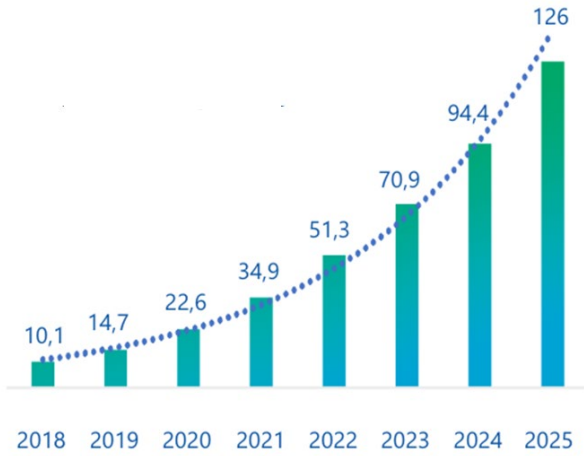


2030-yilga kelib $13 trln foydaga erishish ko‘zda tutilgan. Bu esa Jahon yalpi ichki mahsulotning taxminan 20% ga teng bo‘ladi.

2030-yilda Sun’iy intellektni qo‘llash natijasida kutilayotgan daromad (hajm, %YAIM).

Davlatlar	daromad	foiz
Shimoliy Amerika	$ 3.7 trn.	14,5%
Shimoliy Yevropa	$ 1.8 trn.	9,9%
Janubiy Amerika	$ 0.5 trn.	5,4% $
Xitoy	$ 7 trn.	26%
Rivojlangan Osiyo davlatlari	$ 0.5 trn.	5,4%
Afrika	$ 1.2 tr	5,6%

Sohalarda sun’iy intellektni joriy etish orqali kutilayaotgan foyda hajmi (mlrd.$)



Sun’iy intellekt texnologiyalarini rivojlantirish bo‘yicha yetakchi mamlakatlar erishgan ilmiy natijalar.

Davlatlar	SI sohasida startaplar soni	SI sohasida maqola soni
AQSH	1393	149.066
Xitoy	383	159.818
Buyuk britaniya	245	49.978
Kanada	131	27.019
Yaponiya	113	49.441
Germaniya	106	40.131

O‘zbekistonda sun’iy intellektni rivojlantirishning joriy holati

PQ-4996 17.02.2021	9 ta tajriba sinov loyihalari	O‘zbek tilida izohli lug‘at ishlab chiqildi
PQ-5234 26.08.2021	Maxsus rejim joriy etildi	Tadbirkorlarga imtiyozlar yaratildi
VMQ-475 31.07.2021	Sun’iy intellektni rivojlantirish ilmiy tadqiqot	10 ta maxsus laborotoriya faoliyati yo‘lga qo‘yildi

	instituti tashkil etildi	
VMQ-717 29.11.2021	Maxsus rejim ishchi organi belgilandi	Maxsus rejimning nizomi tasdiqlandi

Hukumatning sun'iy intellektga tayyorgarlik indeksi. Hukumatning sun'iy intellektga tayyorgarlik indeksi - jahon mamlakatlarining sun'iy intellektni joriy qilish uchun tayyorgarlik darajasi va yaratilgan sharoitlarni baholovchi Buyuk Britaniyaning " Oxford Insights " tashkiloti va Kanadaning Xalqaro tadqiqotlarni rivojlantirish instituti tomonidan amalga oshiriladigan xalqaro reytingidir.

Reytingda O'zbekiston Respublikasining o'rnini keskin oshishiga mamlakatimizda yaratilayotgan quyidagi imkoniyatlar sabab bo`lmoqda:

- Hukumat qarorlari;

- Tizimlar rivojlanishi ;

Maxsus rejim imtiyozlari:

- Litsenziya to'lovlari 2 barobarga qisqaradi ;

- IT Park rezidentlariga berilgan imtiyozlar va preferensiyalar tatbiq etiladi;

- O'z xodimlari malakasini oshirish va qayta tayyorlashga ketgan xarajatlari davlat tomonidan kompensatsiya qilinadi (PQ–4862);

Sun'iy intellektning joriy holati. Bugungi kunda quyidagi dolzarb loyihalar amalga oshirilmoqda:

- Ta'lim va kadrlar salohiyatini oshirish maqsadida El-yurt umidi jamg‘armasi hisobidan 11 nafar talaba yetakchi xorijiy OTM larga yuborildi;

- Koreya Respublikasi bilan hamkorlikda sun'iy intellekt yo‘nalishida 50 ga yaqin davlat idoralari xodimlari malakasi oshirildi;

Davlat xizmatlaridan foydalanishda aholi uchun qulayliklar yaratish:

- Masofadan turib biometrik identifikatsiyalash loyihasi(My ID) ishga tushirildi;

- Tizimga ulangan tashkilotlar: 28 ta bank (jami banklarning 85 foizi) 5 ta davlat tashkiloti tizimi foydalanuvchilari soni 14 mlliondan ortiqni tashkil etadi;

- OTMlarda sun'iy intellekt yo‘nalishida 572 ta kvotalar ajratilgan bo‘lib, ular :

510 ta bakalavr va 62 ta magistr;

Sun'iy intellekt yo‘nalishi mavjud OTMlar soni 12 taga yetdi;

Ma'lumki, O‘zbekiston Respublikasi Prezidentinin 2022-yil 28-fevraldagi PF-60-son farmonida sun'iy intellektni mamlakatimizda muvaffaqiyatli amalga oshirish uchun "Sun'iy intellekt texnologiyalarini 2023-2030-yillarda rivojlantirish Strategiyasi" ni ishlab chiqish belgilangan. Shunga asosan Oliy ta'lim muassasalarida

sun’iy intellekt yo‘nalishi bo‘yicha xorijiy mutaxassislarni o‘quv jarayoniga faol jalb etish, ta’lim sifatini xalqaro darajaga yetkazish, xalqaro hamkorlikda ilmiy tadqiqotlar olib borish va sun’iy intellekt sohasi fanlari uchun o‘quv qo‘llanma va darsliklar ishlab chiqish bo‘yicha aniq vazifalar belgilab olingan. Bu vazifalardan kelib chiqib ushbu darslik ishlab chiqilgan. Darslikning birinchi bobida sun’iy intellektning tarixi, asosiy tushunchalari va tamoyillari hamda qo‘llanish sohalari, tatbiq etishdagi axloqiy va huquqiy muammolari ko‘rib chiqilgan. Ikkinchi bobida ma’lumotlarning intellektual tahlili(Data Mining), intellektual agentlar, bilimlar bazasi, ma’lumotlarni qidiruv algoritmlari va ekspert tizimlari keltirilgan. Uchinchi bobida mashinali o‘qitish algoritmlari, o‘qituvchili, o‘qituvchisiz va mustahkamlangan o‘qitish usullari batafsil keltirilgan. To‘rtinchi bobida esa neyron tarmoqlar, neyron tarmoqlarni o‘qitish algoritmlari, chuqur o‘qitish algoritmlari, CNN va RNN arxitekturalari va strukturalari , tabiiy tilga ishlov berish algoritmlari keng yoritilgan. Ushbu darslik mualliflarning Rossiiyadagi Sankt-Peterburg davlat elektrotexnika universiteti, Janubiy Koreyaning Dongseo universiteti, AQSH ning Sent-Luis va Vestern Michigan universitetlari, Germaniyaning Karl Von Ossiyetski universiteti, Toshkent axborot texnologiyalari universitetida olgan ko‘p yillik tajribasi mahsulidir.

Mazkur darslik O‘zbekiston Respublikasidagi barcha oliy o‘quv yurtlari bakalavriat talabalariga mo‘ljallangan. Lekin darslik kompyuter texnologiyalari sohalarida ishlovchi mutaxassislar,

magistrantlar, doktorantlar va sun’iy intellektga qiziquvchi barcha o‘quvchilarga ham foydali bo‘lishi mumkin.

Mualliflar darslikni tayyorlashdagi bevosita yordamlari va qimmatli maslahatlari uchun taqrizchilar t.f.d., professor Kobulov A.V. va t.f.d., professor Nazirova E.Sh. larga chuqur minnatdorchilik bildiradilar.

Shuningdek, darslikni qo‘lyozmasini tayyorlashda katta yordam ko‘rsatganliklari uchun kafedra katta o‘qituvchilari S. Qobulov va J. Modullayevga, doktorantlar X.Ma’mirov, Z.Dadajonova va A.Boytemirovlarga o‘z minnatdorchiliklarimizni bildiramiz. Mualliflar barcha o‘quvchilardan “Sun’iy intellekt asoslari” nomli darslik to‘g‘risidagi o‘zlarining fikr-mulohazalarini quyidagi elektron pochta manzillariga yuborishlarini so‘raymiz: tet2001@rambler.ru va nurmurodov1994@gmail.com. Albatta biz sizlardan kelgan fikr-mulohazalarni darslikning keyingi nashrlarida inobatga olamiz.

I BOB

SUN'IY INTELLEKT

SUN’IY INTELLEKT

Sun’iy intellekt turli amallarni bajarishga mo‘ljallangan algoritm hamda dasturiy tizimlardan iborat texnologiya hisoblanadi. U bugungi kunda inson ongi bajarishi mumkin bo‘lgan bir qancha vazifalarning uddasidan chiqa olish imkoniyatiga ega.

1.1. SUN’IY INTELLEKTGA KIRISH

Hozirgi vaqtga kelib jadal rivojlanayotgan sohalardan biri sun’iy intellekt sohasi bo‘lib, u aqlli mashinalar yaratish orqali dunyoga yangi o‘zgarishlar olib kirmoqda. Sun’iy intellekt mashinalarni inson kabi fikrlashga va inson kabi harakatlanishiga xizmat qiluvchi bir nechta algoritmlar to‘plami hisoblanadi. Bugungi kunda sun’iy intellekt, aqlli avtomobillar, tibbiyot, kiber sport, san’at, kiberxavfsizlik, biznes va shunga o‘xshash umumiy o‘ziga xos turli sohalarda qo‘llanilmoqda. Sun’iy intellekt ikki so‘zdan iborat bo‘lib, Artificial intelligence, bu yerda Artificial-Sun’iy “*inson tomonidan yaratilgan*” va intelligence-intellekt “*aql-idrok, fikrlash*” ni bildiradi, shuning uchun sun’iy intellekt “inson tomonidan yaratilgan aql-idrok, fikrlash” degan ma’noni ham anglatadi. Kompyuter texnologiyasi bo‘yicha quyidagicha ta’riflashimiz ham mumkin:

Sun’iy intellekt - bu kompyuter fanining bir bo‘limi bo‘lib, o‘zini inson kabi boshqara oladigan, inson kabi fikrlaydigan va qaror qabul qila oladigan aqlli mashinalar to‘plamidir.

Sun’iy intellekt tizimlari yoki aqlli tizimlar, sun’iy aqlli obyektlarni birlashtirgan tizimlardir. Aqlli tizimlar axborot va bilimlar asosida maqsadni sintez qilish, qarorlar qabul qilish, maqsad va qarorlarga erishishning eng optimal usullarini topish, inson bilan o‘zaro aloqada bo‘lishidan qat’iy nazar, axborot jarayoni bilan birlashtirilgan texnik va dasturiy vositalar to‘plami sifatida aniqlanadi. Sun’iy

intellekt sohasining rivojlanish tarixiy bosqichlari quyidagilardan iborat.

Sun'iy intellekt g'oyalarining fanga kirib kelishi *(1900-1950)***.** 1900-yillarning boshlarida sun'iy odamlar terminologiyasi ko'plab ommaviy axborot vositalarida keng yoritila boshlangan. Barcha sohadagi olimlar juda katta qiziqish bildirgan va sun'iy ong yaratish mumkinmi? degan savol paydo bo'la boshlagan. Ba'zi olimlar hozirda biz "robotlar" deb ataydigan manupulyatorlarning birinchi namunalarini yaratganlar (*bu so'z 1921-yilda Chexiya spektaklida keltirilgan*), ammo aksariyati sodda ko'rinishda bo'lgan. Ular bug' bilan ishlaydigan, yuz ifodalarini takrorlay oladigan va hatto yurish imkoniyatiga ham ega bo'lgan.



1921-yilda Chex dramaturgi **Karel Chapek** *(1890 - 1938)* birinchi marotaba "Rossumning universal robotlari" ilmiy-fantastik pyesasini chiqardi, unda u robotlar deb atagan "Sun'iy odamlar" g'oyasini taqdim etdi. Bu so'z birinchi marta qo'llanilishi edi.



1929-yilda Yaponiyalik professor **Makoto Nishimura** *(1883-1956)* Yaponiyaning birinchi funksional roboti **Gakutensoku** ixtirochisi sifatida mashhur bo'lgan.

1949-yilda kompyuter olimi **Edmund Kallis Berkli** *(1909-1988)* kompyuterlarning yangi modellarini inson miyasi bilan taqqoslagan **“Ulkan miyalar yoki fikrlaydigan mashinalar”** kitobini nashr etdi.



Sun’iy intellekt paydo bo‘lishi*(1950-1956)*. Bu vaqt oralig‘i sun’iy intellektga bo‘lgan qiziqish juda ham avj olgan payt bo‘lgan. Alan Turing o‘zining “Kompyuter texnikasi va razvedkasi” asarini nashr etdi. Asarda “Sun’iy intellekt” atamasi kiritilgan va juda mashhur bo‘lgan. Bu oxir-oqibat Turing testi bo‘lib, mutaxassislar kompyuter intellektini baholash uchun foydalana boshladilar.

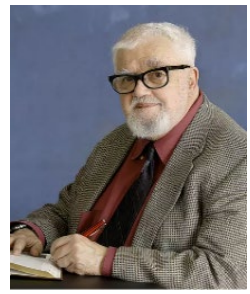
1950-yilda Alan Turing*(1912-1954)* “Kompyuter texnikasi va qidiruvi” ni nashr etdi, u “Taqlid o‘yini” deb nomlangan mashina intellektini sinab ko‘rishni taklif qildi.



1952-yilda Artur Samuel*(1901-1990)* kompyuter olimi shashka o‘ynash dasturini ishlab chiqdi, bu o‘yinni mustaqil ravishda birinchi bo‘lib o‘rgangan.



1955-yilda Jon Makkarti Dartmutda **“Sun’iy intellekt”** bo‘yicha seminar o‘tkazdi, bu so‘zning birinchi qo‘llanilishi va u qanday qilib mashhur bo‘lgani to‘g‘risida so‘z yuritdi.



Sun’iy intellektning yetuklikka erishish davri *(1957-1979).* 1950-yillarning oxiri va 1960-yillar boshida ijod davri bo‘lgan. O‘sha vaqtda qo‘llanilayotgan dasturlash tillaridan tortib robotlar g‘oyasini o‘rganuvchi kitoblar va filmlargacha sun’iy intellekt asosiy g‘oyaga aylandi.Yaponiyada qurilgan birinchi antropomorfik robot, muhandislik fakulteti talabasi tomonidan qurilgan avtonom avtomobilning birinchi namunasi hisoblangan. Biroq, bu sun’iy intellekt tadqiqotlari uchun kurash davri ham bo‘lgan, chunki AQSh hukumati sun’iy intellekt tadqiqotlarini moliyalashtirishni davom ettirishga unchalik qiziqish bildirmagan.

1958-yilda Jon Makkarti LISP (Ro‘yxatni qayta ishlashning qisqartmasi) ni yaratdi, u hali ham mashhur bo‘lgan sun’iy intellekt tadqiqotlari uchun birinchi dasturlash tilidir.

1959-yilda Artur Samuel mashinalarni shaxmat o‘ynashni dasturlashtirgan odamlarga qaraganda yaxshiroq o‘rgatish haqida nutq so‘zlayotganda “*Mashinali o‘qitish*” atamasini yaratdi.

1961-yilda birinchi sanoat roboti Unimate Nyu-Jersidagi

General Motors konveyerda ishlay boshladi, unga qoliplarni va payvandlash qismlarini avtomobillarga tashish vazifasi yuklatildi.

1965-yilda Edvard Feygenbaum va Joshua Lederberg mutaxassislarining fikrlash va qaror qabul qilish qobiliyatlarini takrorlash uchun dasturlashtirilgan sun'iy intellekt shakli bo'lgan birinchi "*ekspert tizimi*" ni yaratdilar.

1966-yilda Jozef Vayzenbaum birinchi "*chatterbot*" (keyinchalik chatbot deb qisqartirildi) yaratadi, u odamlar bilan suhbatlashish uchun ta'biiy tilni qayta ishlash (NLP) dan foydalanadi. "*Avtomatika*" jurnalida, u sun'iy intellektga yangi yondashuvni taklif qiladi, bu keyinchalik biz hozir "*chuqur o'qitish*" deb nomlangan usulga aylandi.

1973-yilda Jeyms Laytxill ismli amaliy matematik Britaniya Ilmiy Kengashiga hisobot berib, olimlar va'da qilinganidek yutuqlarga erisha olmaganini ta'kidladi, bu esa Britaniya hukumati tomonidan sun'iy intellekt bo'yicha tadqiqotlarni moliyalashtirishni sezilarli darajada qisqartirishga olib keladi.

1979-yilda esa Jeyms L., Adams Stendford aravasini yaratadi, u stullar bilan to'la xonani inson aralashuvisiz muvaffaqqiyatli aylanib chiqa oladi. Shu yili sun'iy intellektning rivojlantirish assotsiatsiyasi (ASIA) sifatida tanilgan Amerika sun'iy intellekt assotsiatsiyasiga ham asos solingan.

Sun'iy intellektning rivojlanish davri*(1980-1987)*. 1980-yillarga kelib aksariyat holatlar sun'iy intellektda tez o'sish hamda unga

qiziqish davrini ko‘rsatadi va bu jarayon “*Sun'iy intellekt rivojlanishi*” deb nomlanadi. Bu tadqiqotdagi yutuqlar va tadqiqotchilarni qo‘llab-quvvatlash uchun davlat tomonidan qo‘shimcha moliyalashtirish natijasida yuzaga keladi. Chuqur o‘rganish texnikasi va Ekspert tizimidan foydalanish yanada ommalashadi, bu ikkalasi ham kompyuterlarga o‘z xatolaridan saboq olish va mustaqil qarorlar qabul qilish imkonini berdi.

1980-yilda Stenfordda AAAI (*American Association for Artificial Intelligence*)ning birinchi konferentsiyasi bo‘lib o‘tadi. XCON (*ekspert konfigurator*) nomi bilan tanilgan birinchi ekspert tizimi tijorat bozoriga kirdi. U mijozning ehtiyojlaridan kelib chiqqan holda komponentlarni avtomatik tanlash orqali kompyuter tizimlarini buyurtma qilishda yordam berish uchun mo‘ljallangan.

1981-yilda Yaponiya hukumati Beshinchi avlod kompyuteri loyihasiga 850 million dollar (*bugungi pulda 2 milliard dollardan ortiq*) ajratadi. Ularning maqsadi inson tilida tarjima qila oladigan, suhbatlashadigan va inson darajasida fikr bildira oladigan kompyuterlarni yaratish edi.

1984-yilga kelib esa ASIA kelayotgan “*qish*” haqida ogohlantiradi, bunda sun'iy intellektga qiziqish va moliyalashtirish keskin kamayadi. Bu esa tadqiqotlarni sezilarli darajada qiyinlashtiradi.

1985-yilda ASIA konferensiyasida AARON deb nomlanuvchi avtonom chizmachilik dasturi namoyish etildi.

1986-yilda Ernst Dikmann va uning Myunxendagi Bundesver universitetidagi jamoasi birinchi haydovchisiz avtomobilni (*yoki robot avtomobil*) yaratdi va namoyish qiladi.

Sun’iy intellektning rivojlanishdan to‘xtash davri *(1987-1993).* Amerika sun’iy intellekt assotsiatsiyasi ogohlantirganidek, sun’iy intellektga qish kirib keldi. Bu atama foydalanuvchi, jamoatchilik va xususiy sun’iy intellekt bilan shug‘ullanuvchi kompaniyalarga nisbatan past qiziqish davrini keltirib chiqaradi. Xususiy investorlar ham, hukumat ham sun’iy intellektga qiziqishi kamaydi va yuqori xarajat va past daromadlilik tufayli moliyalashni to‘xtatadi. Ushbu sun’iy intellekt qishi mashinalar bozori va ekspert tizimlaridagi ba’zi muvaffaqiyatsizliklar, jumladan beshinchi avlod loyihasining tugashi, strategik hisoblash tashabbuslarining qisqarishi va ekspert tizimlarini joylashtirishning sekinlashishi tufayli yuzaga keladi.

1987-yilda LISP-ga asoslangan ixtisoslashtirilgan apparat vositalari bozori, IBM va Apple tomonidan taklif qilinadigan arzonroq va qulayroq raqobatchilar tufayli qulab tushdi. Bu ko‘plab ixtisoslashgan LISP kompaniyalarini bankrot yoqasiga olib keladi.

1988-yilda Rollo Karpenter ismli kompyuter dasturchisi Jabberwacky chatbotini ixtiro qildi, u odamlarga qiziqarli suhbatlar berish uchun dasturlashtirgan edi.

Sun’iy intellekt 1993-yildan 2011-yilgacha bo‘lgan davr. Sun’iy intellekt rivojlanishdan to‘xtash davrida mablag‘ yetishmasligiga qaramay 90-yillarning boshlarida sun’iy intellekt bo‘yicha tadqiqotlar,

jumladan, amaldagi jahon chempioni shaxmatchini mag‘lub etishi mumkin bo‘lgan birinchi sun’iy intellekt tizimini joriy etish bo‘yicha dastlabki qadamlar qo‘yildi. Bu davr, shuningdek, birinchi Roomba va Windows kompyuterlarida sotuvda mavjud bo‘lgan birinchi nutqni aniqlash dasturi kabi innovatsiyalar orqali sun’iy intellektni kundalik hayotga kiritdi. Qiziqishning ortishi ortidan tadqiqotni moliyalashtirishning ko‘payishi kuzatildi, bu esa yanada ko‘proq yutuqlarga erishish imkonini berdi.

1997-yilda Deep Blue (*IBM tomonidan ishlab chiqilgan)* shaxmat bo‘yicha jahon chempioni Gari Kasparovni shaxmat o‘yinida mag‘lub etib, inson shaxmat chempionini mag‘lub etgan birinchi dasturga aylandi.

1997-yilda Windows nutqni aniqlash dasturini chiqardi (*Dragon Systems tomonidan ishlab chiqilgan)*

2000-yilda Professor Sintiya Breazeal ko‘zlari, qoshlari, quloqlari va og‘zi mavjud bo‘lgan yuzi bilan inson his-tuyg‘ularini taqlid qila oladigan birinchi robotni yaratdi. U Qismet deb atalgan.

2003-yilda NASA Marsga ikkita rover uchirdi (Ruh va Imkoniyat) va ular sayyora yuzasida inson aralashuvisiz harakat qilishdi.

2006-yilda Twitter, Facebook va Netflix kabi kompaniyalar sun’iy intellektdan reklama va foydalanuvchi tajribasi (UX) algoritmlarining bir qismi sifatida foydalanishni boshladilar.

2010-yilda Microsoft Xbox 360 Kinect-ni ishga tushirdi, bu tana harakatini kuzatish va uni o‘yin yo‘nalishlariga tarjima qilish uchun

mo‘ljallangan birinchi o‘yin uskunasi edi. Shu yilda Uotson *(IBM tomonidan yaratilgan)* ismli savollarga javob berish uchun dasturlashtirilgan NLP kompyuteri televizion o‘yinda ikkita sobiq chempionga qarshi Jeopardy ustidan g‘alaba qozondi.

2011-yilda Apple birinchi mashhur virtual yordamchi dasturini chiqardi.

Sun’iy intellekt 2012-yildan hozirgi kungacha. Bu davrda sun’iy intellektning hozirgi kungacha bo‘lgan eng so‘nggi yutuqlariga olib keladi. Bunda virtual yordamchilar, qidiruv tizimlari va boshqalar kabi keng tarqalgan sun’iy intellekt vositalarining ko‘payishi yuzaga keldi . Bu vaqt oralig‘ida Deep Learning va Big Dataga bo‘lgan talablar ortib bordi.

2015-yilda Ilon Mask, Stiven Xoking va Stiv Voznyak *(3000 dan ortiq)* urush maqsadlarida avtonom qurollarni ishlab chiqishni *(keyinroq esa foydalanishni)* taqiqlovchi dunyo hukumati tizimlariga ochiq xat imzoladilar.

2016-yilda Hanson Robotics kompaniyasi birinchi *“robot fuqarosi”* sifatida tanilgan Sofiya ismli gumanoid robotni yaratdi va insonning real qiyofasi va his-tuyg‘ularni ko‘rish va takrorlash, shuningdek, muloqot qilish qobiliyatiga ega yaratilgan birinchi robot edi.

2017-yilda Facebook ikkita sun’iy intellekt chatbotlarini suhbatlashish va muzokaralar olib borishni o‘rganish uchun

dasturlashtirdi, biroq ular ingliz tilidan voz kechishdi va o‘z tillarini butunlay avtonom tarzda rivojlantirishdi.

2018-yilda Alibabaning tilni qayta ishlovchi sun'iy intellekt deb nomlangan Xitoy texnologik guruhi Stenfordda o‘qish va tushunish testida inson intellektini yutib chiqdi.

2019-yilda Google AlphaStar StarCraft 2 video o‘yinida Grandmasterga yetib bordi, bu esa inson o‘yinchilarining 0,2 foizidan oshib ketdi.

2020-yilda Open Sun'iy intellekt kod, she'riyat va boshqa shunga o‘xshash til va yozish vazifalarini yaratish uchun Deep Learningdan foydalanadigan model GPT-3 sinovini boshladi. Garchi bu birinchi bo‘lmasada, u odamlar tomonidan yaratilgan kontentdan deyarli farq qilmaydigan birinchi na'munasidir.

2021-yilda Open Sun'iy intellekt DALL-E-ni ishlab chiqdi, u aniq lavhalar yaratish uchun yetarlicha tasvirlarni qayta ishlay oladi va tushuna oladi, sun'iy intellektning vizual dunyoni tushunishiga bir qadam yaqinlashtiradi.

Sun'iy intellektda Turing mashinasi. Turing mashinalari sun'iy intellektni o‘rganishda ham qo‘llaniladi. Sun'iy intellekt (SI) tadqiqotchilari kompyuterlar nima qila olishini o‘rganish uchun Turing mashinalaridan foydalanadilar. Turing mashinalarining cheklovlarini bilgan holda, SI tadqiqotchilari aqlliroq algoritmlarni ishlab chiqishlari va yanada aqlli kompyuter dasturlarini yaratishlari mumkin.

Sun’iy intellekt algoritmi nima ekanligini tushunishdan oldin, sun’iy intellekt nima ekanligini tushunish kerak. Bu nafaqat tushunish, balki aqlli obyektlarni, ya’ni turli xil yangi vaziyatlarda qanday samarali va xavfsiz harakat qila oladigan mashinalarni yaratish bilan bog‘liq. Odatda, bu tizimlarni qurish va baholash uchun Turing testidan foydalaniladi. Buyuk kompyuter olimi Alan Turing fikrlash tajribasini ishlab chiqdi, unda aqilli mashina testdan o‘tkaziladi, agar so‘roqchi bergan savollaridan, olgan javoblarini, insondan yoki kompyuterdan kelganligini aniqlay olmasa, bu mashinaga sun’iy intellekt deb qarash mumkin.

Shunga asosan, sun’iy intellekt algoritmlari Turing testidan o‘tish uchun quyidagi imkoniyatlarga ega bo‘lishi shart:

- tabiiy tilni qayta ishlash
- bilimlarni ifodalash
- avtomatlashtirilgan fikrlash
- mashina o‘rganishi
- kompyuter ko‘rish
- robototexnika

Informatika fanida algoritm muammoni hal qilish yoki hisoblashni bajarish uchun cheklangan ko‘rsatmalar to‘plamidir. Sun’iy intellekt algoritmlari esa yuqorida aytib o‘tilgan vazifalarni bajara oladigan aqlli mashinalarni amalga oshiradigan algoritmlar to‘plami hisoblanadi.

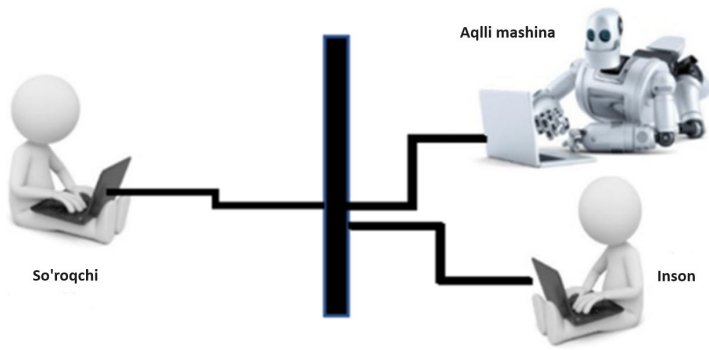
Turing testi sun’iy intellektda muhim ahamiyatga ega, chunki u inson tomonidan bajariladigan vazifalarni bajarish uchun kompyutyerdan qanday foydalanish mumkinligini ko‘rsatadigan modeldir. Masalan, Turing mashinasi shaxmat o‘yinini o‘ynash yoki matematik masalani yechish uchun ishlatilishi mumkin. Mashina inson miyasi kabi boshqa mashinalarning xatti-harakatlarini simulyatsiya qilish uchun ham ishlatiladi.

Turing testi g‘o‘yasi, aqlli mashinaning ish faoliyatini insonnikiga nisbatan o‘lchaydi. Shunga asosan Turing “taqlid o‘yini” deb atagan testi quyidagicha amalga oshiriladi. Bunda yaratilgan aqilli mashina va inson alohida ikkita xonalarga joylashtiriladi. Uchinchi xonada mashina va insonni so‘roq qilish uchun so‘roqchi joylashtiriladi(1.2.3-rasm). So‘roq qiluvchi ularning hech birini ko‘ra olmaydi yoki to‘g‘ridan-to‘g‘ri gaplasha olmaydi, qaysi shaxs aslida mashina ekanligini bilmaydi va ular bilan faqat ovoz uzatuvchi qurilma yoki tekst orqali muloqot qilishi mumkin.

So‘roq qiluvchidan faqat ushbu qurilma orqali savollarga berilgan javoblari asosida kompyuterni odamdan ajratishi so‘raladi. Agar so‘roq qiluvchi mashinani odamdan ajrata olmasa, Turingning ta’kidlashicha, mashinani aqlli deb hisoblash mumkin.

So‘roqchini mashinadan ham, boshqa inson ishtirokchisidan ham ajratib qo‘yish orqali, testda so‘roqchi mashinaning ko‘rinishi yoki uning ovozining biron bir mexanik xususiyatidan foydalanib aniqlay olmasligini ta’minlaydi. Biroq, so‘roq qiluvchining har qanday savoli,

qanchalik ayyor yoki bilvosita bo‘lishidan qat’iy nazar, asosiy urg‘u muloqot orqali kompyuterning ma’lumotlarini ochishga harakat qilinadi. Masalan, so‘roq qiluvchi har ikkala sub’yektdan ham arifmetik hisob-kitoblarni bajarishni so‘rashi mumkin, chunki kompyuter buni odamdan ko‘ra tezroq va to‘g‘ri bajara oladi. Inson ekanligini topish uchun esa uning o‘ziga xosligini, hissiy tabiati asosida aniqlashga harakat qiladi. Bunda so‘roq qiluvchi ikkala subyektdan she’r yoki badiiy asarga javob berishni so‘rashi mumkin. Ushbu strategiya kompyutyerdan insonning hissiy tuzilishi haqida bilimga ega bo‘lishini talab qiladi.



1.1.1-rasm. Turing testidan o‘tqazish jarayoni

Jarayon quyidagi bosqichlardan iborat:

1. U intellektning obyektiv tushunchasini, ya’ni inson tomonidan beriladigan savollarga javob berish va xatti-harakatlarini ifodalashga harakat qiladi.

2. Kompyuterga tegishli barcha imkoniyatlarni ishlatadimi yoki yo‘qmi, bundan tashqari mashina o‘z harakatlaridan haqiqatda

xabardormi yo‘qmi, degan chalkash va javobsiz savollarga yechim topadi.

3. U so‘roqchini faqat savollarga berilgan javoblar mazmuniga e’tibor qaratishga majburlash orqali, inson omiliga bog‘liq muammolarni hal qiladi.

Ushbu afzalliklar tufayli Turing testi zamonaviy sun’iy intellekt dasturlarini baholash uchun ishlatiladigan ko‘plab tizimlari uchun asos bo‘lib xizmat qiladi. Mutaxassislikning ba’zi sohalarida potentsial imkoniyatga erishgan dastur, uning ma’lum muammolar to‘plami bo‘yicha ishlashini inson mutaxassisi bilan solishtirish orqali baholanishi mumkin. Ushbu baholash usuli Turing testining o‘zgarishi, bir guruh odamlardan kompyuter va insonning muayyan muammolar to‘plamidagi ish faoliyatini ko‘r-ko‘rona solishtirish so‘raladi. Ko‘rib turganimizdek, ushbu metodologiya zamonaviy ekspert tizimlarini ishlab chiqishda ham, tekshirishda ham muhim vositaga aylandi.

Nazorat savollari:

1. Sun’iy intellekt tushunchasi qanday ma’no anglatadi?
2. Sun’iy intellekt asosida birinchi shaxmat o‘yinin yaratgan olim kim?
3. Sun’iy intellekt g‘oyasi qaysi yillarda kirib kelgan?
4. Sun’iy intellektning yetuklikka erishishi qaysi yillarga to‘g‘ri keladi?
5. Sun’iy intellektda Turing testi qanday ahamiyatga ega?

1.2. SUN’IY INTELLEKT IMKONIYATLARI VA TURLARI

Bugungi kunga kelib sun’iy intellekt yordamida mashinalar yangi vazifalarni o‘rganishi va moslashishi mumkin, bu ularni an’anaviy mashinalarga qaraganda samaraliroq ishlashini ta’minlaydi. Masalan, sun’iy intellekt qobiliyatiga ega robotlar ishlab chiqarish, logistika va sifatni nazorati vazifalarini bajarishi mumkin. Bu mahsulot va xizmatlar sifatini yaxshilab samaradorlikni oshiradi. Sun’iy intellekt imkoniyatlari va kamchiliklari quyidagilardan iborat.

Sun’iy intellektning imkoniyatlari

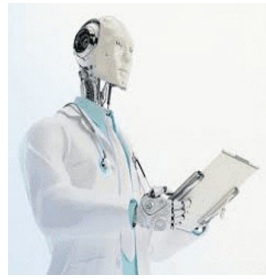
Qishloq xo‘jaligi sohasida. Yerni masofadan zondlash ma’lumotlari asosida tuproq va qishloq xo‘jaligi ekinlari holatini, shuningdek, qishloq xo‘jaligi texnikasi, shu jumladan, kombaynlar ishini monitoring qilish jarayonida sun’iy intellekt texnologiyalarini qo‘llash ko‘zda tutilgan.





Bank sohasida. Tijorat banklari faoliyatini monitoring qilish samaradorligini oshirish va ular tomonidan tartibga solish talablari (SubTech va RegTech) bajarilishini soddalashtirish, shuningdek, bank xizmatlari ko‘rsatish sifatini tahlil qilish, foydalanuvchilarni masofadan biometrik identifikasiyalash (Face-ID) va kredit tavakkalchiliklarini baholash uchun sun’iy intellekt texnologiyalarini qo‘llash vazifasi belgilangan

---



Tibbiyot sohasida. Shifokorlarning o‘rniga robot qo‘yishning asosiy sabablari: Virus yuqush xavfi tug‘ilmaydi*(sababi u robot*). Robot o‘z nomi bilan robot, u tezlik bilan bir bemorga qarab ikkinchi bemor tomon tez harakatlanadi. Ish haqi olmaydi.



Soliq sohasida.

1) yuridik shaxslarning soliq tushumlarini tahlil qilish, soliq to‘lovlaridagi tafovutlarni aniqlashda sun’iy intellekt texnologiyalarini qo‘llash.

2) soliq xabarnomalarini qayta ishlash.

3) soliq solinadigan bazani aniqroq hisoblash, chegirmalarni aniqlash va zarur tuzatishlar kiritish maqsadida hisob varaqlar va balanslar bo‘yicha qoldiqlarni prognozlash.

O‘yinlarda. Sun’iy intellekt o‘yin maqsadlarida ishlatilishi mumkin. Sun’iy intellekt mashinalari shaxmat kabi strategik o‘yinlarni o‘ynashi mumkin, bu yerda mashina ko‘p sonli yurish mumkin bo‘lgan joylarni o‘ylashi kerak.

Moliya sohasida. Sun’iy intellekt va moliya sohalari bir-biriga eng mos keladi. Moliya sanoati moliyaviy jarayonlarga avtomatlashtirish, chatbot, moslashuvchan razvedka, algoritm savdosi va mashinalarni o‘rganishni joriy qilmoqda.

Axborot xavfsizligi sohasida. Ma’lumotlar xavfsizligi har bir kompaniya uchun juda muhim va raqamli dunyoda kiberhujumlar juda tez o‘sib bormoqda. Sun’iy intellekt ma’lumotlaringizni xavfsiz saqlash uchun ishlatilishi mumkin. AEG boti, AI2 platformasi kabi ba’zi misollar dasturiy ta’minotdagi xato va kiberhujumlarni yaxshiroq aniqlash uchun ishlatiladi.

Ijtimoiy tarmoqlarda. Facebook, Twitter va Snapchat kabi ijtimoiy media saytlari juda samarali tarzda saqlanishi va boshqarilishi kerak bo‘lgan milliardlab foydalanuvchi profillarini o‘z ichiga oladi. Sun’iy intellekt katta hajmdagi ma’lumotlarni tartibga solishi va boshqarishi mumkin. Sun’iy intellekt so‘nggi tendentsiyalarni, heshtaglarni va turli foydalanuvchilarning talablarini aniqlash uchun ko‘plab ma’lumotlarni tahlil qilishi mumkin.

Sayohat va transportda. Sun’iy intellekt sayohat sanoati uchun juda rivojlanib bormoqda. Sun’iy intellekt sayohat bilan bog‘liq turli ishlarni bajarishga qodir, masalan sayohatni tashkil qilishdan tortib, mehmonxonalar, parvozlar va mijozlarga eng yaxshi yo‘nalishlarni taklif qilishgacha. Sayohat industriyasi sun’iy intellektga asoslangan chatbotlardan foydalanmoqda, ular mijozlar bilan yaxshi va tezkor aloqa qilish uchun insonga o‘xshash muloqotni amalga oshiradi.

Avtomobil sanoatida. Ba’zi avtomobilsozlik sanoati o‘z foydalanuvchilariga yaxshi ishlashi uchun virtual yordamchini taqdim etish uchun sun’iy intellektdan foydalanadi. Masalan, Tesla TeslaBot aqlli virtual yordamchisini taqdim etdi. Ayni paytda turli sanoat korxonalari sayohatingizni xavfsizroq qiladigan o‘zi boshqariladigan avtomobillarni ishlab chiqish ustida ishlamoqda.

Robototexnikada. Robototexnika sohasida sun’iy intellektning o‘rni katta. Odatda, umumiy robotlar shunday dasturlashtirilganki, ular ba’zi bir takroriy vazifalarni bajara oladilar, ammo sun’iy intellekt yordamida biz oldindan dasturlashtirilmagan holda o‘z tajribalari bilan vazifalarni bajara oladigan aqlli robotlarni yaratish mumkin.

Ta’limda. Sun’iy intellekt baholashni avtomatlashtirishi mumkin, shunda o‘qituvchi dars berishga ko‘proq vaqt ajratadi. Sun’iy intellekt chatbot talabalar bilan o‘qituvchi yordamchisi sifatida muloqot qilishi mumkin. Kelajakda Sun’iy intellekt talabalar uchun shaxsiy virtual o‘qituvchi sifatida ishlashi mumkin, unga istalgan vaqtda va istalgan joyda osongina kirish mumkin.

Axloqiy muammolar. Sun’iy intellekt texnologiyasi taraqqiy etgani sayin, undan foydalanish bilan bog‘liq axloqiy muammolar tobora ko‘proq namoyon bo‘lmoqda. Masalan, ishga yollash yoki kredit berish kabi qarorlar qabul qilishda sun’iy intellektdan foydalanish haqida xavotirlar mavjud. Bundan tashqari, sun’iy intellekt tizimlarda mavjud noto‘g‘ri qarashlar va kamsitishlarni davom ettirish xavfi mavjud. Shu sababli, sun’iy intellekt tizimlari axloqiy va shaffof tarzda ishlab chiqilishi va amalga oshirilishini ta’minlash juda muhimdir.

Ishsizlik muammolari. Sun’iy intellekt rivojlanishidan kelib chiqadigan eng katta muammolardan biri bu ish joyini almashtirish xavfi. Mashinalar yanada aqlli va samarali bo‘lib, ular ko‘plab sohalarda inson ish joylarini egallashi mumkin. Bunda insonlar ish

joylarini yo‘qotishi jamiyatda ijtimoiy-iqtisodiy tengsizlikka sabab bo‘ladi. Shu sababli, ishchilarni o‘zgaruvchan mehnat bozoriga tayyorlash va ularga sun'iy intellekt bilan birga ishlash uchun zarur ko‘nikmalarni berish juda muhimdir.

Xavfsizlik muammolari. Sun'iy intellekt tizimlari kiberhujumlarga qarshi zaif bo‘lishi mumkin va bunday hujumlarning oqibatlari og‘ir bo‘lishi mumkin. Misol uchun, xakerlar sun'iy intellekt tizimlaridagi zaifliklardan foydalanib kerakli ma'lumotlarni o‘g‘irlash, qaror qabul qilishda manipulyatsiya qilish yoki jismoniy zarar etkazish uchun foydalanishi mumkin. Shu sababli, sun'iy intellekt tizimlarini kibertahdidlardan himoya qilish uchun mustahkam xavfsizlik choralarini ishlab chiqish juda muhimdir.

Huquqiy asoslanmaganligi. Sun'iy intellekt texnologiyasining jadal rivojlanishi me'yoriy-huquqiy bazaning o‘sishidan oshib ketdi. Bu sun'iy intellekt tizimlaridan jamiyat uchun zararli usullarda foydalanish xavfini tug‘diradi.

Sun'iy intellekt turlari. Sun'iy intellekt o‘rganishlar va bilimlarni qay darajada qo‘llash mumkinligiga qarab uchta qobiliyat turiga bo‘lish mumkin.

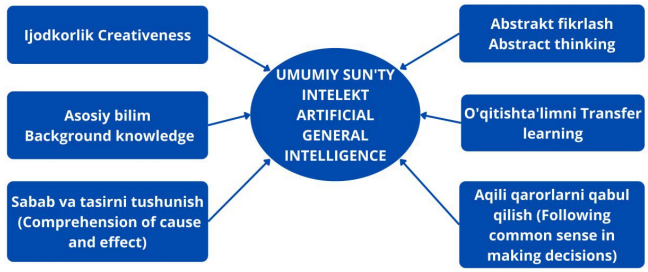
Zaif Sun'iy intellekt - bu aql bilan maxsus topshiriqni bajarish qobilyatiga ega bo‘lgan sun'iy intellektning bir turi. Sun'iy intellekt usullari ichida eng keng tarqalgan va hozirda mavjud bo‘lgan turi hisoblanadi. Zaif sun'iy intellekt o‘z sohasi yoki cheklovlaridan tashqarida ishlay olmaydi, chunki u faqat bitta aniq vazifani bajarish

uchun o‘rgatilgan. IBM kompaniyasining Watson superkompyuteri ham Zaif Sun’iy intellekt ostida ishlaydi, chunki u mashinani o‘rgatish va tabiiy tilni qayta ishlash bilan birgalikda ekspert tizimi yondashuvidan foydalanadi. Zaif sun’iy intellektga ba’zi misollar: shaxmat o‘ynash, elektron tijorat saytida takliflar sotib olish, o‘zini o‘zi boshqaradigan avtomobillar, nutqni aniqlash va tasvirni tanish.



1.2.1-rasm. Zaif sun’iy intellekt

Umumiy Sun’iy intellekt - bu inson kabi har qanday intellektual vazifani samarali bajara oladigan aql - idrok turi. Umumiy sun’iy intellekt tushunchasida aqlliroq bo‘lishi mumkin bo‘lgan va o‘z-o‘zidan inson kabi fikrlay oladigan tizim yaratish g‘oyasi mavjud. Hozirgi vaqtda umumiy sun’iy intellektda har qanday vazifani inson kabi mukammal bajara oladigan bunday tizim mavjud emas.



1.2.2-rasm. Umumiy Sun’iy intellekt

Kuchli sun’iy intellect – bu aql va kognitiv qobiliyatli, inson kabi vazifalarni bajarishga qodir bo‘lgan sun’iy intellekt ma’nosini bildiradi. Sun’iy intellektning bu darajasi ko‘pincha fikrlash, o‘rganish, muammolarni hal qilish, tabiiy tilni tushunish va yangi vaziyatlarga moslashish kabi insonga o‘xshash kognitiv qobiliyatlar bilan bog‘liq. Kuchli sun’iy intellekt kontekstni tushunish, tajribadan o‘rganish va sohalardagi muammolarni hal qilish uchun bilimlarni qo‘llash qobiliyatidir.



1.2.3-rasm. Kuchli sun’iy intellekt

Agar kuchli sun’iy intellekt ishlab chiqilsa, mashina inson kabi va undan yuqori fikrlash qobilyatiga ega bo‘ladi. Bunda sun’iy intellekt muammolarni hal qilish, o‘rganish va kelajakni bashoratlash qobiliyatiga ega bo‘lgan, o‘z-o‘zini anglay oladigan ong paydo bo‘ladi. Kuchli sun’iy intellekt inson ongidan farq qilib bo‘lmaydigan aqlli mashinalarni yaratishga qaratilgan.

Nazorat savollari

1. Zaif sun’iy intellekt qaysi turga kiradi?
2. Kuchli sun’iy intellekt qaysi turga kiradi?
3. Umumiy sun’iy intellekt qaysi turga kiradi?
4. Sun’iy intellekt bank sohasining qaysi bo‘limlarida qo‘llaniladi ?

I Bob bo‘yicha amaliy qism

Birinchi bob bo‘yicha amaliy qism quyidagi mavzular bo‘yicha keltirilgan. Bunda keltirilgan havola yoki QR kodni skaner qilish orqali GitHubga murojat qilib dasturiy kodlardan foydalanish imkoniyatiga ega bo‘lasiz.

https://github.com/TUITAI/

II BOB

MA’LUMOTLARNING INTELLEKTUAL TAHLILI, AGENTLAR VA EKSPERT TIZIMLARI

MA’LUMOTLARNING INTELLEKTUAL TAHLILI

Ushbu bobda agentlar, ma’lumotlarning intellektual tahlili va ekspert tizimlari keltirib o‘tilgan. Agentlar atrof-muhitga ta’sir qilishi mumkin bo‘lgan harakatlarni amalga oshiradi. Bunday harakatlarga boshqaruv tizimlari, qo‘l harakatlari, suhbatlashish, ma’lumotlarni ko‘rsatish yoki veb-saytga pochta buyrug‘ini yuborish kabilar kiradi. Ma’lumotlarning intellektual tahlili yoki ma’lumot qazib olish (Data Mining) sohasi kompyuter ilmi, statistika va sun’iy intellekt sohalarining kesishish nuqtasi bo‘lib, ma’lumotlar bazasi, ma’lumotlar tahlili va kompyuter texnologiyalarining birikmasi hisoblanadi. Ekspert tizimlari inson mutaxassisi kabi faktlar va evristik usullardan foydalangan holda murakkab muammolarni hal qilishda yordam beradigan algoritmlar hisoblanadi.

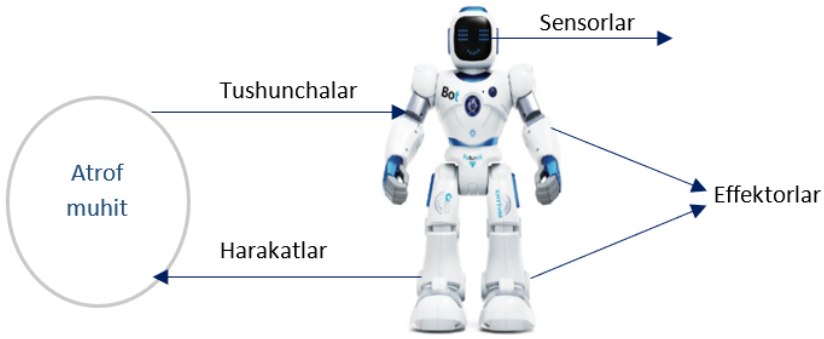
2.1. INTELLEKTUAL AGENT

Sun’iy intellekt tizimini ratsional agent va uning atrof-muhitini o‘rganish deb ta’riflash mumkin. Agentlar atrof-muhitni sensorlar orqali sezadilar va aktuatorlar orqali ta’sir ko‘rsatadi. Sun’iy intellekt agenti bilim, fikrlash, qaror qabul qilish va boshqa aqliy xususiyatlarga ega bo‘lishi mumkin. Agent idrok etish, fikrlash va harakat qilish siklida ishlaydi.

Intellektual agent – bu foydalanuvchi ma’lumotlari va tajribalari asosida qarorlar qabul qilishi yoki turli buyruqlarni amalga oshirishi mumkin bo‘lgan dastur.

Quyida sun’iy intellekt agenti uchun asosiy to‘rtta qoida keltirish mumkin:

Agent atrof-muhitni idrok etish qobiliyatiga ega bo‘lishi kerak.
Qaror qabul qilish uchun kuzatishdan foydalanish kerak.
Qaror harakatga olib kelishi kerak.
Agent amalga oshirgan harakat intellektual bo‘lishi kerak.



2.1.1-rasm. Intellektual agent

Agentlarning quyidagi ko‘rinishlari mavjud:

Inson agenti - inson ko‘zlari, quloqlari va boshqa organlari.

Robot agenti - kameralar, infraqizil diapazonni topuvchi sensorlar, tabiiy tilni qayta ishlash va aktuatorlar uchun turli xil motorlar bo‘lishi mumkin.

Sensor - bu atrof-muhit o‘zgarishini aniqlaydigan va ma’lumotni boshqa elektron qurilmalarga yuboradigan qurilma. Agent o‘z muhitini o‘z navbatida sensorlar orqali kuzatadi.

Aktuatorlar - energiyani harakatga aylantiradigan mashinalarning tarkibiy qismidir. Aktuatorlar faqat tizimni harakatlantirish va boshqarish uchun javobgardir va ular elektr motor, tishli, relslar va hokazo bo‘lishi mumkin.

Effektorlar - atrof-muhitga ta’sir qiluvchi qurilmalardir. Effektorlar oyoqlar, g‘ildiraklar, qo‘llar, barmoqlar va displey ekrani bo‘lishi mumkin.

Intellektual agentlar ikki katta guruhga bo‘linadi: dasturiy ta’minot agentlar va jismoniy agentlar.

Dasturiy ta’minot agenti - dasturiy ta’minot agenti sensorli kirish sifatida tugmachalarni bosishi, fayl tarkibiga ega bo‘lishi va ushbu kirishlar bo‘yicha harakat qilishi va ekranda chiqishi kerak bo‘ladi. Shuning uchun atrofimizdagi dunyo, uyali telefon, kamera kabi agentlarga to‘la va hatto biz ham agentmiz.

Jismoniy agent *(robot)* - bu turli xil vazifalarni bajarish uchun ishlatilishi mumkin bo‘lgan dasturlashtiriladigan tizim, masalan, oddiy robotlar payvandlash yoki bo‘yash kabi turli xil muntazam ishlarni bajarish uchun ishlab chiqarish sohalarida ishlatilishi mumkin. Ba’zi tashkilotlar pochta yoki yozishmalarni turli xonalarga tarqatish kabi oddiy yetkazib berish ishlarini bajarish uchun mobil robotlardan foydalanadilar.

Sun’iy intellekt agentining tuzilishi. Sun’iy intellektning vazifasi agent funksiyasini amalga oshiradigan dasturini ishlab chiqishdir. Intellektual agentning tuzilishi arxitektura va agent dasturining kombinatsiyasidir. Buni quyidagicha ko‘rish mumkin:

$$Agent = Arxitektura + Agent\ dasturi$$

Quyida sun’iy intellekt agenti tuzilishiga tegishli uchta asosiy atama keltirilgan:

Arxitektura - bu sun’iy intellekt agenti bajaradigan mashina.

Agent funksiyasi - idrokni harakatga solishtirish uchun ishlatiladi.

$$f{:}P^* \rightarrow A$$

Agent dasturi- agent funksiyasini amalga oshirishdir. Agent dasturi f funksiyasini ishlab chiqarish uchun jismoniy arxitekturada ishlaydi.

SAAS (PEAS) - bu sun’iy intellekt agenti asosida ishlaydigan model turi. Sun’iy intellekt agenti yoki ratsional agentni

aniqlaganimizda, biz uning xususiyatlarini PEAS modeli yordamida guruhlashimiz mumkin. U to‘rtta so‘zdan iborat:

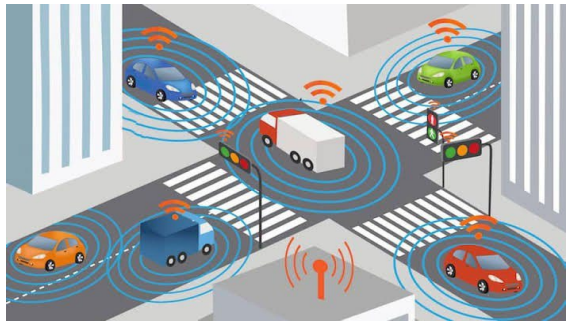
S *(P) - samaradorlik o‘lchovi (Performance measure)*

A *(E) - Atrof-muhit (Environment)*

A *(A) - Aktuatorlar (Actuators)*

S *(S) - Sensorlar (Sensors)*

Bu yerda samaradorlik o‘lchovi agentning xatti-harakati muvaffaqiyatining maqsadi hisoblanadi.



2.1.2-rasm. Aqlli avtomobillarda SAAS (PEAS) algoritmi ifodasi

Faraz qilaylik, o‘zini o‘zi boshqaradigan mashina: unda SAAS modeli quyidagicha bo‘ladi:

Ishlash- xavfsizlik, vaqt, qonuniy harakat, qulaylik.

Atrof-muhit-yo‘llar, boshqa transport vositalari, yo‘l belgilari.

Aktuatorlar- rul boshqaruvi, tezlatgich, tormoz, signal, ovoz signali.

Sensorlar- Kamera, GPS, tezlik o‘lchagich, odometr, akselerometr

2.1.1.
Intellektual agent turlari

Sun’iy intellekt agentlarining turlari. Agentlarni idrok etilgan aql va qobiliyat darajasiga qarab beshta turga ajratish mumkin. Bu agentlarning barchasi o‘z ish faoliyatini yaxshilashi va vaqt o‘tishi bilan yaxshiroq imkoniyatlar yaratishi mumkin. Bular quyida keltirilgan:

Oddiy refleks agent

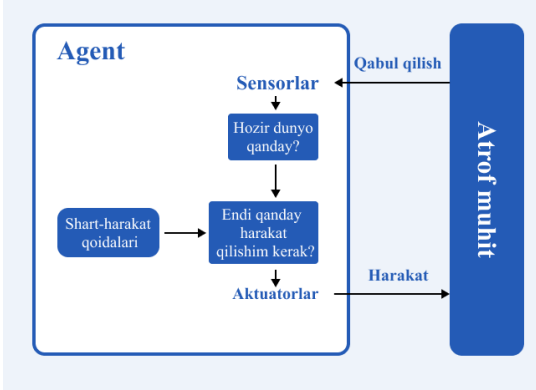
Modelga asoslangan refleks agent

Maqsadga asoslangan agentlar

Foydali dasturlarga asoslangan agent

O‘quv agenti

Oddiy refleks agent - eng oddiy agentlardir. Ushbu agentlar joriy idrok asosida qaror qabul qiladilar va idrok tarixining qolgan qismini e’tiborsiz qoldiradi. Ushbu agentlar faqat to‘liq kuzatiladigan muhitda muvaffaqiyatli ishlaydi. Oddiy refleks agenti qaror va harakat jarayonida avvalgi biror-bir ma’lumotlarni hisobga olmaydi. Simple refleks agenti shartli harakat qoidasida ishlaydi, ya’ni u joriy holatni harakatga moslashtiradi. Xonani tozalash vositasi kabi, u faqat xona tartibsiz bo‘lsa ishlaydi. Oddiy refleks agentini loyihalash yondashuvidagi muammolar quyidagilar. Ular juda cheklangan aqlga ega. Ular hozirgi holatning sezilmaydigan qismlari haqida ma’lumotga ega emaslar.



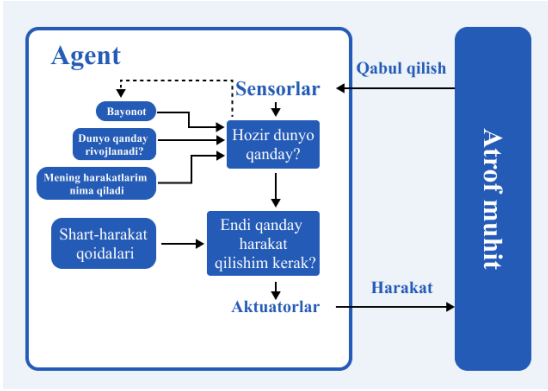
2.1.3-rasm. Oddiy refleks agent strukturasi

Modelga asoslangan refleks agent- bu agent qisman kuzatiladigan muhitda ishlashi va vaziyatni kuzatishi mumkin. Modelga asoslangan agent ikkita muhim omilga ega.

1) Model - "dunyoda voqealar qanday sodir bo'lishi" haqidagi bilimdir, shuning uchun u modelga asoslangan agent deb ataladi.

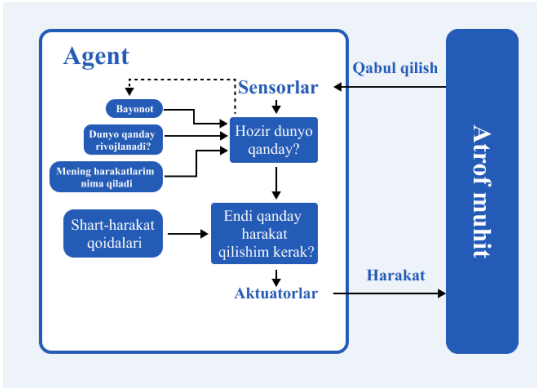
2) Ichki holat - bu idrok tarixiga asoslangan hozirgi holatning ifodasidir. Ushbu agentlar "dunyoni bilish" modeliga ega va modelga asoslanib, ular harakatlarni amalga oshiradilar.

Agent holatini yangilash uchun quyidagilar haqida ma'lumot kerak:

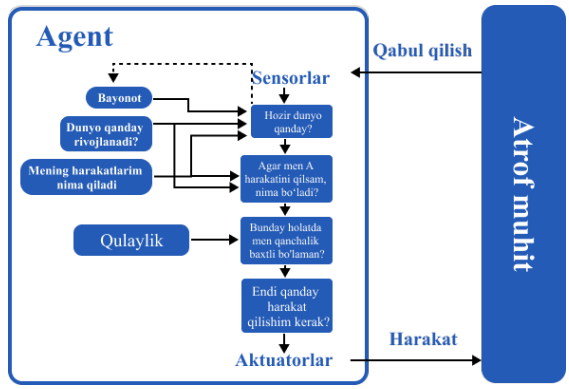


2.1.4-rasm. Modelga asoslangan refleks agent strukturasi

Maqsadga asoslangan agentlar - mavjud vaziyatni bilish har doim ham agentga nima qilish kerakligini hal qilish uchun yetarli emas. Agent kerakli vaziyatlarni tavsiflovchi maqsadini bilishi kerak. Maqsadga asoslangan agentlar “maqsad” ma’lumotlariga ega bo‘lish orqali modelga asoslangan agentning imkoniyatlarini kengaytiradi. Ular maqsadga erishish uchun harakatni tanlaydilar. Ushbu agentlar maqsadga erishiladimi yoki yo‘qmi, hal qilishdan oldin mumkin bo‘lgan harakatlar ketma-ketligini ko‘rib chiqishlari kerak.



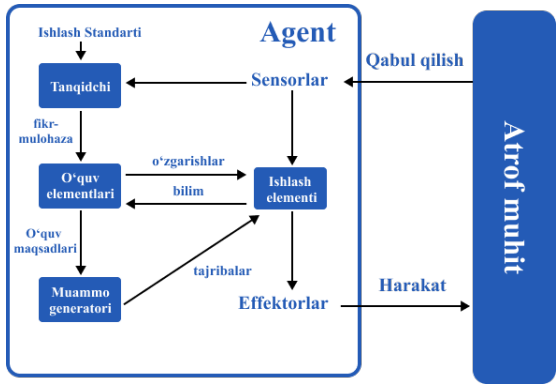
Foydali dasturlarga asoslangan agentlar. Ushbu agentlar maqsadga asoslangan agentga o‘xshaydi, lekin foydalilikni o‘lchashning qo‘shimcha komponentini ta’minlaydi, bu ularni ma’lum bir holatda muvaffaqiyat o‘lchovini ta’minlash orqali farq qiladi. Yordamchi funksiya har bir harakat maqsadga qanchalik samarali erishayotganini tekshirish uchun har bir holatni haqiqiy raqam bilan taqqoslaydi.



2.1.6-rasm. Modelga asoslangan refleks agent strukturasi

O‘quv agentlari - bu o‘tmish tajribasidan o‘rganishi mumkin bo‘lgan yoki o‘rganish qobiliyatiga ega bo‘lgan agent turi. U asosiy bilimlar bilan harakat qila boshlaydi va keyin o‘rganish orqali avtomatik ravishda harakat qilish va moslashishga qodir.

Shunday qilib, o‘quv agentlari o‘rganish, ishlashni tahlil qilish va ishlashni yaxshilashning yangi usullarini izlash imkoniyatiga ega.



2.1.7-rasm. O‘quv agentlariga asoslangan refleks agent strukturasi

Agent muhiti. Atrof-muhit-bu agentni o‘rab turgan dunyodagi barcha narsalar hisoblanadi, lekin u agentning o‘zi emas. Atrof-muhitni agent mavjud bo‘lgan vaziyat deb ta’riflash mumkin. Atrof-muhit agent yashaydigan, faoliyat yuritadigan va agentga uni his qilish va unga amal qilish uchun biror ma’lumot beradigan joy.

Atrof-muhitning xususiyatlari. Rassell va Norvigga fikriga ko‘ra, atrof-muhit agenti turli xil xususiyatlarga ega bo‘lishi mumkin:

To‘liq kuzatiladigan va qisman kuzatiladigan muhit. Agar agent sensori vaqtning har bir nuqtasida atrof-muhitning to‘liq holatini sezsa yoki unga kira olsa, u **to‘liq kuzatiladigan** muhitdir, aks holda u **qisman kuzatiladi**. To‘liq kuzatiladigan muhit oson, chunki dunyo tarixini kuzatib borish uchun ichki holatni saqlashga hojat yo‘q. Barcha muhitda sensori bo‘lmagan agent muhiti **kuzatilmaydigan** deb ataladi.

Deterministik va stoxastik agent. Agar agentning hozirgi holati va tanlangan harakati atrof-muhitning keyingi holatini to‘liq aniqlay olsa, unda bunday muhit deterministik muhit deb ataladi. Stoxastik

muhit tabiatda tasodifiy bo‘lib, uni agent tomonidan to‘liq aniqlash mumkin emas. Deterministik, to‘liq kuzatiladigan muhitda agent noaniqlik haqida tashvishlanishga hojat yo‘q.

Diskret va uzluksiz. Agar muhitda uning ichida bajarilishi mumkin bo‘lgan cheklangan miqdordagi hislar va harakatlar mavjud bo‘lsa, unda bunday muhit diskret muhit deb ataladi, aks holda u uzluksiz muhit deb ataladi. Shaxmat o‘yini diskret muhitga kiradi, chunki bajarilishi mumkin bo‘lgan cheklangan sonli harakatlar mavjud. O‘z-o‘zini boshqaradigan mashina uzluksiz muhitga misol bo‘la oladi.

Nazorat savollari:

1. Sun’iy intellektda nechta agent turi mavjud?
2. Oddiy refleks agent turi qanday ma’no anglatadi?
3. Sun’iy intellektda dasturiy ta’minot agenti qanday ma’no anglatadi?
4. Statik va dinamik agent qanday ma’no anglatadi?

2.1-bo‘lim bo‘yicha amaliy qism

2.2-bo‘lim bo‘yicha amaliy qism quyidagi mavzular bo‘yicha keltirilgan. Bunda keltirilgan havola yoki QR kodni skaner qilish orqali GitHubga to‘g‘ridan to‘g‘ri murojat qilish imkoniyatiga ega bo‘lasiz.

https://github.com/TUITAI/

2.2. MA’LUMOTLARNING INTELLEKTUAL TAHLILI

Ma’lumotlarning intellektual tahlili va (Data Mining) termini odatda ikki xil talqin qilinadi. Bu ikki jarayon ham katta hajmdagi ishlov berilmagan ma’lumotlarning keraklisini ajratib olish, tadqiq etish va izlab topish uchun qo‘llaniladi. Ma’lumotlarni intellektual tahlil qilish termini ba’zan ma’lumotlarni qazib chiqarish, ma’lumotlarni tortib chiqarish, ma’lumotlarni kovlab olish, ma’lumotlarni intellektual tahlili, qonuniyatlarni topish muhiti, bilimlarni kengaytirish, bilimlarni kovlab olish kabi tarjima qilinib Data Mining (DM) atamasi bilan sinonim deb ham hisoblash mumkin.

2.2.1. Ma’lumotlar intellektual tahlilning rivojlanish bosqichlari

Ma’lumotlarning intellektual tahlilining rivojlanishiga zamonaviy texnalogiyalar va ilmiy taraqqiyot asosiy omil bo‘lib xizmat qilib kelmoqda. Bu sohada tarixan quyidagi bosqichlarni ko‘rish mumkin.

Paydo bo‘lishi (1950-1970 yillar):

Statistika usullari. Ma’lumotlarni tahlil qilish sohasidagi dastlabki usullar asosiy statistik tahlilga asoslangan bo‘lib, ularni o‘z ichiga regressiya va korrelyatsiya kabi algoritmlari kiradi. Bu davrda

dastlabki kompyuterlarning vujudga kelishi va ma’lumotlarni qayta ishlash va tahlil qilish muhim omil hisoblangan.

Rivojlanish davri (1980-1990 yillar):

Ekspert tizimlari va sun’iy intellekt. Ekspert tizimlari va sun’iy intellekt texnologiyalari ma’lumotlarni tahlil qilish va o‘rganish uchun yangi imkoniyatlarni yaratdi. Bundan tashqari ma’lumotlar bazasi texnologiyalarining rivojlanishi ma’lumotlarni samarali saqlash va ajratib olishga imkon berdi. Bu ma’lumotlarni intellektual tahlil qilish sohasining yangi bosqichga chiqishiga katta yordam berdi.

Data Mining (DM) sohasining paydo bo‘lishi (1990-2000 yillar). 1990-yillarda “data mining” (ma’lumot qazib olish) atamasi paydo bo‘ldi, hamda ma’lumotlar tahlili va intellektual tahlil uchun yangi metodologiyalar va usullarni o‘zida mujassamlashtirdi. Ma’lumotlar hajmi oshib borishi bilan ma’lumot qazib olish (DM) usullari katta ma’lumotlar (big data) bilan ishlash imkoniyatini oshirib bordi. Bugungi kunga kelib ma’lumotlarni yig‘ish va saqlash, qolaversa ularni tahlil qilish orqali sohalarni rivojlanishida foydalanish muhim omil hisoblanib va ular hajmining keskin oshishiga sabab bo‘lmoqda.



2.2.1-rasm. Katta xajmli ma’lumotlarning sohalarda qo‘llanilishi

Mashinali o‘qitish va chuqur o‘qitish usullarining rivojlanishi (2000 yildan keyin).

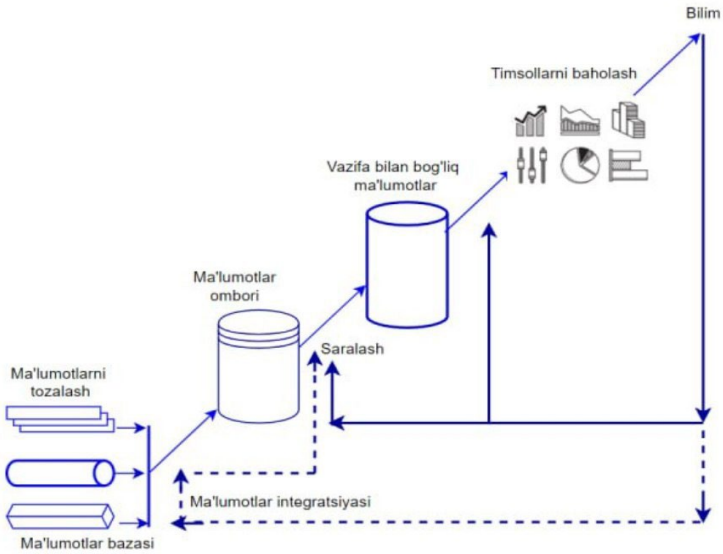
Mashinali o‘qitish. Bu davrda mashinali o‘qitish texnologiyalari taraqqiy etib, klassifikatsiya, regressiya, klasterlash va boshqa usullar bilan ma’lumotlarni intellektual tahlil qilish imkoniyatini yaratdi.

Chuqur o‘qitish. Sun’iy neyron tarmoqlar (ANN) va chuqur o‘qitish (deep learning) sohalari 2010-yillardan boshlab keng tarqaldi va ma’lumotlarni intellektual tahlil qilishda innovatsion usullarni taklif etdi.

Bugungi kunda. *Katta hajimli ma'lumotlar va real vaqt rejimidagi tahlil.* Hozirgi davrda katta hajimli ma'lumotlar (big data) va real vaqtli tahlil (real-time analysis) ma'lumotlarni intellektual tahlil qilishda muhim o'rin egallamoqda. Bugungi kunga kelib ma'lumotlarning intellektual tahlili biznes, marketing, tibbiyot, moliya va boshqa ko'plab sohalarda qo'llanilishi sohaning rivojlanish tendensiyasini yangi bosqichga olib chiqdi.

Ma'lumotlarni dastlabki qayta ishlash (Data processing)

Ma'lumotlarni dastlabki qayta ishlash-bu ma'lumotlarni intellektual tahlil qilish uchun tayyorlash jarayonidir. Ushbu jarayon ma'lumotlar sifatini oshirish, noaniqliklarni bartaraf etish va tahlil natijalarini yaxshilashga yordam beradi. Bundan tashqari ma'lumotlarga dastlabki ishlov berish asosida bilim hosil qilinadi. Bilimlarni hosil qilish-bu qazib olingan ma'lumotlarni foydalanuvchiga taqdim etish uchun eng kerakli ma'lumotlarni aniq va tushunarli tarzda ifodalash jarayonidir. Ma'lumotlarni dastlabki qayta ishlash mashinali o'qitish(ML) usullari uchun muhim hisoblanadi, chunki u ML modellarini yaratish va o'qitishda foydalanish uchun kerakli ma'lumotlarni tayyorlaydi. Ma'lumotlarni qayta ishlashning maqsadi ma'lumotlarni tozalash, o'zgartirish va modellashtirish uchun mos formatga keltirishdir.



2.2.2-rasm. Ma’lumotlar intellektual tahlil (Data Mining) jarayoni bosqichlari

Ma’lumotlarni dastlabki qayta ishlash jarayoni quyidagi bosqichlarni o‘z ichiga oladi:

- Ma’lumotlarni tushunish;
- Ma’lumotlarni tayyorlash
- Ma’lumotlarni tozalash

Ma’lumotlarni tushinish (data understanding)

Ma’lumotlarni tushunish ma’lumotlar tahlilining asosiy bosqichlaridan biridir. Bu bosqichda ma’lumotlar to‘plamini tanib olish, uning xususiyatlarini ajratib olish, ma’lumotlar tuzilishini tahlil qilish uchun ularni qanday tayyorlash kerakligi to‘g‘risida tahlillar olib boriladi. Ma’lumotlar ustida tahlil olib borish va tushunish uchun

ularning turlari to‘g‘risida ham yetarlicha ma’lumotga ega bo‘lishimiz zarur. Ma’lumotlar to‘plamining quyidagi turlari mavjud:

Yozib olingan ma’lumotlar - ma’lumotlar matritsasi, hujjat ma’lumotlari va tranzaksiya ma’lumotlariga bo‘linadi.

Ma’lumotlar matritsasi. Agar ma’lumotlar obyektlari bir xil raqamli atributlar to‘plamiga ega bo‘lsa, u holda ma’lumotlar obyektlarini ko‘p o‘lchovli fazodagi nuqtalar sifatida ko‘rish mumkin, har bir o‘lchov boshqa atributni ifodalaydi. Bunday ma’lumotlar to‘plami *m* dan *n* gacha bo‘lgan *m* qatorli, har bir obyekt uchun bittadan va har bir atribut uchun bittadan *n* ta ustunli matritsa bilan ifodalanishi mumkin.

Hujjat ma’lumotlari. Har bir hujjat “termin” vektoriga aylanadi. Har bir atama vektorning komponenti (atributi) hisoblanadi. Har bir komponentning qiymati tegishli atamaning hujjatda necha marta paydo bo‘lishidir.

Tranzaksiya ma’lumotlari. Yozuv ma’lumotlarining maxsus turi, bu yerda har bir yozuv (tranzaksiya) bir qator elementlarni o‘z ichiga oladi. Misol uchun, oziq-ovqat do‘konini ko‘rib chiqaylik. Xaridor tomonidan bitta xarid safari davomida sotib olingan mahsulotlar to‘plami tranzaktsiyani tashkil qiladi, sotib olingan alohida mahsulotlar esa tovarlardir.

Maxsus belgilangan ma’lumotlar. Fazoviy ma’lumotlar. Vaqtinchalik ma’lumotlar. Ketma-ket ma’lumotlar. Genetik ketma-ketlik ma’lumotlari.

Atributlar. Atributlarning har xil turlari mavjud. Nominal/kategorik cheklangan sohadagi atributlarning qiymatlari, toifalar, “narsalar nomlari”. Ikkilik-faqat ikkita holatga ega nominal atribut (0 va 1). Simmetrik ikkilik-ikkala natija ham bir xil darajada muhim (masalan, jins). Asimmetrik ikkilik-natijalar bir xil darajada muhim emas. (masalan, ijobiy yoki salbiy tibbiy test).

Yuqoridagi ma’lumot turlarini o‘rganish natijasida ma’lumotlarning (data understanding) tushunarli bo‘lishi uchun quyidagi misolni ko‘rib chiqamiz.

Misol: datasetdagi atributlardan foydalanib, ma’lumotlarni vizuallashtirish, jadval ko‘rinishga keltirish, bir biriga bog‘liq ma’lumotlar grafigini chizish va statistik tahlil o‘tqazish jarayonini amalga oshiramiz. Buning uchun *kaggle.com* saytidan avtomabillar datasetini(*cars.csv*) yuklab olamiz.

2.1-jadval. Ma’lumotlar to‘plami

Car ID	Brand	Model	Yil	Yurgan yo`li	Yoqilg`i turi	Uzatmalar qutisi	Egasi turi	Mileage	Dvigatel	Ot kuchi	O'rindiqlar	Narxi
1	Toyota	Corolla	2018	50000	Petrol	Manual	First	15	1498	108	5	800000
2	Honda	Civic	2019	40000	Petrol	Automatic	Second	17	1597	140	5	1000000
3	Ford	Mustang	2017	20000	Petrol	Automatic	First	10	4951	395	4	2500000
4	Maruti	Swift	2020	30000	Diesel	Manual	Third	23	1248	74	5	600000
5	Hyundai	Sonata	2016	60000	Diesel	Automatic	Second	18	1999	194	5	850000
6	Tata	Nexon	2019	35000	Petrol	Manual	First	17	1198	108	5	750000
7	Mahindra	Scorpio	2018	45000	Diesel	Automatic	Second	15	2179	140	7	900000
8	Volkswagen	Polo	2020	25000	Petrol	Automatic	First	18	999	76	5	650000
9	Audi	A4	2017	30000	Diesel	Automatic	First	18	1968	187	5	2200000
10	BMW	X1	2019	20000	Diesel	Automatic	Second	20	1995	190	5	2700000
11	Mercedes	C-Class	2018	28000	Petrol	Automatic	First	16	1991	181	5	2300000
12	Ford	Endeavour	2017	35000	Diesel	Automatic	Second	12	2198	158	7	2000000
13	Hyundai	Creta	2019	22000	Petrol	Manual	Third	16	1497	113	5	850000
14	Tata	Harrier	2018	40000	Diesel	Automatic	First	17	1956	167	5	1600000
15	Maruti	Ertiga	2020	18000	Petrol	Manual	First	19	1462	103	7	850000
16	Honda	City	2017	42000	Diesel	Manual	Second	25	1498	98	5	650000
17	Volkswagen	Tiguan	2018	32000	Diesel	Automatic	First	17	1968	141	5	1800000
18	Audi	Q3	2016	38000	Petrol	Automatic	Second	15	1395	148	5	1900000
19	BMW	5 Series	2019	24000	Diesel	Automatic	First	18	1995	187	5	3000000
20	Mercedes	GLC	2017	26000	Petrol	Automatic	Second	12	1991	241	5	2500000

Bu dataset yordamida har bir avtomabil modeli yurgan yo‘lining statistik ma’lumotini aniqlab jadval ko‘rinishida ifoda etamiz.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt


# Ma’lumotlar to‘plamini yuklang (“cars.csv” xuddi shu katalogda mavjud bo‘lsa)
df = pd.read_csv('cars.csv')
# DataFrame-da "Model" va "Yurgan yo'li" ustunlari mavjudligini tekshiring


if 'Model' in df.columns and 'Yurgan yo\'li' in df.columns:
    # "Yurgan yo'li" raqamli ustun ekanligiga ishonch hosil qiling
    df['Yurgan yo\'li'] = pd.to_numeric(df['Yurgan yo\'li'], errors='coerce')
    # "Yurgan yo'li" ustunida qiymatlari etishmayotgan qatorlarni tashlang
    df.dropna(subset=['Yurgan yo\'li'], inplace=True)

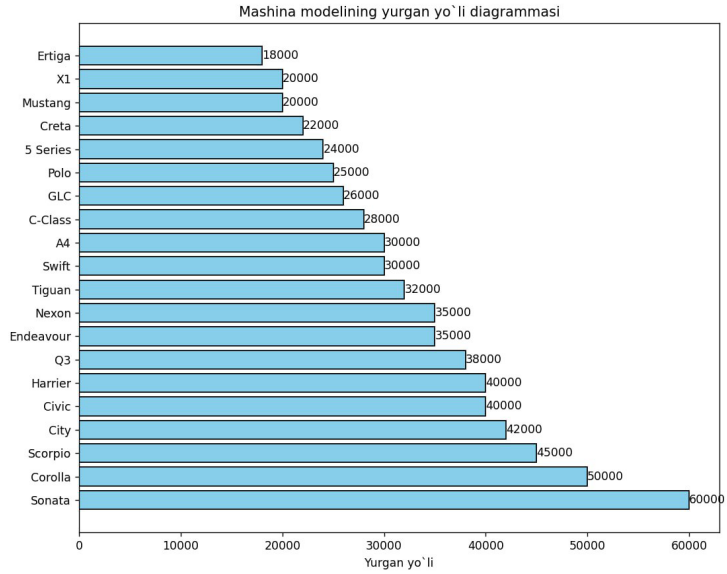

    # DataFrame-ni "Yurgan yo'li" bo‘yicha tartiblang
    df.sort_values('Yurgan yo\'li', inplace=True)
     models = df['Model']
    kilometers_driven = df['Yurgan yo\'li']


    # Diagramma yaratish
    fig, ax = plt.subplots(figsize=(10, 8))
```

```
    y_pos = np.arange(len(models))
    bars = ax.barh(y_pos, kilometers_driven, align='center',
color='skyblue', edgecolor='black')
     ax.set_yticks(y_pos)
    ax.set_yticklabels(models)
    ax.invert_yaxis()  # Y tengligini o'zgartirish (yuqoridan pastga
qarab)
    ax.set_xlabel('Yurgan yo`li')
    ax.set_title('Mashina modelining yurgan yo`li diagrammasi')

    # Har bir bar uchun qiymatlarni text shaklida qo'shish
    for bar, km in zip(bars, kilometers_driven):
        ax.text(bar.get_width(), bar.get_y() + bar.get_height()/2, f'{km}',
va='center', ha='left', fontsize=10, color='black')

    plt.show()
else:
    print("Ma'lumotlar to‘plamida \"Model\" yoki \"Yurgan yo'li\"
ustuni mavjud emas.")
```

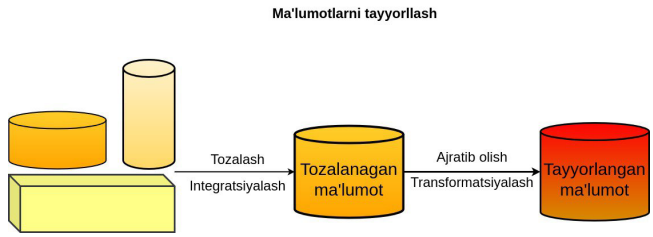


2.2.3-rasm. Avtomabillar yurgan yo‘lining statistik ma’lumotlarining diagrammasi

Yuqoridagi diagrammada avtomabillar modellarining yurgan yo‘llari ketma-ket holatda joylashgan ma’lumotlari vizual ko‘rinishda ifodalangan. Bunda aralash kelgan ma’lumotlarni tushunarli holatga o‘tkazish jarayoni keltirib o‘tilgan.

Ma’lumotlarni tayyorlash (data preparation)

Ma’lumotlarni tayyorlash (data preparation) yoki ma’lumotlarni oldindan qayta ishlash, ma’lumotlarni tahlil qilish va modellash uchun moslashtirish jarayonidir. Ma’lumotlarni tayyorlash muhim bosqich bo‘lib, u ma’lumotlar to‘plamini toza, barqaror va ishonchli qilishga yordam beradi.



2.2.4-rasm. Ma’lumotlarni tayyorlash jarayoni

Ma’lumotlarni tayyorlash turlari:

- Birlashtirish
- Ma’lumotlarni qisqartirish (o‘quv tanlanma)
- O‘lchamlarni qisqartirish
- Xususiyatlar to‘plamini tanlash
- Xususiyatlar yaratish
- Diskretlashtirish va Binarizatsiya
- Atributlarni o‘zgartirish

Birlashtirish. Ikki yoki undan ortiq atributlarni (yoki obyektlarni) bitta atributga (yoki obyektga) birlashtirish.

Ma’lumotlarni qisqartirish. Ma’lumotlar hajmini kamaytirish.

– Yozuvlar sonini kamaytirish
  - Ma’lumotlardan tanlanma olish
  - Klasterlash
– Ustunlar (atributlar) sonini kamaytirish
  - Atributlar to‘plami
  - Yangi (kichikroq) atributlar to‘plamini yaratish.
– Tanlanmalarni ajratib olish

- Tanlanma olish ma'lumotlarni qisqartirish uchun qo'llaniladigan asosiy texnikadir

Tanlanma olish odatda Data Miningda qo'llaniladi, chunki kerakli ma'lumotlarning butun to'plamini qayta ishlash juda qimmat yoki ko'p vaqt talab qiladi.

Kerakli tanlanma olishning asosiy printsipi quyidagilardan iborat:

- Namunadan foydalanish, agar namuna vakili bo'lsa, deyarli butun ma'lumotlar to'plamidan foydalangandek ishlaydi.
- Agar namuna dastlabki ma'lumotlar to'plami bilan taxminan bir xil mos bo'lsa, namuna vakili hisoblanadi.

Misol uchun: Katta o'lchamdagi tasvir ma'lumotlarni qisqartirishni ko'rib chiqamiz.

```
import numpy as np
from PIL import Image
import matplotlib.pyplot as plt
class TasvirSiqish:
    def init(self, file_path):
        self.image = Image.open(file_path).convert('L')
        self.image_array = np.array(self.image)
    def average_pooling(self, block_size):
        h, w = self.image_array.shape
        new_h = h // block_size
        new_w = w // block_size
        reduced_image = np.zeros((new_h, new_w),
dtype=np.float32)
```

```
        for i in range(new_h):
            for j in range(new_w):
                block =
                    self.image_array[i*block_size:(i+1)*block_size,
                    j*block_size:(j+1)*block_size]
                reduced_image[i, j] = np.mean(block)
        return reduced_image
    def tasvir_korish(self, block_sizes):
        fig, axes = plt.subplots(1, len(block_sizes) + 1, figsize=(15,
            5))
        # Dastlabki tasvirni ko`rish
        axes[0].imshow(self.image_array, cmap='gray')
        axes[0].set_title('Dastlabki holatdagi
tasvir(1024x1024)')
        axes[0].axis('off')
        # Siqilgan tasvirlarni ko`rish
        for i, block_size in enumerate(block_sizes):
            compressed_image = self.average_pooling(block_size)
            axes[i+1].imshow(compressed_image, cmap='gray')
                axes[i+1].set_title(f'{block_size} marta qisqartirilgan
                holat
                ({int(1024/block_size)}x{int(1024/block_size)})')
            axes[i+1].axis('off')
        plt.show()
# File path of the image
file_path = 'image1.jpg‘
# Tasvirni siqish o`lchami.
block_sizes = [8, 16]
```

```
siqish = TasvirSiqish(file_path)
# Tasvirning dastlabki va qisqargandan keyingi holatni ko`rish`
siqish.tasvir_korish(block_sizes)
```

Yuqoridagi dasturiy algoritm yordamida tasvirni quyidagicha qisqartirish jarayonini ko‘rib chiqamiz:



2.2.5-rasm. Tasvirni qisqartirish

Tasvir piksellar soni 1024x1024 matritsa ko‘rinishidan, 64x64 matritsa ko‘rinishiga keltirib olindi. Bunda pythonning tayyor kutubxonalari yordamida tasvirni siqish va bu orqali esa ma’lumotlar hajmini kamaytirish amalga oshirildi.

O‘lchamlarni qisqartirish- bu imkon qadar kichik va ma’lumotlarni tahlil qilish uchun yetarli bo‘lgan atributlar to‘plamini tanlashdir. Bundan tashqari ahamiyatsiz xususiyatlarni (ko‘p yoki kamroq) olib tashlash ham deyiladi. Misol: do‘kondagi tovarlarning kunlik sotilish hajmi statistikasini baholash uchun, ta’minotchi korxonaning nomi ahamiyatsiz. Bu misolni quyidagicha tahlil qilib chiqamiz.

Ma’lumotlar to‘plamini hosil qilib olish.

```
import pandas as pd
```

```
from datetime import datetime
data = {
    'ID': [0, 1, 2, 3, 4, 5],
    'Nomi': ['Pepsi', 'Fanta', 'Sprite', 'CocaCola', 'Lipton', 'FuseTea'],
    'Shartnoma_nomer': [3434, 5322, 2434, 2133, 6432, 3210],
    'Taminotchi_korxona': ['Pepsi MCHJ', 'CocaCola MCHJ',
      'Pepsi MCHJ', 'CocaCola MCHJ', 'CocaCola MCHJ', 'Pepsi
      MCHJ'],
    'Narxi': [15000, 14000, 16000, 15000, 10000,  9000],
    'Kelgan_Mahsulot_soni': [240, 180, 120, 240, 60, 60],
    'Sotilgan_Mahsulot_soni': [85, 50, 40, 105, 15, 20],
    'Kelgan_sana': ['2024, 5, 19', '2024, 5, 18', '2024, 5, 18', '2024,
      5, 17', '2024, 5, 15', '2024, 5, 10'],
    'Muddati_oy': [12, 12, 12, 12, 12, 12]}
df = pd.DataFrame(data)
print(df)
def save_to_excel(dataframe, file_name):
    dataframe.to_excel(file_name, index=False)
    print(f"Data saved to {file_name}")
# DataSet-ni Excel fayliga saqlang
save_to_excel(df, 'guruhlangan_data.xlsx')
```

2.2-jadval. Dataset

ID	Nomi	Shartnoma nomer	Taminotchi korxona	Narxi	Kelgan Mahsulot soni	Sotilgan Mahsulot soni	Kelgan sana	Muddati oy
0	Pepsi	3434	Pepsi MCHJ	15000	240	85	2024, 5, 19	12
1	Fanta	5322	CocaCola MCHJ	14000	180	50	2024, 5, 18	12
2	Sprite	2434	Pepsi MCHJ	16000	120	40	2024, 5, 18	12
3	CocaCola	2133	CocaCola MCHJ	15000	240	105	2024, 5, 17	12
4	Lipton	6432	CocaCola MCHJ	10000	60	15	2024, 5, 15	12
5	FuseTea	3210	Pepsi MCHJ	9000	60	20	2024, 5, 10	12

Yuqoridagi datasetdan bizga kerak bo'lmagan ma'lumotlarni olib tashlaymiz. Buning uchun yuqoridagi dasturiy algoritmni davom ettiramiz.

```
columns_needed = ['Narxi', 'Kelgan_Mahsulot_soni',
'Sotilgan_Mahsulot_soni', 'Kelgan_sana', 'Muddati_oy']
data2 = df[columns_needed]
print(data2)
def save_to_excel(dataframe, file_name):
    dataframe.to_excel(file_name, index=False)
    print(f"Data saved to {file_name}")

# DataSet-ni Excel fayliga saqlang
save_to_excel(df, 'guruhlangan_data2.xlsx')
```

2.3-jadval. O‘lchamlari qisqartirilgan ma’lumotlar to‘plami

Nomi	Narxi	Kelgan Mahsulot soni	Sotilgan Mahsulot soni	Kelgan sana	Muddati oy
Pepsi	15000	240	85	2024, 5, 19	12
Fanta	14000	180	50	2024, 5, 18	12
Sprite	16000	120	40	2024, 5, 18	12
CocaCola	15000	240	105	2024, 5, 17	12
Lipton	10000	60	15	2024, 5, 15	12
FuseTea	9000	60	20	2024, 5, 10	12

Dasturiy algoritm yordamida bizga kerakli datasetni hosil qilib olindi.

Xususiyat yaratish. Ma’lumotlar to‘plamidagi muhim ma’lumotlarni asl atributlarga qaraganda ancha samaraliroq qamrab oladigan yangi atributlarni yaratish hisoblanadi. Masalan: zichlik atributini olish uchun massani hajmga bo‘lish.

2.4-jadval. Dataset

ID	Nomi	Shartnoma nomer	Taminotchi korxona	Narxi	Kelgan Mahsulot soni	Sotilgan Mahsulot soni	Kelgan sana	Muddati oy
0	Pepsi	3434	Pepsi MCHJ	15000	240	85	2024, 5, 19	12
1	Fanta	5322	CocaCola MCHJ	14000	180	50	2024, 5, 18	12
2	Sprite	2434	Pepsi MCHJ	16000	120	40	2024, 5, 18	12
3	CocaCola	2133	CocaCola MCHJ	15000	240	105	2024, 5, 17	12
4	Lipton	6432	CocaCola MCHJ	10000	60	15	2024, 5, 15	12
5	FuseTea	3210	Pepsi MCHJ	9000	60	20	2024, 5, 10	12

Yuqoridagi dataset yordamida yangi xususiyatlarni yaratish jarayonini ko‘rib chiqamiz. Bunda bizga kerakli xususiyatlarni jadvalga qo‘shimcha ustun shaklida qo‘shib olamiz.

```
current_date = datetime.now()
df['Kelgan_sana'] = pd.to_datetime(df['Kelgan_sana'], format='%Y, %m, %d')
df['Qolgan_Mahsulotlar_soni'] = df['Kelgan_Mahsulot_soni'] - df['Sotilgan_Mahsulot_soni']
df['Sotilgan_mahsulotlar_narxi'] = df['Sotilgan_Mahsulot_soni'] * df['Narxi']
df['Mahsulot_kelganiga_qancha_vaqt'] = (current_date - df['Kelgan_sana']).dt.days
print(df)
def save_to_excel(dataframe, file_name):
    dataframe.to_excel(file_name, index=False)
    print(f"Data saved to {file_name}")

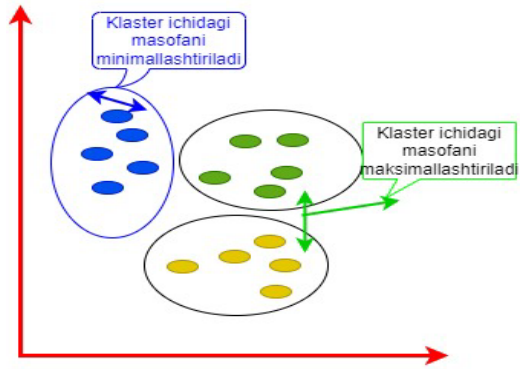
# DataSet-ni Excel fayliga saqlang
save_to_excel(df, 'guruhlangan_data3.xlsx')
```

2.5-jadval. Yangi xususiyat hosil qilingan dataset

ID	Nomi	Shartnoma nomer	Taminotchi korxona	Narxi	Kelgan mahsulot soni	Sotilgan mahsulot soni	Muddati oy	Qolgan mahsulotlar soni	Sotilgan mahsulotlar narxi	Mahsulot kelganiga qancha vaqt
0	Pepsi	3434	Pepsi MCHJ	15000	240	85	12	155	1275000	0
1	Fanta	5322	CocaCola MCHJ	14000	180	50	12	130	700000	1
2	Sprite	2434	Pepsi MCHJ	16000	120	40	12	80	640000	1
3	CocaCola	2133	CocaCola MCHJ	15000	240	105	12	135	1575000	2
4	Lipton	6432	CocaCola MCHJ	10000	60	15	12	45	150000	4
5	FuseTea	3210	Pepsi MCHJ	9000	60	20	12	40	180000	9

Yuqoridagi jadvalda bizga kerakli uchta yangi xususiyatni hosil qilib oldik.

Klasterlash. Guruhdagi obyektlar bir-biriga o‘xshash (bog‘langan) va boshqa guruhlardagi obyektlardan farqli(bog‘liq bo‘lmagan) ekanligini bildiruvchi kattalik yoki alohida belgilar orqali obyektlar guruhlarini topish. Klasterlash usuli yordamida bir nechta sinflarga ajratib olish mumkin. Misol: aralash turdagi mevalarni(olma, behi va o‘rik) alohida qutilarga joylashtirish kerak bo‘lsa, bunda har bir meva turini sinflarga ajratib ularni alohida qutilarga joylashtirish imkoniyatini beradi.



2.2.6-rasm. Klasterlash

Ma’lumotlarni tozalash (Data cleaning)

Ma’lumotlarni tozalash yoki ma’lumotlarni oldindan qayta ishlash (data cleaning) - bu ma’lumotlar to‘plamini tahlil va modellash uchun toza va ishonchli holatga keltirish jarayonidir. Ushbu jarayon

ma’lumotlarda mavjud bo‘lishi mumkin bo‘lgan xatolar, bo‘sh qiymatlar va boshqa muammolarni bartaraf etishga qaratilgan.

Eslatma: u raqamli qiymatlarning asl taqsimotiga ta’sir qilishi mumkin.

Yo‘qolgan qiymatlarni boshqarish. Qiymatlarni almashtirish.

– O‘rtacha/median rejimdan foydalanish.

– Mavjud qiymatlarning ehtimollik taqsimotidan foydalanib, yetishmayotgan qiymatlarni hisoblash.

–Ma’lumotlarni segmentatsiyalash va har bir segmentning o‘rtacha/median rejimidan foydalanish.

– Ma’lumotlarni segmentatsiyalash va segment ichida ehtimollik taqsimotidan foydalanish.

–Yetishmayotgan qiymatlarni hisoblash uchun tasniflash/regressiya modelini yaratish.

Misol: ma’lumotlar to‘plamidagi, yo‘q ma’lumotlarni hosil qilishni ko‘rib chiqamiz. Bunda birinchi navbatda dataset hosil qilib olamiz.

```
import pandas as pd
import numpy as np
data = {
   'Gruh_ID': [101, 102, np.nan, 103],
   'Ism': ['Jasur', 'Fayzullo‘, 'Sunnatillo‘,    'Xurshid'],
```

```
    'Fan': ['Matematika', 'Ingliz_tili', 'Fizika',   'Rus_tili',],
    'Modeli': [np.nan, 95, 80, 88],
    'Reytingi': [2, 1, 4, 3],
    'Baho‘: ['A', 'C', 'B', 'D'] }
df = pd.DataFrame(data)
print("Ma’lumotlar")
print(df)
```

```
Ma'lumotlar
   Gruh_ID         Ism          Fan  Modeli  Reytingi Baho
0    101.0       Jasur   Matematika     NaN         2    A
1    102.0    Fayzullo  Ingliz_tili    95.0         1    C
2      NaN  Sunnatillo       Fizika    80.0         4    B
3    103.0     Xurshid     Rus_tili    88.0         3    D
```

Ma’lumotlarni tozalash yordamida kerakli dataset hosil qilib olinadi.

```
df_cleaned = df.dropna()
print("\n Qiymatlari yo‘q bo‘lgan qatorlarni olib tashlaganingizdan so‘ng DataFrame:")
print(df_cleaned)
```

Qiymatlari yo‘q bo‘lgan qatorlarni olib tashlagandan keyingi dataset

	Gruh_ID	Ism	Fan	Modeli	Reytingi	Baho
1	102.0	Fayzullo	Ingliz_tili	95.0	1	C
3	103.0	Xurshid	Rus_tili	88.0	3	D

Ma’lumotlarni to‘ldirishda - hosil qilingan datasetdagi yetishmayotgan qiymatlarni o‘rtacha qiymat bilan to‘ldirib amalga oshiriladi. Bunda datasetdagi barcha qiymatlar yig‘indisi hisoblanib, soniga bo‘lib o‘rtacha qiymat hosil qilib olinadi.

```
mean_imputation = df['Gruh_ID'].fillna(df['Gruh_ID'].mean())
mean_imputation2 = df['Modeli'].fillna(df['Modeli'].mean())
print("\nO‘rtacha yordamida Gruh_Id hisoblash:")
print(mean_imputation)
print("\nO‘rtacha yordamida Modeli hisoblash:")
print(mean_imputation2
```

O‘rtacha qiymat topish yordamida Gruh_Id sini hisoblash:

0 **101.0**

1 102.0

2 102.0

3 103.0

O‘rtacha qiymat yordamida Modeli hisoblash:

0 **87.666667**

1 95.000000

2 80.000000

3 88.000000

Keyingi holatda datasetdagi yetishmayotgan qiymatlarni median usuli yordamida hisoblashlarni amalga oshiramiz . Bunda dataset soniga nisbatan qaraladi. Agar dataset soni toq bo‘lsa [np.nan, 95, 80, 88], o‘sish tartibida joylashtirib olamiz [np.nan, 95, 88, 80] va o‘rtacha qiymatini topib np.nan o‘rniga to‘ldiramiz [88, 95, 80, 88].

```
median_imputation =
df['Gruh_ID'].fillna(df['Gruh_ID'].median())
median_imputation2 = df['Modeli'].fillna(df['Modeli'].median())
print("\nMedian yordamida Gruh_id hisoblash:")
print(median_imputation)
print("\nMedian yordamida Modeli hisoblash:")
print(median_imputation2)
Median yordamida Gruh_id hisoblash:
0   101.0
1   102.0
2   102.0
3   103.0
Median yordamida Modeli hisoblash:
0   88.0
1   95.0
2   80.0
3   88.0
```

Keltirilgan misollar yordamida datasetdagi nol ya’ni bo‘sh ma’lumot joylarini o‘rtacha va median qiymatlarni hisoblash natijasida to‘ldirib olish jarayonini ko‘rib chiqdik. Qaysi hisoblash usulidan foydalanish obyektdagi muammodan kelib chiqib tanlanadi.

Qiymatlarni diskretlashtirish. Diskretlashtirish - uzluksiz atributni tartibli atributga aylantirish jarayoni hisoblanadi va quyidagi hollarda qo‘llaniladi.

Potensial cheksiz sonli qiymatlar oz sonli toifalarga ajratiladi.

Tasniflashda diskretizatsiya qo‘llaniladi.

Ko‘pgina tasniflash algoritmlari, agar mustaqil va bog‘liq o‘zgaruvchilar bir nechta qiymatlarga ega bo‘lsa, diskretlash samarali hisoblanadi.

Miqdoriy atributlarni diskretlash. Har bir qiymat o‘zi tegishli bo‘lgan interval bilan almashtiriladi. Quyida miqdoriy atributlarni diskretlashni amalga oshiramiz.

```
data = {
   'yoshi': [12, 50, 20, 36, 14, 21, 45, 60],
   'vazni': [38, 62, 60, 85, 40, 56, 75, 80],
   'daromadi': [50000, 15000000, 9000000, 28000000, 500000,
   18000000, 24000000, 31000000]    }
```

```
import pandas as pd
import numpy as np
# Uzluksiz ma’lumotlar bilan DataSet yaratish
data = {
   'yoshi': [12, 50, 20, 36, 14, 21, 45, 60],
   'vazni': [38, 62, 60, 85, 40, 56, 75, 80],
   'daromadi': [50000, 15000000, 9000000, 28000000, 500000,
                18000000, 24000000, 31000000] }
df = pd.DataFrame(data)
# Balandlik uchun sinf va teglarni belgilash
yosh_bins = [0, 16, 36, 70, np.inf]
yosh_labels = ['0-16', '17-36', '37-70', '>70']
```

```
df['Yosh Guruhlari'] = pd.cut(df['yoshi'], bins=yosh_bins,
labels=yosh_labels)

# Og‘irlik uchun sinflar va teglarni belgilash

vazn_bins = [0, 40, 60, 80, np.inf]

vazn_labels = ['0-40kg‘, '41-60kg‘, '61-80kg‘, '>80kg‘]

df['Vazn Guruhlari'] = pd.cut(df['vazni'], bins=vazn_bins,
labels=vazn_labels)

# Daromad uchun sinflar va teglarni belgilash

daromad_bins = [0, 10000000, 20000000, 25000000, 30000000,
np.inf]

daromad_labels = ['0-10mln', '11-20mln', '21-25mln', '26-30mln',
'>30mln']

df['Daromad Guruhlari'] = pd.cut(df['daromadi'],
bins=daromad_bins, labels=daromad_labels)

print(df)

# DataSet-ni Excelga saqlash funksiyasi

def save_to_excel(dataframe, file_name):

    dataframe.to_excel(file_name, index=False)

    print(f"Data saved to {file_name}")

# DataSet-ni Excel fayliga saqlang

save_to_excel(df, 'guruhlangan_data.xlsx')
```

2.6-jadval. Datasetni diskretlash

yoshi	vazni	daromadi	Yosh Guruhlari	Vazn Guruhlari	Daromad Guruhlari
12	38	50000	0-16	0-40kg	0-10mln
50	62	15000000	37-70	61-80kg	11-20mln
20	60	9000000	17-36	41-60kg	0-10mln
36	85	28000000	17-36	>80kg	26-30mln
14	40	500000	0-16	0-40kg	0-10mln
21	56	18000000	17-36	41-60kg	11-20mln
45	75	24000000	37-70	61-80kg	21-25mln
60	80	31000000	37-70	61-80kg	>30mln

Dasturiy algoritm asosida gruhlangan diskret ma'lumot hosil bo'ladi.

Qiymatlarni tabiiy bog'lash (Data binning) - bu kichik kuzatuv xatolarining ta'sirini kamaytirish va katta ma'lumotlar to'plamlarini tahlil qilishni soddalashtirish uchun ishlatiladigan kerakli ma'lumotlarni oldindan qayta ishlash usuli. Boshqacha qilib aytganda qiymatlarni tartiblash, qiymatlar diapazonini bir xil o'lchamdagi k-sinflarga bo'lish. Ma'lumotlarning asl qiymatlari sinflar deb nomlanuvchi kichik intervallarga bo'linadi va keyin ular o'sha sinf uchun hisoblangan umumiy qiymat bilan almashtiriladi. Bu kiritilgan ma'lumotlarga ortiqcha ta'sir qiluvchi tebranishlarni kamaytiradi va kichik ma'lumotlar to'plamining moslashish ehtimolini oldini oladi.

Ma'lumotlarni sinflarga bo'lishning 2 usuli mavjud:

1. **Teng chastotali bog'lash** (Binning): ajratilgan sinflar teng chastotaga ega. Misol uchun bizga quyidagicha ma'lumot berilgan bo'lsin. Bunda k-sinflar sonini ixtiyoriy tanlaymiz: k = 4

X = [1, 8, 23, 40, 61, 66, 75, 90, 100, 123, 156, 190, 220, 250, 276, 290]

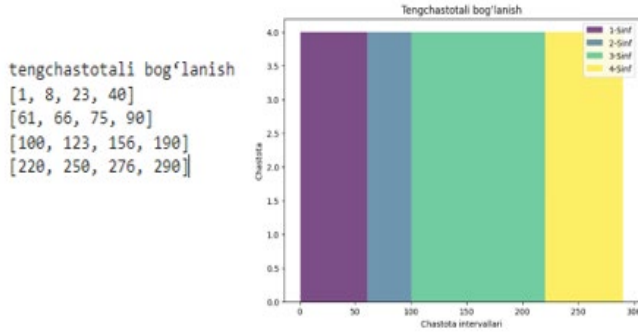
$$\sigma = \frac{N}{k}$$

N – kiruvchi malumot uzunligi, k – sinflar soni, σ – xosil qilingan sinf chastotasi.

```
# Teng chastotali bog'lash
def tengchastota(arr1, k):
    N = len(arr1)
```

```
    sigma = int(N/k)
    for i in range(0, k):
        arr = []
        for j in range(i * sigma, (i + 1) * sigma):
            if j >= N:
                break
            arr = arr + [arr1[j]]
        print(arr)
# data to be binned
data = [1,8,23,40,61,66,75,90,100,123,156,190,220,250,276,290]
# sinf
k = 4
print("tengchastotali bog‘lanish")
tengchastota(data, k)
```

Yuqoridagi dastur orqali olingan natija quyidagicha:



2.1.7-rasm. Teng chastotali bog‘lash

2. Teng masshtabda joylashtirish. Bunda sinflarning masshtabi bir-biriga teng bo‘ladi.

$$x_j \in [x_{min} + i\delta,\ x_{min} + (i+1)\delta)$$

δ – sinf masshtabi, x_j - sinf elementi.

$$\delta = \frac{x_{\max} - x_{\min}}{k}$$

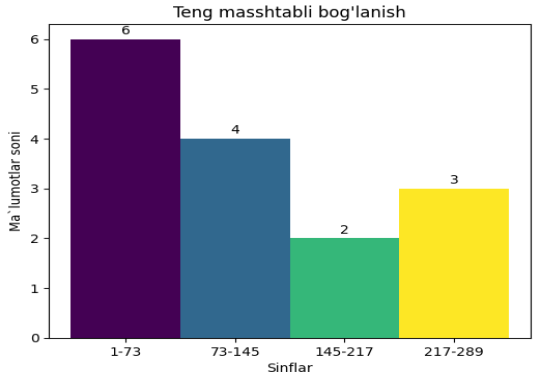
```
# tengmasshtab
def tengmasshtab(arr1, k):
    a = len(arr1)
    delta = int((max(arr1) - min(arr1)) / k)
    min1 = min(arr1)
    arr = []
    for i in range(0, k + 1):
        arr = arr + [min1 + delta * i]
    arri=[]

    for i in range(0, k):
        temp = []
        for j in arr1:
            if j >= arr[i] and j <= arr[i+1]:
                temp += [j]
        arri += [temp]
    print(arri)
# kiruvchi ma'lumot
data = [1,8,23,40,61,66,75,90,100,123,156,190,220,250,276,290]
# sinf
k = 4
```

```
print("teng masshtabli bog‘lanish")
tengmasshtab(data, k)
```

Yuqoridagi dastur orqali olingan natija quyidagicha:

```
teng masshtabli bog'lanish
[[1, 8, 23, 40, 61, 66], [75, 90, 100, 123], [156, 190], [220, 250, 276]]
```



2.1.8-rasm. Teng masshtabli bog‘lash

Ma’lumotlarni siqish (data compression)

Ma’lumotlarni siqish (data compression) – ma’lumot hajmini kamaytirish uchun ishlatiladigan jarayon bo‘lib, bu jarayon ma’lumotlarni samarali saqlash, uzatish va qayta ishlashga yordam beradi. Ma’lumotlarni siqish ko‘pincha disk maydoni yoki tarmoqli kenglikni tejash, uzatish va saqlash xarajatlarini kamaytirish uchun qo‘llaniladi. Ma’lumotlarni siqish ikki asosiy turga bo‘linadi: yo‘qotishli (lossy) va yo‘qotishsiz (lossless) siqish.

Yo‘qotishli siqish (Lossy Compression):

Yo‘qotishli siqish usullari ma’lumotlar hajmini katta miqdorda kamaytirishga imkon beradi, lekin ma’lumotlarning ba’zi qismini yo‘qotadi. Bu turdagi siqish, masalan, rasmlar (JPG), videolar va audio (MP3) fayllar kabi multimediya ma’lumotlarida keng qo‘llaniladi.

- Bu usullar ba’zi ma’lumotlarni yo‘qotadi, lekin ko‘pincha inson ko‘zi va quloqlari uchun bu yo‘qotish sezilmaydi.

Yo‘qotishsiz siqish (Lossless Compression):

Yo‘qotishsiz siqish usullari ma’lumotlarni hajmini kamaytiradi, lekin asl ma’lumotlarni butunlay qayta tiklash imkonini beradi. Bu turdagi siqish matnli ma’lumotlar va boshqa ma’lumotlar uchun keng qo‘llaniladi.

Bugungi kunda sun’iy intellekt texnologiyalaridan samarali foydalanish uchun ma’lumotlarni siqish yoki kichik hajmga keltirib olish muhim masalalardan biri hisoblanadi. Mashinali o‘qitishning bir nechta usullari, ayniqsa k-means clustering algoritmlari xususiyatini saqlab qolgan holda ma’lumotlarni siqishda keng foydalanilmoqda. Neyron tarmoqlarni o‘qitishda esa CNN-larning konvolyutsion qatlami ma’lumotlarni siqishni amalga oshiradi. Bundan tashqari tasvirlarni siqishda hozirda veyvlet usullari keng qo‘llanilmoqda.

Ma’lumotlarni almashinish jarayonlarida esa ma’lumotlarni siqishda quyidagilardan foydalaniladi:

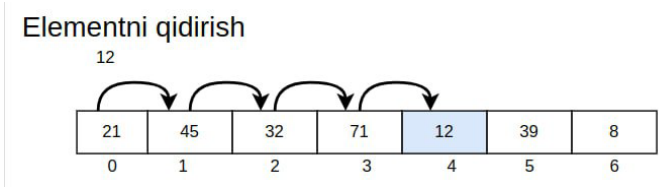
- ZIP va RAR: Foydalanuvchi fayllarini siqish uchun mashhur formatlar. Ushbu formatlar ma’lumotlarni yo‘qotishsiz siqadi.

- Huffman kodlash: Ushbu algoritm ma’lumotlar ichida takrorlanadigan elementlarni kodlash uchun foydalaniladi.

2.1.2.
Ma’lumotlarni qidirish algoritmlari (Data search)

Qidiruv algoritmlari sun’iy intellektning eng muhim sohalaridan biridir. Sun’iy intellektda qidiruv algoritmlari muammoni hal qilishning universal usullaridir. Ratsional agentlar yoki sun’iy intellektdagi muammolarni hal qiluvchi agentlar asosan ma’lum bir muammoni hal qilish va eng yaxshi natijani ta’minlash uchun ushbu qidiruv strategiyalari yoki algoritmlaridan foydalanadi. Qidiruv algoritmlari tushunarli bo‘lishi uchun birinchi navbatda chiziqli qidiruv algoritmini keltirib o‘tamiz.

Chiziqli qidiruv-berilgan ma’lumotlarning bir chetidan boshlanadigan va kerakli element topilgunga qadar ro‘yxatning har bir elementi bo‘ylab o‘tadigan ketma-ket qidiruv algoritmi sifatida aniqlanadi , aks holda qidiruv ma’lumotlar to‘plamining oxirigacha davom etadi. Har bir element kalit uchun potentsial moslik sifatida qaraladi va bir xilligi tekshiriladi. Agar kalitga teng element topilsa, qidiruv muvaffaqiyatli bo‘ladi va ushbu elementning indeksi qaytariladi. Agar kalitga teng bo‘lgan element topilmasa, qidiruv “Hech qanday moslik topilmadi” deb javob qaytaradi.

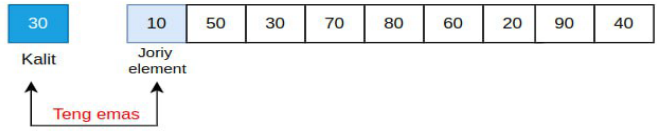


2.1.9-rasm. Chiziqli qidiruv algoritmi

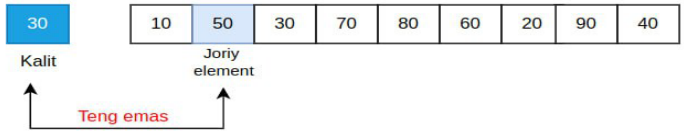
Masalan: arr [] = {10, 50, 30, 70, 80, 20, 90, 40} va kalit = 30 massivni ko'rib chiqamiz.

1-qadam: Birinchi elementdan boshlang (indeks 0) va massivni har bir element bilan solishtiramiz (arr[i]).

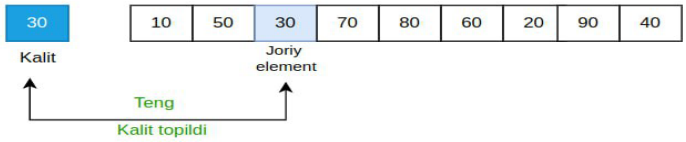
Massiv arr[0] birinchi elementi bilan solishtirish. Teng bo'lmagani uchun iterator potentsial moslik sifatida keyingi elementga o'tadi.



Massiv keyingi element bilan solishtirish arr[1]. Teng bo'lmagani uchun iterator potentsial moslik sifatida keyingi elementga o'tadi.



2-qadam: Endi arr [2] ni arr[1] bilan solishtirganda qiymat mos keladi.

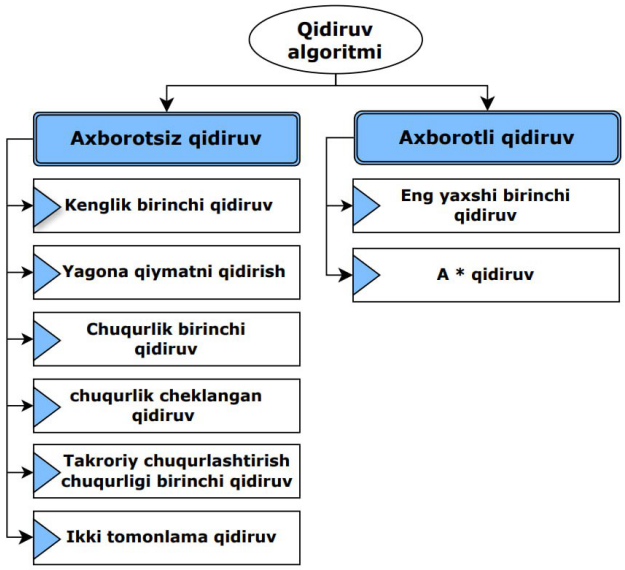


2.1.10-rasm. Chiziqli qidiruv algoritmi asosida kerakli qiymatni topish

Shunday qilib, chiziqli qidiruv algoritmi massivning qiymatini to‘g‘ri topgani aniq bo‘ladi va massiv topilganda element indeksini saqlab qoladi.

Axborotsiz va axborotli qidiruv algoritmlari

Ma’lumotlarning xususiyatlari va muammolari asosida qidiruv algoritmlarining axborotsiz va axborotli turlari mavjud.



2.1.11-rasm. Axborotsiz va axborotli qidiruv algoritmlari strukturasi

Axborotsiz qidiruv – o‘zining maydoni haqida hech qanday ma’lumotga ega bo‘lmagan holda qidirishni amalga oshiradi, shuning uchun u ko‘r-ko‘rona qidiruv deb ham ataladi. U maqsad tuguniga erishgunga qadar daraxtning har bir tugunini tekshiradi. Uni quyidagi asosiy turga bo‘lish mumkin:

Kenglik-birinchi qidiruv*(Breadth First Search)*

Chuqurlik birinchi qidiruv*(Depth first search)*

Chuqurlik bilan cheklangan qidiruv*(Depth-limited search)*

Takroriy chuqurlashtirish chuqurlik - birinchi qidiruv*(Iterative deepening depth- first Search)*

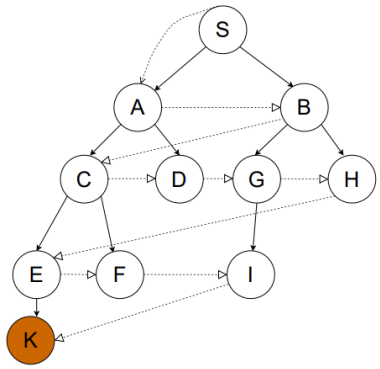
Yagona xatolik qidiruv*(Uniform Cost Search)*

Ikki tomonlama qidiruv *(Bidirectional Search)*

Kenglik-birinchi qidiruv (Breadth First Search yoki BFS) – bu qidiruv daraxt yoki grafikni kesib o‘tish uchun eng keng tarqalgan qidiruv strategiyasi. Ushbu algoritm daraxt yoki grafikda kenglik bo‘yicha qidiradi, shuning uchun u kenglik bo‘yicha birinchi qidiruv deb ataladi.

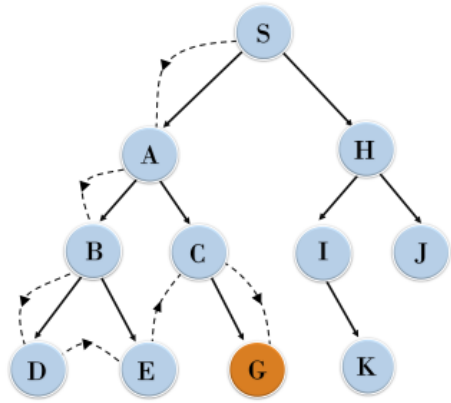
BFS algoritmi daraxtning ildiz tugunidan qidirishni boshlaydi va keyingi darajadagi tugunlarga o‘tishdan oldin joriy darajadagi barcha voris tugunni kengaytiradi. Agar yechim ildiz tugunidan uzoqda bo‘lsa kenglik birinchi qidiruv ko‘p vaqt talab qiladi. Quyidagi daraxt tuzilmasida biz daraxtning BFS algoritmi yordamida S ildiz tugunidan maqsad tuguniga K o‘tishini ko‘rib o‘tamiz. BFS qidiruv algoritmi

qatlamlar bo‘ylab o‘tadi, shuning uchun u nuqtali strelka bilan ko‘rsatilgan yo‘ldan boradi va bosib o‘tgan yo‘l quyidagicha bo‘ladi:



2.1.12-rasm. Kenglik-birinchi qidiruv strukturasi

Chuqurlikdagi birinchi qidiruv *(Depth first search yoki DFS)* - bu daraxt yoki grafik ma’lumotlar strukturasini kesib o‘tish uchun rekursiv algoritm. U chuqurlikdagi birinchi qidiruv deb ataladi, chunki u ildiz tugunidan boshlanadi va keyingi yo‘lga o‘tishdan oldin har bir yo‘lni eng katta chuqurlik tuguniga qidirib boradi. DFS uni amalga oshirish uchun stek ma’lumotlar strukturasidan foydalanadi. DFS algoritmining jarayoni BFS algoritmiga o‘xshaydi.

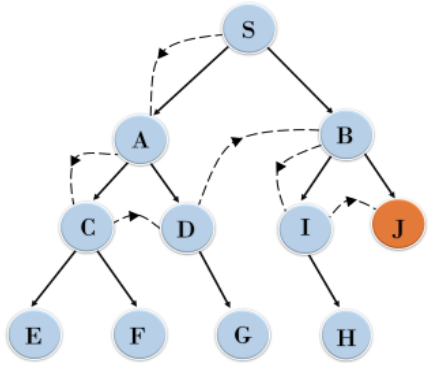


2.1.13-rasm. Chuqurlikdagi birinchi qidiruv

Chuqurlik bilan cheklangan qidiruv*(Depth-limited search)* - algoritmi oldindan belgilangan chegara bilan chuqurlikdagi birinchi qidiruvga o'xshaydi. Chuqurlik bilan cheklangan qidiruv, birinchi uchragan chuqurlikdagi yo'lning kamchiliklarini hal qilishi mumkin. Ushbu algoritmda chuqurlik chegarasidagi tugunga ishlov beriladi, chunki uning keyingi tugunlari yo'q. Chuqurlik bilan cheklangan qidiruv ikkita muvaffaqiyatsizlik holati bilan tugatilishi mumkin:

Standart nosozlik qiymati: bu muammoning yechimi yo'qligini ko'rsatadi.

Chiqib ketish nosozlik qiymati: u berilgan chuqurlik chegarasida muammoning yechimini aniqlamaydi.



2.1.14-rasm. Chuqurlik bilan cheklangan qidiruv

Agar yechim chuqurlik chegarasidan yuqori bo‘lsa, DLS qidiruv algoritmi to‘liq hisoblanadi.

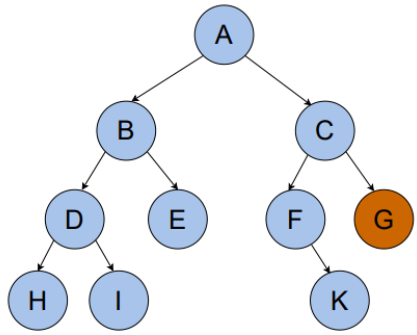
DLS algoritmining vaqt murakkabligi $O(b^{\ell})$.

DLS algoritmining fazoviy murakkabligi $O(b \times \ell)$ dir .

Chuqurlik bilan cheklangan qidiruvni DFSning alohida holati sifatida ko‘rish mumkin va u $\ell > d$ bo‘lsa ham optimal emas.

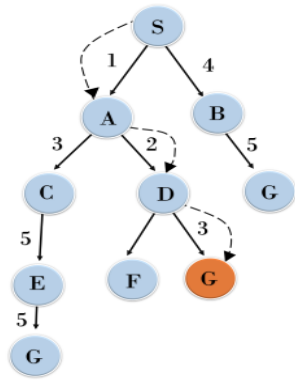
Takroriy chuqurlashtirish chuqurlik-birinchi qidiruv.*(Iterative deepening depth-first Search)* Iterativ chuqurlashtirish algoritmi DFS va BFS algoritmlarining birikmasidir. Ushbu qidiruv algoritmi eng yaxshi chuqurlik chegarasini topadi va maqsadga erishmaguncha ittaratsiyani bosqichma-bosqich oshirish orqali amalga oshiradi. Ushbu algoritm ma’lum bir “Chuqurlik chegarasi” gacha chuqurlikdan birinchi bo‘lib qidirishni amalga oshiradi va maqsad tugun topilmaguncha, har bir iteratsiyadan keyin chuqurlik chegarasini oshirishda davom etadi. Ushbu qidiruv algoritmi kenglik-birinchi

qidiruvning tezkor qidiruvi va chuqurlik-birinchi qidiruvning xotira samaradorligi afzalliklarini birlashtiradi. Takroriy qidiruv algoritmi qidiruv maydoni katta bo‘lsa va maqsad tugunining chuqurligi noma'lum bo‘lsa, axborotsiz qidiruv uchun foydalidir.



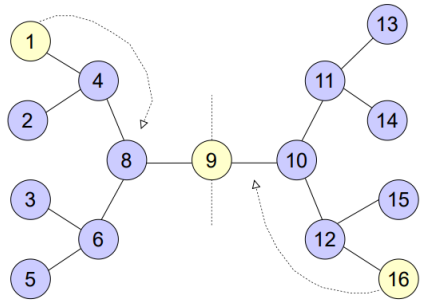
2.1.15-rasm. Chuqurlik bo‘yicha qidiruv

Yagona xatolikni qidirish (*Uniform Cost Search yoki UCS***)** - bu vaznli daraxt yoki grafikni kesib o‘tish uchun ishlatiladigan qidiruv algoritmi. Ushbu algoritm har bir chekka qidiruv uchun har xil xatolik mavjud bo‘lganda kuchga kiradi. Yagona xatolikni qidirishning asosiy maqsadi eng kam umumiy xatolikka ega bo‘lgan maqsad tuguniga yo‘l topishdir. Yagona xatolik qidiruvi tugunlarini ildiz tugunini tashkil etadigan yo‘l xatoliklariga qarab kengaytiradi. U optimal xatolik talab qilinadigan har qanday graf ma'lumotlar tuzilmasini hal qilish uchun ishlatilishi mumkin. Bu qidiruvda qancha qadamlar borligiga ahamiyat bermaydi va faqat xatolikni kamaytirishga intiladi. Shu sababli, bu algoritm cheksiz siklga aylanib qolishi mumkin.



2.1.16-rasm. Yagona xatolikni qidirish strukturasi

Ikki tomonlama qidiruv *(Bidirectional Search).* Bir vaqtning o‘zida ikkita qidiruvni amalga oshiradi, birinchisi shaklning boshlang‘ich holati oldinga qidiruv deb ataladi, ikkinchisi esa maqsad tugunini topish uchun orqaga qidiruv deb ataladi. Ikki tomonlama qidiruv bitta qidiruv grafigini ikkita kichik pastki grafik bilan almashtiradi, ulardan biri qidiruvni boshlang‘ich cho‘qqidan, ikkinchisi esa maqsad cho‘qqisidan boshlanadi. Ushbu ikki grafik bir-birini kesib o‘tganda qidiruv to‘xtaydi. Ikki tomonlama qidiruv BFS, DFS, DLS va boshqalar kabi qidiruv usullaridan foydalanishi mumkin.



2.1.17-rasm. Ikki yo‘nalishli qidiruv algoritmialmas

Quyidagi qidiruv daraxtida ikki tomonlama qidiruv algoritmi qo‘llaniladi. Bu algoritm bitta graf ma’lumotlar tuzilmasini, ikkita kichik grafga ajratadi. U 1-tugundan oldinga yo‘nalishda o‘tishni boshlaydi va orqaga yo‘nalishda 16-tugundan boshlanadi. Algoritm ikkita qidiruv uchrashadigan 9-tugunda tugaydi.

Axborotlashtirilgan qidiruv algoritmlari

Axborotli qidiruv algoritmi katta qidiruv maydoni uchun foydaliroqdir. Axborotlashtirilgan qidiruv algoritmi evristik g‘oyadan foydalanadi, shuning uchun uni evristik qidiruv deb ham ataladi.

Evristik qidiruv - bu axborotlashtirilgan qidiruvda qo‘llaniladigan algoritm bo‘lib, u eng yaqin qidiruv yo‘lni topadi. U agentning joriy holatini kirish sifatida qabul qiladi va agentning maqsadga qanchalik yaqin ekanligi haqidagi taxminni ishlab chiqadi. Biroq, evristik usul har doim ham eng yaxshi yechimni bermasligi mumkin, ammo u kam vaqt ichida yaxshi yechim topishni amalga oshiradi. Evristik qidiruvda, evristik funksiya holatning maqsadga qanchalik yaqinligini baholaydi. Evristik funksiyaning qiymati har doim ijobiy bo‘ladi.

Evristik funksiya quyidagicha ifodalanadi.

$$h(n) <= h^*(n)$$

Bu yerda h(n) evristik qidiruv qiymati, h*(n) esa taxminiy qiymat. Demak, evristik qiymat taxminiy qiymatidan kichik yoki unga teng bo‘lishi kerak.

Sof evristik qidiruv - evristik qidiruv algoritmlarining eng oddiy shaklidir. U h(n) evristik qiymati asosida tugunlarni kengaytiradi. U ikkita ro‘yxatni o‘z ichiga oladi. OCHIQ va YOPIQ ro‘yxat. OCHIQ ro‘yxatida kengaytirilgan tugunlarni va YOPIQ ro‘yxatida hali kengaytirilmagan tugunlarni joylashtiradi. Har bir iteratsiyada eng past evristik qiymatga ega har bir n tugun kengaytiriladi va uning barcha vorislarini hosil qiladi va n yopiq ro‘yxatga joylashtiriladi. Axborotli qidiruvda biz quyida keltirilgan ikkita asosiy algoritmni o‘rganib chiqamiz:

Eng yaxshi birinchi qidiruv algoritmi

A* Qidiruv algoritmi

Eng yaxshi birinchi qidiruv algoritmi:

1-qadam. Boshlang‘ich tugun OCHIQ ro‘yxatga joylashtiriladi.

2-qadam. Agar OCHIQ ro‘yxati bo‘sh bo‘lsa to‘xtatadi va ortga qaytaradi.

3-qadam. h(n) ning eng past qiymatiga ega bo‘lgan OCHIQ ro‘yxatidan n tugunini olib tashlanadi va uni YOPIQ ro‘yxatiga joylashtiriladi.

4-qadam. n tugunini kengaytiradi va n tugunining vorislarini yaratadi.

5-qadam. n tugunining har bir davomchisini tekshiradi va har qanday tugun maqsad tugunmi yoki yo‘qligini aniqlaydi. Agar biron bir

voris tugun maqsad tugun bo‘lsa natijaga erishadi va qidiruvni to‘xtatadi, aks holda 6-bosqichga o‘tadi.

6-qadam. Har bir voris tugun uchun algoritm f(n) baholash funksiyasini tekshiradi, so‘ngra tugun OCHIQ yoki YOPIQ ro‘yxatda bo‘lganligini tekshiradi. Agar tugun ikkala ro‘yxatda bo‘lmagan bo‘lsa, uni OCHIQ ro‘yxatiga qo‘shadi.

2) A* Qidiruv algoritmi. A* qidiruvi eng yaxshi birinchi qidiruvning keng tarqalgan ko‘rinishidir. U h(n) evristik funksiyasidan va g(n) boshlang‘ich holatidan n tuguniga yetib borish qiymatidan foydalanadi. U UCS va chuqurlik eng yaxshi birinchi qidiruv xususiyatlarini birlashtirgan bo‘lib, ular yordamida muammoni samarali hal qiladi. A* qidiruv algoritmi evristik funksiyadan foydalangan holda qidiruv maydoni bo‘ylab eng qisqa yo‘lni topadi. Ushbu qidiruv algoritmi kamroq qidiruv daraxtini kengaytiradi va optimal natijani tezroq beradi. A* algoritmi UCS ga o‘xshaydi, faqat u g(n) o‘rniga g(n)+h(n) dan foydalanadi.

A* qidiruv algoritmida biz qidiruv evristikasidan hamda tugunga yetib borish algoritmidan foydalanamiz. Shunday qilib, biz ikkala xatolikni quyidagi tarzda birlashtira olamiz va bu summa fitnes raqami deb ataladi.

A* qidiruv algoritmi:

OCHIQ ro‘yxatiga boshlang‘ich tugunni joylashtiradi.

OCHIQ ro‘yxati bo‘sh yoki yo‘qligini tekshiradi, agar ro‘yxat bo‘sh bo‘lsa, natijaga erishmaydi va to‘xtaydi.

OCHIQ ro‘yxatidan baholash funksiyasining eng kichik qiymatiga (g+h) ega bo‘lgan tugun tanlanadi, agar n tugun maqsad tugun bo‘lsa, natijaga erishadi va itteratsiya to‘xtaydi, aks holda N tuguni kengaytirilib va uning barcha vorislari yaratiladi hamda n yopiq ro‘yxatga kiritiladi. Har bir n vorisi uchun, n ning OCHIQ yoki YOPIQ ro‘yxatida ekanligi tekshiriladi, agar bo‘lmasa, n uchun baholash funksiyasi hisoblanadi va OCHIQ ro‘yxatga joylashtiriladi. Aks holda, agar n tugun OCHIQ va YOPIQ bo‘lsa, u eng past g(n) qiymatini aks ettiruvchi ko‘rsatkichga biriktirishni talab etadi.

Nazorat savollari

1. Axborotsiz qidiruv qanday algoritm?
2. Axborotli qidiruv qanday algoritm?
3. Sof evristik qidiruv qanday algoritm?

2.1-bo‘lim bo‘yicha amaliy qism

2.1-bo‘lim bo‘yicha amaliy qism quyidagi mavzular bo‘yicha keltirilgan. Bunda keltirilgan havola yoki QR kodni skaner qilish orqali GitHubga murojat qilib dasturiy koddan foydalanish imkoniyatiga ega bo‘lasiz.

https://github.com/TUITAI/

2.3. BILIMLAR BAZASI

Bilim - axborot yoki ma’lumotlar obyektlari o‘rtasidagi aniq funksional aloqadir. Ma’lumotlar, axborotlar va bilimlar bir-biriga bog‘liq tushunchalar bo‘lib, faktlar va kuzatishlarni tushunish va izohlash jarayonining turli bosqichlarini ifodalaydi. Ayniqsa ular bilimlar bazasini shakillantirishda bir-biriga uzviy bog‘liq holda qo‘llaniladi.

Axborot - bu ma’no va kontekstni beradigan tarzda tashkil etilgan, tuzilgan yoki qayta ishlangan ma’lumotlar. Axborot - bu aniq savollarga javob berish yoki tushuncha berish uchun talqin qilingan ma’lumotlar.

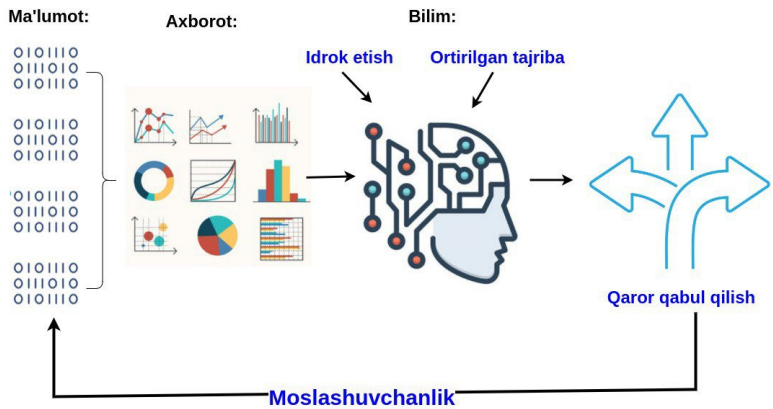
Misol uchun, kunlik haroratlar ro‘yxatidan hisoblangan o‘rtacha harorat axborotdir. Xuddi shunday, sotib olish xatti-harakatlaridagi xususiyatlarni ochib beradigan mijozlar xaridlarining qisqacha mazmuni axborotdir.

Ma’lumotlar - bu xom, qayta ishlanmagan faktlar yoki kuzatishlarni anglatadi. Bu axborot va bilimlar quriladigan poydevordir. Ma’lumotlar sifatli yoki miqdoriy bo‘lishi mumkin va foydali bo‘lishi uchun tartibga solinishi va qayta ishlanishi kerak bo‘lgan raqamlar, xususiyatlar, belgilar, tasvirlar, tovushlar yoki boshqa kiritish shakllarini o‘z ichiga olishi mumkin.

Ma’lumotga misol: ma’lum vaqt davomida qayd etilgan haroratlar ro‘yxati, mijozlar xaridlari yozuvlari yoki kameradan olingan tasvirlar to‘plami.

Bilim - bu inson tushunchasi, talqini va tajribasini o‘zida mujassamlashtirgan holda axborotdan tashqariga chiqadi. Bilim ko‘pincha ma’lum sohalar, fanlar yoki tajribalar kontekstida sintez qilingan va tushunilgan ma’lumotlar to‘plami sifatida ko‘riladi. Bu ma’lum bir sohada muammolarni hal qilish, qarorlar qabul qilish va tushunchalarni shakllantirish uchun ma’lumot va axborotlarni samarali qo‘llash qobiliyatini o‘z ichiga oladi.

Masalan, meteorolog ob-havoni bashorat qilish va ogohlantirishlar berish uchun ob-havo sharoitlari va turli manbalardan olingan ma’lumotlardan foydalanadi.



2.3.1-rasm. Ma’lumot, Axborot, Bilim

Ma'lumot, axborot va bilimga quyidagicha misol keltiramiz.

5°C	*ma'lumot.*
Tashqari sovuq	*axborot.*
Tashqarida sovuq va sovuq bo'lsa, siz issiq palto kiyishingiz kerak	*bilim.*

Xulosa qilib aytganda, ma'lumotlar xomashyo, axborot ma'no va kontekstni ta'minlash uchun qayta ishlanadigan ma'lumotlar, bilim esa tushunish, o'rganish va qaror qabul qilish uchun ma'lumotlarni sintez qilish va qo'llashdir. Ma'lumotlardan bilimga o'tish ko'pincha ma'lumotlar va ma'lumotlarning turli qismlarini tajriba bilan izohlash, tahlil qilish va birlashtirishni o'z ichiga oladi.

Bilim uchta turga bo'linadi:

- *Deklarativ bilim:* bizga narsalar haqida faktlarni aytib beradi. Misol uchun, "Lampochka porlashi uchun elektr tokini talab qiladi" degan gap haqiqatan ham to'g'ri.

- *Protsessual bilim:* bilim olish uchun faktlardan foydalanishga asoslangan muqobil harakatlarni ta'minlaydi. Misol uchun, odam odatda choynakni yoqishdan oldin ichidagi suv miqdorini tekshiradi; agar choynakda suv yetarli bo'lmasa, unda ko'proq qo'shiladi.

- *Meta-bilim:* bilim haqidagi bilimdir. Bu bizga mutaxassislar qaror qabul qilish uchun bilimlardan qanday foydalanishini tushunishga yordam beradi. Masalan, samolyotlar va poezdlar haqidagi bilim uzoq

sayohatni rejalashtirishda foydali bo‘lishi mumkin, piyodalar va velosipedlar haqidagi bilim esa qisqa sayohatni rejalashtirishda foydali bo‘lishi mumkin.

Bilimga asoslangan tizimlar - bu ma’lum bilim sohalari bo‘yicha mutaxassislarning ishiga taqlid qilish uchun mo‘ljallangan kompyuter dasturlari.

Bilimlarga asoslangan tizimlar (BAT)ning oltita asosiy turi mavjud:

1. *Ekspert tizimlari.* Ekspert tizimlari miyaning yuqori darajadagi kognitiv funksiyalarini modellashtiradi. Ular inson mutaxassislarining qaror qabul qilish jarayoniga taqlid qilish uchun ishlatilishi mumkin. Ekspert tizimlari odatda insonning qaror qabul qilish jarayonini modellashtirish uchun ishlatiladi. Ekspert tizimlarida algoritmlar mavjud bo‘lsa-da, bu algoritmlarning ko‘pchiligi statik bo‘ladi, ya’ni ular vaqt o‘tishi bilan o‘zgarmaydi. Bu tizim qanday ishlashiga ishonch hosil qilsa-da, bu ekspert tizimi tajribadan o‘rganish uchun mo‘ljallanmaganligini bildiradi. Shuni ta’kidlash kerakki, ekspert tizimlari ko‘pincha BAT bilan sinonim sifatida tilga olinadi. Biroq, ekspert tizimlari shunchaki BAT toifasidir.

2. *Neyron tarmoqlar.* Neyron tarmoqlar esa miyani biologik darajada modellashtiradi. Miya ko‘rish va nutqni aniqlash kabi belgilarni aniqlab olish vazifalarini bajarishga imkon beradi.

3. *Genetik algoritmlar.* Genetik algoritm murakkab masalalarning yechimlarini ishlab chiqish usulidir. Misol uchun, bunday usul

millionlab mumkin bo‘lgan yechimlar orasidan imtihonlarni rejalashtirish (xonalar, talabalar, nazoratchilar va hatto jihozlar) muammosiga ko‘plab yaxshi yechimlardan birini topish uchun ishlatilishi mumkin.

“Genetik” atamasi algoritmlarning xatti-harakatlarini anglatadi. Bunday vaziyatda xatti-harakatlar evolyutsiyada ishtirok etadigan biologik jarayonlarga juda o‘xshaydi.

4. *Intellektual agentlar.* Aqlli agent, odatda uning maqsadi yoki umumiy vazifasi ko‘rsatilgan ba’zi qarorlarni qabul qiladigan dasturdir.

Aksariyat agentlar orqa fonda ishlaydi (ya’ni ularni foydalanuvchi ko‘rmaydi) va faqat o‘z topilmalari haqida xabar berish uchun paydo bo‘ladi. Agentlar ko‘pincha o‘rganish va o‘z foydalanuvchilari nomidan tobora murakkab qarorlar qabul qilish qobiliyatiga ega. Eng oddiy agentlar shunchaki ma’lumotni olishadi, eng murakkablari esa qaror qabul qilish uchun deduktiv fikrlashni o‘rganadilar va foydalanadilar.

5. *Data mining.* Data mining - bu ma’lumotlardagi ilgari noma’lum munosabatlarni aniqlash orqali bilimlarni kashf qilishni tavsiflash uchun ishlatiladigan atama. Data miningning klassik misollaridan biri supermarketlar ichidagi sotuvlarni tahlil qilish bilan bog‘liq.

6. *Intellektual o‘rgatuvchi tizimlari.* Kompyuterga asoslangan o‘quv muhitiga qiziqish arzon narxlarda yuqori sifatli ta’limga bo‘lgan talab bilan ortadi. Shu bilan birga, kompyuterlar arzonroq, kuchliroq va

foydalanuvchilarga qulayroq bo‘ladi. Ta’lim strategiyasini ma’lum bir talaba/foydalanuvchining o‘ziga xos o‘rganish uslubiga moslashtirish uchun murakkab uslubda javob beradigan muhit yaratish murakkab hisoblanadi. Repetitorlik tizimi aqlli bo‘lishi uchun u talabaning o‘rganishiga ko‘ra doimiy ravishda reaksiyaga kirishishi (o‘rgatishi) kerak.

2.3.1. Bilimlarni ajratib olish

Bilimlarni ajratib olish - bu tartiblangan (relyatsion ma’lumotlar bazalari, XML) va tartiblanmagan (matn, hujjatlar, tasvirlar) manbalardan bilim yaratish. Olingan bilimlar mashinada o‘qiladigan va mashina tomonidan talqin qilinadigan formatda bo‘lishi kerak va bilimlarni xulosa chiqarishni osonlashtiradigan tarzda ifodalashi kerak. U uslubiy jihatdan axborotni ajratib olish (NLP) va ETL (ma’lumotlar ombori) ga o‘xshash bo‘lsa-da, asosiy mezon - ekstraksiya natijasi tuzilgan ma’lumotni yaratish yoki relyatsion sxemaga aylantirishdan tashqariga chiqadi. Bu mavjud rasmiy bilimlardan qayta foydalanishni (identifikatorlar yoki ontologiyalarni qayta ishlatish) yoki manba ma’lumotlariga asoslangan sxemani yaratishni talab qiladi. Bilimlarni ajratib olishning yana bir mashhur misoli Vikipediyani tuzilgan ma’lumotlarga aylantirish, shuningdek, mavjud bilimlarni xaritalashdir. Bilimlarni ajratib olish va ma’lumotlarni amaliy bilimga aylantirish uchun turli usullardan foydalaniladi. Ushbu usullar mashinali o‘qitish, tabiiy tilni qayta ishlash va boshqa ma’lumotlarni tahlil qilish usullarini

o‘z ichiga olishi mumkin. Bilimlarni ajratib olishning ba’zi umumiy usullarini keltirib o‘tamiz:

Data mining usuli:

Data mining shakllar, munosabatlar yoki tushunchalarni aniqlash uchun katta ma’lumotlar to‘plamini o‘rganishni o‘z ichiga oladi.

Texnikalari - klasterlash, tasniflash, assotsiatsiya qoidalarini ajratib olish va anomaliyalarni aniqlash kiradi.

Misol: Talabalarning ikkita fanining umumiy o‘zlashtirishini baholash.

```
Talaba = [
    {"name": "Xurshid", "matematika": 92, "fizika": 94},
    {"name": "Fayzullo", "matematika": 85, "fizika": 75},
    {"name": "Jasur", "matematika": 50, "fizika": 70},
    {"name": "Sunnatillo", "matematika": 80, "fizika": 85},
    {"name": "Jahongir", "matematika": 75, "fizika": 65},]
def classify(matematika, fizika):
    if matematika >= 90 or fizika >= 90:
        return "5"
    elif matematika >= 70 and fizika >= 70:
        return "4"
    elif matematika >= 60 and fizika >= 60:
        return "3"
    else:
```

```
        return "Yiqildi"
    results = []
    for talaba in Talaba:
        name = talaba["name"]
        math = talaba["matematika"]
        fizika = talaba["fizika"]
        result = classify(math, fizika)
        talaba_result = {
            "name": name,
            "matematika": math,
            "fizika": fizika,
            "natija": result }
        results.append(talaba_result)
    for result in results:
        print(f"Talaba: {result['name']}, Matematika:
          {result['matematika']}, Fizika: {result['fizika']}, Natija:
          {result['natija']}")
```

2.7-jadval. Data mining usulida ajratib olingan bilimlar jadvali

Talaba	Matematika	Fizika	Natija
Xurshid	92	94	5
Fayzullo	85	75	4
Jasur	50	70	Yiqildi
Sunnatillo	80	85	4
Jahongir	75	65	3

Tabiiy tilni qayta ishlash (NLP) usuli:

NLP - bu AI sohasi bo‘lib, u kompyuterlarga inson tilini tushunish va qayta ishlash imkonini beradi.

Texnikalari - nomini tanib olish, hissiyotlarni tahlil qilish, mavzuni modellashtirish va matnni umumlashtirishni o‘z ichiga oladi.

Misol: Umumiy mavzular va his-tuyg‘ularni aniqlash uchun mijozlar sharhlarini tahlil qilish.

Mashinali o‘qitish usuli: mashinali o‘qitish algoritmlari bashorat qilish, ma’lumotlarni tasniflash yoki xususiyatlarni aniqlash uchun ma’lumotlarga o‘rgatilgan bo‘lishi mumkin.

Texnikalari – o‘qituvchili o‘qitish (tasniflash va regressiya), o‘qituvchisiz o‘qitish (klasterlash va o‘lchovlarni kamaytirish) va mustahkamlovchi o‘qitish kiradi.

Misol: Tarixiy ma’lumotlarga asoslanib, mijozlarning ishlamay qolishini bashorat qilish uchun mashinali o‘qitish modelidan foydalanish.

*Ontologiya muhandisligi usuli:*ontologiya muhandisligi ma’lum bir sohada bilimlarni ifodalash uchun tuzilgan tizimlarni yaratishni o‘z ichiga oladi.

Texnikalari - sohadagi bilimlarini modellashtiruvchi ontologiyalarni yaratish uchun tushunchalar, munosabatlar va ierarxiyalarni aniqlashni o‘z ichiga oladi.

Misol: tibbiy tashxis va davolash usullarini ifodalash uchun ontologiyani yaratish.

Bilim grafiklari usuli: bilim grafiklari - bu obyektlar va tushunchalar o‘rtasidagi munosabatlarni ifodalovchi ma’lumotlar tuzilmalari.

Texnikalari - ma’lumotlar manbalariga asoslangan grafiklarni qurish va tushunchalarni olish uchun grafik algoritmlaridan foydalanishni o‘z ichiga oladi.

Misol: Axborotni qidirishni osonlashtirish uchun kompaniyaning ichki bilimlar bazasi uchun bilimlar grafigini yaratish.

Qoidalarga asoslangan tizimlar usuli:

Qoidalarga asoslangan tizimlar qaror qabul qilish yoki ma’lumotlardan bilim olish uchun oldindan belgilangan qoidalardan foydalanadi.

Qoidalar odatda shartlar va harakatlarni ko‘rsatib, “agar-u holda” formatida ifodalanadi.

Misol: noodatiy tarmoq faolligi kabi ma’lumotlardagi muayyan shartlarga asoslangan ogohlantirishlarni ishga tushiradigan tizim.

*Matn tahlili:*matn tahlili matnli ma’lumotlardan ma’lumotlarni tahlil qilish va ajratib olishni o‘z ichiga oladi.

Texnikalari - kalit so‘zlarni ajratib olish, hujjatlarni tasniflash va nomli obyektni tanib olishni o‘z ichiga oladi.

Misol: huquqiy hujjatlar yoki yangiliklar maqolalaridan tegishli ma’lumotlarni olish.

Vizual tahlil: vizual tahlil foydalanuvchilarga murakkab ma'lumotlarni tushunishga yordam berish uchun ma'lumotlar vizualizatsiyasini tahliliy usullar bilan birlashtiradi.

Texnikalari - interaktiv vizualizatsiya, issiqlik xaritalari va asboblar paneli kiradi.

Misol: Savdo ma'lumotlaridagi tendentsiyalar va o'zgarishlarni aniqlash uchun vizualizatsiyadan foydalanish.

Ushbu usullar birlashtirilishi va muayyan sohalarga moslashtirilishi va ma'lumotlardan bilimlarni samarali ajratib olish uchun holatlardan foydalanishi mumkin. Usulni tanlash ma'lumotlar turiga, sohaga va bilim olish jarayonining maqsadlariga bog'liq.

2.3.2. Bilimlarni taqdim etish modellari

Bilimlarni taqdim etish modellari, bilimlarni aqlli tizimlar tomonidan tushunish, qayta ishlash va ulardan foydalanish uchun tizimlashtirish va ifodalash usullaridir. Turli xil modellar turli maqsadlarga xizmat qiladi va ularning har biri bilim va qo'llash turiga qarab o'zining kuchli va zaif tomonlariga ega. Bu yerda bilimlarni taqdim etishning to'rtta asosiy modeli keltirib o'tilgan:

freym model:

freymlar - stereotipli vaziyatlar, hodisalar yoki obyektlarni aks ettirish uchun ishlatiladigan ma'lumotlar tuzilmalari.

Freym haqida aniq ma'lumotni saqlaydigan yacheykalardan va shu slotlar uchun qiymatlarni ta'minlovchi to'ldirgichlardan iborat.

Freymlar meros mexanizmlarini o‘z ichiga olishi mumkin.

Misol: “Avtomobil” freymida “g‘ildiraklar”, “dvigatel turi” va “o‘tirish hajmi” uchun bo‘shliqlar bo‘lishi mumkin, to‘ldirgichlar har bir atribut uchun qiymatlarni belgilaydi.

Slot nomi	**Slot qiymati**	**Slot qiymatining turi**
Ism	Valiyev. V.V.	Belgilar qatori
Tug‘ulgan yili	01.01.1965	Sana
Mutaxassislik	Huquqshunos, advokat	Belgilar qatori
yoshi	59	Protsedura
Bo‘lim	Kadrlar bo‘limi	Belgilar qatori
Ish haqi	8000	Raqam
Manzil	Uy manzili	freym

2.3.1-rasm. Freym model

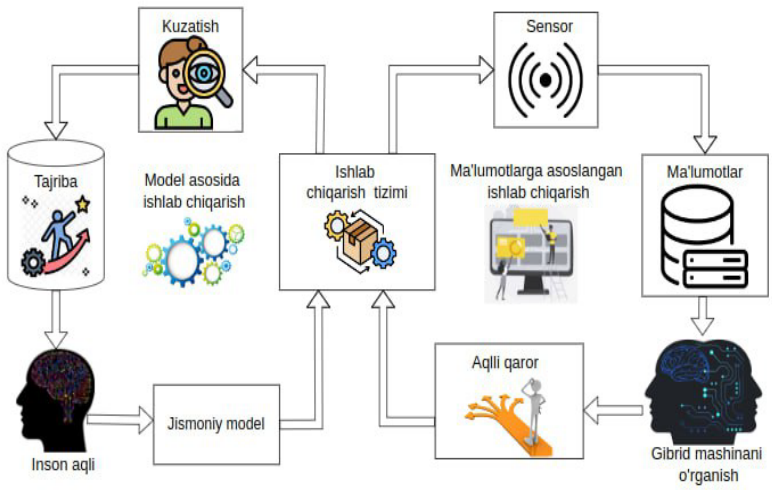
Produksion model:

Ishlab chiqarish tizimlari bilimlarni ifodalash uchun odatda “agar - keyin” iboralari ko‘rinishidagi qoidalar to‘plamidan foydalanadi.

"Agar" qismi bajarilishi kerak bo'lgan shartlarni, "keyin" qismi esa shartlar bajarilganda bajarilishi kerak bo'lgan harakatlarni belgilaydi.

Ishlab chiqarish tizimlari qaror qabul qilish va avtomatlashtirilgan fikrlash uchun foydalidir.

Misol: Tibbiy diagnostika bo'yicha ekspert tizimida "Agar bemorda yuqori isitma va toshmalar bo'lsa, qizamiq tashxisini qo'ying" degan qoida bo'lishi mumkin.



2.3.2-rasm. Produksion model.

Mantiqiy (logic) model: mantiqiy modellar bilimlarni ifodalash uchun rasmiy mantiqdan foydalanadi, masalan, taklif mantiqi yoki predikat mantiqi.

Ushbu modellar faktlar, munosabatlar va xulosalarni aniq ifodalash imkonini beradi.

Mantiqiy modellar ko‘pincha bilim bazalari va ma’lumotlar bazalarida qo‘llaniladi, bu ma’lumotlar bo‘yicha so‘rov va mulohaza yuritish imkonini beradi.

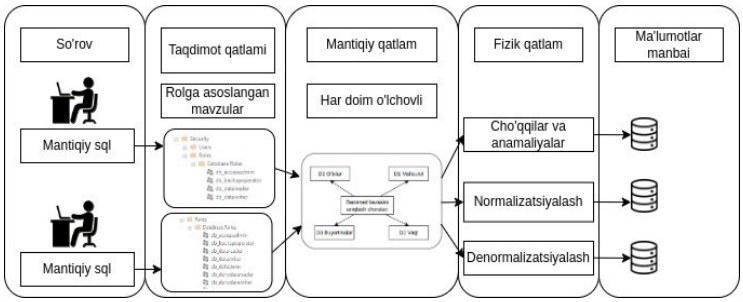
Misol: “Sokrat - inson” va “Barcha odamlar o‘lgan” kabi mavjudotlar o‘rtasidagi munosabatlarni ifodalash uchun predikat mantiqidan foydalanish “Sokrat - o‘limli” kabi mantiqiy xulosalar chiqarishga imkon beradi.

Semantik model. Semantik modellar sohadagi tushunchalar o‘rtasidagi ma’no va munosabatlarni ifodalashga qaratilgan.

Bu semantik tarmoqlar, ontologiyalar yoki bilim grafiklarini o‘z ichiga olishi mumkin.

Semantik modellar ierarxik tuzilmalar va RDF va OWL kabi rasmiy tillardan foydalangan holda tushunchalarning munosabatlari va ma’nolarini qo‘lga kiritish va kodlash uchun mo‘ljallangan.

Misol: Tibbiy sohaning ontologiyasi kasalliklar, simptomlar, davolash usullari va dorilar o‘rtasidagi munosabatlarni aniqlashi mumkin.



2.3.3-rasm. Semantik model

Ushbu modellarning har biri o‘zining afzalliklariga ega va turli xil bilimlarni namoyish qilish va ilovalar uchun mos keladi. Masalan, freymga asoslangan modellar obyektlar va ularning xususiyatlari to‘g‘risida tuzilgan ma’lumotlarni taqdim etishda samarali hisoblanadi. Mantiqiy modellar esa fikrlash va rasmiy xulosa chiqarish uchun juda mos keladi. Ishlab chiqarish tizimlari qaror qabul qilish qoidalarini ifodalash uchun foydalidir va semantik modellar tushunchalar o‘rtasidagi munosabatlarni aks ettiradi. Amalda, bu modellar birlashtirilishi yoki muayyan foydalanish holatlariga moslashtirilishi mumkin.

2.3.3. Noravshan bilim modellari

Noravshan to‘plamlar murakkab nochiziqli, noaniq obyektlarni modellashtirish va funksiyalarni approksimatsiyalashda keng qo‘llaniladi. Noravshan modellashtirish g‘oyasi shundan iboratki, obyekt parametrlari o‘rtasidagi matematik bog‘lanish odatda lingvistik qoidalarning atamalari bilan ifodalanuvchi sifatli munosabatlar bilan almashtirishdir.

Ehtimoliy fikrlash - bu bilimlarni ifodalash usuli bo‘lib, bilimdagi noaniqlikni ko‘rsatish uchun ehtimollik tushunchasini qo‘llashdir. Ehtimoliy fikrlashda noaniqlikni hal qilish uchun ehtimollik nazariyasini mantiq bilan birlashtiriladi. Ehtimoliy fikrlashga quyidagi misollarni keltirish mumkin. Real vaqtda “Bugun yomg‘ir yog‘adi” va “Ikki jamoa yoki ikkita o‘yinchi o‘rtasidagi o‘yin” kabi biror hodisani

oldindan natijasini bashorat qilish. Bunda natijalar oldindan aniq emas shuning uchun ehtimollik asoslaridan foydalaniladi.

Ehtimollik - bu noaniq voqea sodir bo‘lish ehtimoli sifatida belgilanishi mumkin. Bu voqea sodir bo‘lish ehtimolining raqamli o‘lchovidir. Ehtimollik qiymati har doim 0 dan 1 gacha bo‘lib, bu ideal noaniqliklarni ifodalaydi.

1. $0 \leq P(A) \leq 1$, bu yerda P(A) A hodisaning ehtimoli.
2. $P(A) = 0$, A hodisasida umumiy noaniqlikni ko‘rsatadi.
3. $P(A) = 1$, A hodisasida to‘liq aniqlikni ko‘rsatadi.

Noaniq hodisa yuzaga kelish ehtimolligini quyidagi formuladan foydalanib topishimiz mumkin.

$$Probabilitiy\,of\,occurrence = \frac{Number\,of\,desired\,outcomes}{Total\,number\,of\,outcomes}$$

Number of desired outcomes - ehtimoliy natijalar soni.

Probability of occurrence - hodisa ehtimoli.

Total number of outcomes- Natijalarning umumiy soni.

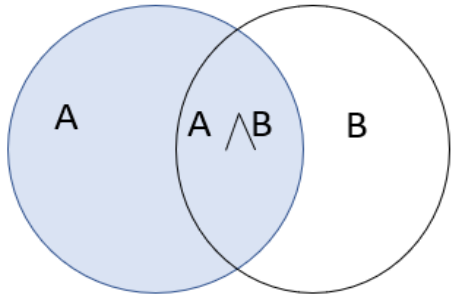
Shartli ehtimollik - bu boshqa hodisa sodir bo‘lgan paytda sodir bo‘lish ehtimoli. Aytaylik, biz B hodisasi sodir bo‘lgan A hodisasini hisoblamoqchimiz, “B shartlari bo‘yicha A ehtimoli”, uni quyidagicha yozish mumkin:

$$P(A \mid B) = \frac{P(A \wedge B)}{P(B)} \qquad (2.3.1)$$

Bu yerda P($A \wedge B$)= A va B ning birgalikdagi ehtimoli P(B)= B ning chegaraviy ehtimoli. Agar A ning ehtimolligi berilgan bo'lsa va biz B ning ehtimolligini topishimiz kerak bo'lsa, u quyidagicha hisoblanadi

$$P(B \mid A) = \frac{P(A \wedge B)}{P(A)} \qquad (2.3.2)$$

Buni quyidagi Venn diagrammasi yordamida ham aniqlash mumkin.



2.3.4-rasm. Venn diagrammasi

Misol: Bir sinfda ingliz tilini yaxshi biladigan o'quvchilar 65 foiz, ingliz tili va matematikani yaxshi biladigan o'quvchilar 35 foizi bor, ingliz tilini yaxshi biladigan o'quvchilarning necha foizi matematikani yaxshi biladi?

Yechim: A – talabaga matematikani yaxshi bilish hodisa.

B – talaba ingliz tilini yaxshi bilish hodisasi.

$$P(B \mid A) = \frac{P(A \wedge B)}{P(B)} = \frac{0.35}{0.65} = 0.54\%$$

Shunday qilib, 54% ingliz tilini yaxshi biladigan o‘quvchilar matematikani ham yaxshi biladi.

Bayes teoremasi ingliz matematigi Tomas Bayes sharafiga nomlangan. Bayes xulosasi Bayes teoremasining qo‘llanilishi bo‘lib, u Bayes statistikasi uchun asos hisoblanadi. Bayes teoremasi, shuningdek , Bayes qoidasi, Bayes qonuni yoki Bayes fikrlash deb nomlanadi. Bu noaniq bilimga ega bo‘lgan hodisaning ehtimolini belgilaydi. Ehtimollar nazariyasida u ikkita tasodifiy hodisaning shartli ehtimolligi va chegaraviy ehtimolliklarini bog‘laydi. Bu P(A|B) ni bilish bilan P(B|A) qiymatini hisoblash usulidir. Bayes teoremasi real vaqtda yangi ma’lumotlarini kuzatish orqali hodisaning ehtimollik bashoratini yangilash imkonini beradi.

Misol: Agar saraton kasalligi odamning yoshiga nisbatan qaralsa, Bayes teoremasidan foydalanib, biz saraton kasalligi ehtimolini yoshga qarab aniqroq aniqlashimiz mumkin. Bayes teoremasi mahsulot qoidasi va ma’lum B hodisasi bo‘lgan A hodisasining shartli ehtimolligi yordamida olinishi mumkin:

Mahsulot qoidasiga ko‘ra biz quyidagilarni yozishimiz mumkin:

$$P(A \wedge B) = P(A|B) \cdot P(B) \text{ yoki} \qquad (2.3.3)$$

Xuddi shunday, ma'lum A hodisasi bilan B hodisasining ehtimoli:

$$P(A \wedge B) = P(B|A) \cdot P(A) \qquad (2.3.4)$$

Ikkala tenglamaning o'ng tomonini tenglashtirib, quyidagi tenglikni hosil qilamiz.

$$P(B|A) = \frac{P(B|A) \cdot P(A)}{P(A)} \qquad (2.3.4)$$

Yuqoridagi (2.3.4) tenglama Bayes qoidasi yoki Bayes teoremasi deb ataladi. Ushbu tenglama ehtimollik xulosasi uchun zamonaviy sun'iy intellekt tizimlarining asosiy qismidir. U qo'shma va shartli ehtimollar orasidagi oddiy munosabatni ko'rsatadi. Bu yerga, P(A|B) biz hisoblashimiz kerak bo'lgan posterior deb nomlanadi va biz B dalilini yuzaga keltirganimizda u A gipoteza ehtimoli deb o'qiladi. P(B|A) ehtimollik deb ataladi, bunda biz gipotezani to'g'ri deb hisoblaymiz, keyin dalilning ehtimolligini hisoblaymiz. P (A) dalilni ko'rib chiqishdan oldin oldingi ehtimollik, gipoteza ehtimoli deb ataladi. P (B) marginal ehtimollik deb ataladi.

(a) tenglamada umuman P *(B) = P(A)·P(B|A_i)* ni yozishimiz mumkin, shuning uchun Bayes qoidasini quyidagicha yozish mumkin:

$$P(B|A_i) = \frac{P(A_i) \cdot P(B|A_i)}{\sum_{i=1}^{k} P(A_i) \cdot P(B|A_i)} \qquad (2.3.5)$$

Bu yerda A_1 , A_2 , A_3 ,....., A_n bir-birini istisno qiluvchi va to'liq bo'ladigan hodisalar to'plamidir.

Bayes qoidasini qo‘llanilishi. Bayes qoidasi P(B|A) yagona atamasini P(A|B), P(B) va P(A) ko‘rinishida hisoblash imkonini beradi . Bu uchta shartning ehtimoli yuqori bo‘lgan va to‘rtinchisini aniqlamoqchi bo‘lgan holatlarda juda foydali. Aytaylik, biz noma’lum sababning ta’sirini sezmoqchimiz va bu sababni hisoblamoqchi bo‘lsak, Bayes qoidasi quyidagicha bo‘ladi:

$$P(cause|effect) = \frac{P(effect|cause)\cdot P(cause)}{P(effect)} \qquad (2.3.6)$$

1-misol: Bemorda bo‘yin qotib qoladigan meningit kasalliklari bo‘lish ehtimoli qanday?

Berilgan ma’lumotlar: Shifokor meningit kasalligi bemorda bo‘yinning qotib qolishiga olib kelishini biladi va bu 80% hollarda sodir bo‘ladi. Shuningdek, u yana bir qancha faktlardan xabardor bo‘lib, ular quyidagicha keltirilgan:

- Bemorda meningit kasalligi bo‘lishining ma’lum ehtimoli 1/30 000 ni tashkil qiladi.
- Bemorning bo‘yni qotib qolishining ma’lum ehtimoli 2% ni tashkil qiladi.

Bemorning bo‘yni qotib qolganligi haqidagi taklif a, bemorda meningit borligi haqidagi taklif b bo‘lsin, shuning uchun biz quyidagilarni hisoblashimiz mumkin:

$$P(a|b) = 0{,}8$$

$$P(b) = 1/30000$$

$$P(a) = 0.02$$

$$P(b \mid a) = \frac{P(a \mid b)P(b)}{P(a)} = \frac{0.8 \cdot (\frac{1}{30000})}{0.02} = 0.0013333$$

Demak, 750 nafar bemordan 1 nafarida bo‘yin qotishi bilan meningit kasalligi bor deb taxmin qilishimiz mumkin.

2.3-bo‘lim bo‘yicha amaliy qism

2.3-bo‘lim bo‘yicha amaliy qism quyidagi mavzular bo‘yicha keltirilgan. Bunda keltirilgan havola yoki QR kodni skaner qilish orqali GitHubga to‘g‘ridan to‘g‘ri murojat qilish imkoniyatiga ega bo‘lasiz.

https://github.com/TUITAI/

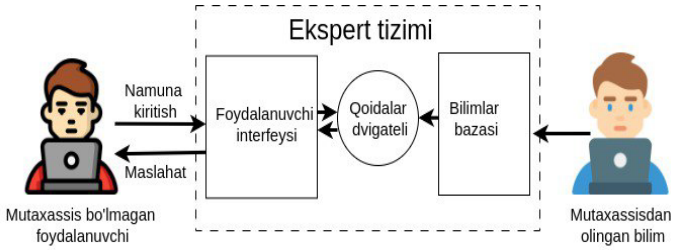
2.4. EKSPERT TIZIMLARI

Ekspert tizimi – bu murakkab muammolarni hal qilish va inson mutaxassisi kabi qaror qabul qilish qobiliyatini ta’minlash uchun mo‘ljallangan kompyuter dasturi. U buni foydalanuvchi so‘rovlariga muvofiq fikrlash va xulosa chiqarish qoidalaridan foydalangan holda bilim bazasidan bilim olish orqali amalga oshiradi.

Ekspert tizimi sun’iy intellektning bir qismidir va birinchi ET 1970-yilda ishlab chiqilgan bo‘lib, bu sun’iy intellektning birinchi muvaffaqiyatli yondashuvi edi. U o‘zining bilim bazasida saqlangan bilimlarni ajratib olish orqali mutaxassis sifatida eng murakkab masalani hal qiladi. Tizim inson mutaxassisi kabi faktlar va evristik usullardan foydalangan holda murakkab muammolarni hal qilishda yordam beradi . U ma’lum bir soha bo‘yicha ekspert bilimlarini o‘z ichiga olganligi va ushbu aniq sohaning har qanday murakkab muammosini hal qilishi mumkinligi sababli shunday nomlanadi. Ushbu tizimlar tibbiyot, fan va boshqalar kabi ma’lum bir soha uchun mo‘ljallangan.

Ekspert tizimining ishlashi uning bilimlar bazasida saqlanadigan ekspert bilimlariga asoslanadi. Bilimlar bazasida qancha ko‘p bilim saqlansa, bu tizim uning ish faoliyatini yaxshilaydi. Ekspert tizimlarinining keng tarqalgan misollaridan biri bu Google qidiruv maydoniga yozishda imlo xatolarining taklifidir.

Quyida ekspert tizimining ishlashini aks ettiruvchi blok diagrammasi keltirilgan:



3.4.1-rasm. Expert tizimlari strukturasi

Quyida Ekspert tizimining ba’zi bir misollaridan keltirib o‘tamiz.

DENDRAL - bu kimyoviy tahlil ekspert tizimi sifatida yaratilgan sun’iy intellekt loyihasi edi. U organik kimyoda noma’lum organik molekulalarni ularning massa spektrlari va kimyo bilimlari bazasi yordamida aniqlash uchun ishlatilgan.

MYCIN - bu bakteriemiya va meningit kabi infeksiyalarni keltirib chiqaradigan bakteriyalarni topish uchun mo‘ljallangan eng qadimgi ortga qaytish zanjirli ekspert tizimlaridan biri edi. Shuningdek, u antibiotiklarni tavsiya qilish va qon ivish kasalliklarini tashxislash uchun ishlatilgan.

PXDES - bu o‘pka saratoni turi va darajasini aniqlash uchun ishlatiladigan ekspert tizimi. Kasallikni aniqlash uchun u tananing yuqori qismidan soyaga o‘xshash rasmni oladi. Bu soya zararning turi va darajasini aniqlaydi.

CaDeT-CaDet ekspert tizimi saratonni erta bosqichlarda aniqlay oladigan diagnostik yordam tizimidir.

Ekspert tizimining xususiyatlari

Yuqori samaradorlik- ekspert tizimi ma’lum bir sohadagi har qanday turdagi murakkab muammolarni yuqori samaradorlik va aniqlik bilan hal qilish uchun yuqori samaradorlikni ta’minlaydi.

Tushunarlilik-foydalanuvchi tomonidan oson tushuniladigan tarzda javob beradi. U inson tilida kirishni qabul qilishi va xuddi shu tarzda chiqishni ta’minlashi mumkin.

Ishonchlilik-samarali va aniq chiqishni yaratish uchun juda ishonchli.

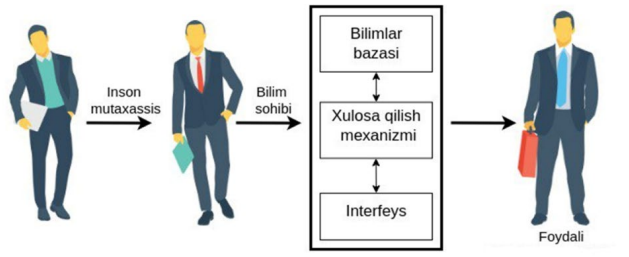
Yuqori sezgirlik- Ekpert tizimlari qisqa vaqt ichida har qanday murakkab so‘rovlar uchun natija beradi.

Ekspert tizimining komponentlari asosan uchta komponentdan iborat:

Foydalanuvchi interfeysi

Xulosa qilish mexanizmi

Bilimlar bazasi



3.4.2-rasm. Foydalanuvchi interfeysi grafigi

Foydalanuvchi interfeysi - yordamida ekspert tizimi foydalanuvchi bilan o‘zaro aloqada bo‘ladi, so‘rovlarni o‘qilishi mumkin bo‘lgan formatda kirish sifatida qabul qiladi va uni xulosa qilish mexanizmiga uzatadi. Xulosa qilish mexanizmidan javob olgandan so‘ng, u foydalanuvchiga chiqishni ko‘rsatadi. Boshqacha qilib aytganda, bu mutaxassis bo‘lmagan foydalanuvchiga yechim topish uchun ekspert tizimi bilan bog‘lanishga yordam beradigan interfeysdir .

Inference Engine - Xulosa qilish mexanizmi ekspert tizimining miyasi sifatida tanilgan, chunki u tizimning asosiy ishlov berish birligidir. U xulosa chiqarish yoki yangi ma’lumotlarni chiqarish uchun bilimlar bazasiga xulosa chiqarish qoidalarini qo‘llaydi. Bu foydalanuvchi tomonidan so‘raladigan so‘rovlarning xatosiz yechimini topishga yordam beradi. Xulosa qilish mexanizmi yordamida tizim bilimlar bazasidan bilimlarni chiqaradi. Xulosa qilish mexanizmining ikki turi mavjud:

Deterministik xulosa mexanizmi- xulosalar mexanizmidan olingan xulosalar to‘g‘ri deb hisoblanadi. Bu faktlar va qoidalarga asoslanadi .

Ehtimoliy xulosa mexanizmi-xulosa chiqarish mexanizmi xulosalarda noaniqlikni o‘z ichiga oladi va ehtimollikka asoslangan.

Xulosa qilish mexanizmi yechimlarni olish uchun quyidagi rejimlardan foydalanadi:

Oldinga siljish - u ma’lum faktlar va qoidalardan boshlanadi va ma’lum faktlarga o‘z xulosasini qo‘shish uchun xulosa qoidalarini qo‘llaydi.

Orqaga siljish - bu maqsaddan boshlanadigan va ma’lum faktlarni isbotlash uchun orqaga qarab ishlaydigan orqaga fikrlash usuli.

Bilimlar bazasi - ma’lum bir sohaning turli mutaxassislaridan olingan bilimlarni saqlash turidir. Bu bilimlarning katta ombori hisoblanadi. Bilimlar bazasi qanchalik ko‘p bo‘lsa, ekspert tizimi shunchalik aniq bo‘ladi. Ma’lum bir soha yoki mavzuning ma’lumotlari va qoidalarini o‘z ichiga olgan ma’lumotlar bazasiga o‘xshaydi. Bilimlar bazasini obyektlar va ularning atributlari to‘plami sifatida ham ko‘rish mumkin. Masalan, Arslon obyektlar va uning atributlari - bu sutemizuvchi, u uy hayvoni emas va hokazo.

Bilimlar bazasining tarkibiy qismlari. Bilim-faktlarga asoslangan va bilim muhandislari tomonidan qabul qilingan bilimlar faktik bilimga kiradi. Evristik bilimlar - bu bilimlar amaliyotga, taxmin qilish qobiliyatiga, baholashga va tajribaga asoslangan. Bilimlarni ifodalash If - else qoidalari yordamida bilimlar bazasida saqlangan bilimlarni rasmiylashtirish uchun foydalaniladi. Bilimlarni o‘zlashtirishda sohaga oid bilimlarni olish, tartibga solish va tizimlashtirish, turli mutaxassislardan bilimlarni olish qoidalarini belgilash va bu bilimlarni bilimlar bazasida saqlash jarayoni.

Ekspert tizimini ishlab chiqish - bu yerda MYCIN misolida ekspert tizimining ishlashini ko‘rib chiqamiz. Quyida MYCIN yaratish uchun bir necha qadamlar keltirilgan:

Birinchidan, ekspert tizimlari mutaxassis bilimlari bilan to‘ldirilishi kerak. MYCIN holatida bakterial infektsiyaning tibbiyot sohasida ixtisoslashgan inson mutaxassislari ushbu sohadagi sabablar, alomatlar va boshqa bilimlar haqida ma’lumot beradi. MYCIN tekshirish uchun shifokor unga yangi muammo beradi. Muammo bemorning alomatlari, hozirgi holati va tibbiy tarixni o‘z ichiga olgan tafsilotlarini kiritish orqali bakteriyalar mavjudligini aniqlashdir. Ekspert tizimiga bemor to‘g‘risidagi umumiy ma’lumotlarni, masalan, jinsi, yoshi va boshqalarni bilish uchun bemor tomonidan to‘ldiriladigan so‘rovnoma kerak bo‘ladi.

Endi tizim barcha ma’lumotlarni to‘pladi, shuning uchun xulosa chiqarish mexanizmidan foydalangan holda va bilimlar bazasi ichida saqlangan faktlardan foydalangan holda agar-then qoidalarini qo‘llash orqali muammoning yechimini topadi.

Oxir-oqibat, u foydalanuvchi interfeysi yordamida bemorga javob beradi.

Ekspert tizimini ishlab chiqish ishtirokchilari

Ekspert tizimini qurishda uchta asosiy ishtirokchi mavjud:

1. Mutaxassis. Ekpert tizimining aniqligi ko‘p jihatdan inson mutaxassislari tomonidan taqdim etilgan bilimlarga bog‘liq. Bu mutaxassislar ma’lum bir sohada ixtisoslashgan shaxslardir.

2. Bilim muhandisi. Bilim muhandisi - bu soha mutaxassislaridan bilimlarni to‘playdigan va keyin bu bilimlarni tizimga rasmiylashtirish bo‘yicha kodlaydigan shaxs.

3. Yakuniy foydalanuvchi. Bu mutaxassis bo‘lmasligi mumkin bo‘lgan ma’lum bir shaxs yoki odamlar guruhi va ekspert tizimida ishlash uning murakkab so‘rovlari uchun yechim yoki maslahatga muhtoj shaxs.

Har qanday texnologiyadan foydalanishdan oldin, biz ushbu texnologiyadan nima uchun foydalanish kerakligi va shuning uchun Ekspert tizimi uchun ham xuddi shunday fikrga ega bo‘lishimiz kerak. Garchi bizda har bir sohada insoniy mutaxassislar mavjud bo‘lsada, kompyuterga asoslangan tizimni ishlab chiqishning nima keragi bor. Shunday qilib, quyida ekpert tizimlariga bo‘lgan ehtiyojni tavsiflovchi fikrlar mavjud:

1. Xotirada cheklovlar yo‘q- kerakli darajada ko‘p ma’lumotlarni saqlashi mumkin va uni qo‘llash vaqtida eslab qolishi mumkin. Ammo inson mutaxassislari uchun har doim hamma narsani eslab qolish uchun ba’zi cheklovlar mavjud.

2. Yuqori samaradorlik- agar bilimlar bazasi to‘g‘ri bilimlar bilan yangilansa, u yuqori samaradorlikni ta’minlaydi, bu esa inson uchun imkoni bo‘lmasligi mumkin.

3. Soha bo‘yicha mutaxassislik- har bir sohada ko‘plab inson mutaxassislari mavjud va ularning barchasi turli malakalarga, turli tajribalarga va turli ko‘nikmalarga ega, shuning uchun so‘rov uchun yakuniy natijani olish oson emas. Ammo agar biz insoniy mutaxassislardan olingan bilimlarni ekspert tizimiga kiritadigan bo‘lsak, u barcha faktlar va bilimlarni aralashtirib, samarali natija beradi.

4. Hissiyotlar ta’sir qilmaydi-bu tizimlar charchoq, g‘azab, tushkunlik, tashvish va boshqalar kabi inson his-tuyg‘ulariga ta’sir qilmaydi. Shuning uchun ishlash doimiy bo‘lib qoladi.

5. Yuqori xavfsizlik- tizimlar har qanday so‘rovni hal qilish uchun yuqori xavfsizlikni ta’minlaydi.

6. Barcha faktlarni ko‘rib chiqish-har qanday so‘rovga javob berish uchun u barcha mavjud faktlarni tekshiradi , ko‘rib chiqadi va shunga mos ravishda natija beradi. Ammo inson mutaxassisi biron bir sababga ko‘ra ba’zi faktlarni hisobga olmasligi mumkin.

7. Muntazam yangilanishlar unumdorlikni yaxshilash- agar ekspert tizimlari tomonidan taqdim etilgan natijada muammo yuzaga kelsa, bilimlar bazasini yangilash orqali tizim ish faoliyatini yaxshilashimiz mumkin.

Ekspert tizimining imkoniyatlari

Quyida ekspert tizimining ayrim imkoniyatlari keltirilgan:

Maslahat berish - insonga ma’lum ekspert tizimidan istalgan soha so‘rovi bo‘yicha maslahat berishga qodir.

Qaror qabul qilish imkoniyatlarini taqdim etish-har qanday sohada qaror qabul qilish qobiliyatini ta’minlaydi, masalan, har qanday moliyaviy qarorlarni qabul qilish, tibbiyot fanida qarorlar qabul qilish va hokazo.

Qurilmani namoyish etish - har qanday yangi mahsulotlarni, masalan, uning xususiyatlari, texnik xususiyatlari, ushbu mahsulotdan qanday foydalanish va hokazolarni namoyish etishga qodir.

Muammoni tushuntirish - kirish muammosining batafsil tavsifini taqdim etishga qodir.

Kirishni tashkil qilish - foydalanuvchi tomonidan kiritilgan ma’lumotlarni sharhlash imkoniyatiga ega.

Natijalarni bashorat qilish - natijani bashorat qilish uchun ishlatilishi mumkin.

Tashxis-tibbiyot sohasi uchun mo‘ljallangan ekspert tizimlari bir nechta komponentlardan foydalanmasdan kasallikni tashxislash imkoniyatiga ega, chunki u allaqachon o‘rnatilgan turli xil tibbiy asboblarni o‘z ichiga oladi.

Ekspert tizimining cheklovlari:

Agar ma’lumotlar bazasida noto‘g‘ri ma’lumotlar mavjud bo‘lsa, ekspert tizimining javobi noto‘g‘ri bo‘lishi mumkin.

Inson kabi, u turli ssenariylar uchun ijodiy natijani ishlab chiqara olmaydi.

Uni saqlash va rivojlantirish xarajatlari juda yuqori.

Dizayn uchun bilim olish juda qiyin.

U o‘z-o‘zidan o‘rgana olmaydi va shuning uchun qo‘lda yangilanishlarni talab qiladi.

Nazorat savollari

1. Ekspert tizimi nima?
2. Bilim darajasi nima ma’no anglatadi ?
3. Foydalanuvchi interfeysi nima ma’no anglatadi?

2.4-bo‘lim bo‘yicha amaliy qism

2.4-bo‘lim bo‘yicha amaliy qism quyidagi mavzular bo‘yicha keltirilgan. Bunda keltirilgan havola yoki QR kodni skaner qilish orqali GitHubga to‘g‘ridan to‘g‘ri murojat qilish imkoniyatiga ega bo‘lasiz.

---

https://github.com/TUITAI/

III BOB

MASHINALI O‘QITISH

MASHINALI O‘QITISH

Ushbu bobda mashinali o‘qitish turlari va tasniflarini nazariy hamda amaliy yoritib o‘tamiz. Mashinali o‘qitish sun’iy intellektning bir bo‘lagi hisoblanib kompyuter ilmining bir qismidir. Bu soha bugungi kunda eng rivojlanayotgan zamonaviy sohalardan bo‘lib, ular aqlli tizimlarni ishlab chiqishda qo‘llaniladi.

Mashinali o‘qitish - ma’lumotlarni tahlil qiluvchi, ushbu ma’lumotlardan o‘rganadigan va o‘rganganlari asosida qarorlar qabul qila oladigan algoritmlarni o‘z ichiga olgan sun’iy intellektning asosiy yo‘nalishlaridan biridir.

3.1. MASHINALI O‘QITISHGA KIRISH

Mashinali o‘qitish – sun’iy intellektning bir bo‘lagi hisoblanib, kompyuterga ma’lumotlar va oldingi tajribalardan mustaqil ravishda o‘rganish imkonini beruvchi algoritmlarni yaratishga qaratilgan.

Yoshligida ta’lim olishga e’tiborsiz bo‘lgan kishi, o‘tmishini yo‘qotib, kelajagini barbod qiladi (o‘ldiradi).

Euripides (miloddan avvalgi 484 yil – 406 yil), Phrixus, Frag.

Mashinali o‘qitish-kompyuterlarga aniq dasturlovchi algoritmlar yozmagan holda o‘rganish imkonini beruvchi ilmiy soha.

Artur Samuel

Mashinali o‘qitish algoritmlari sohalarga kirib kelish tarixiga qaraydigan bo‘lsak, birinchi dasturiy algoritm 1952-yilda **Artur Samuel** tomonidan yaratilgan. Bu dastur yordamida shashka o‘yinini o‘ynash algoritmi yaratilgan bo‘lib, IBM kompyuteriga o‘rnatilgan va inson bilan to‘g‘ridan-to‘g‘ri o‘ynash qobilyatiga ega edi. Shashka o‘yini o‘ynalgan sari, o‘yin yaxshilanib borib, qaysi harakatlar asosida g‘alaba qozonish strategiyasini ishlab chiqish va bu harakatlarni saqlab qolish imkoniyatiga ham ega edi. Keyingi bosqichda 1967-yilga kelib kompyuter yordamida, bir-biriga mos xususiyatlaridan foydalanib, o‘xshash narsalarni aniqlash imkonini beruvchi “Eng yaqin qo‘shni (k-NN)” algoritmini ishlab chiqish soha uchun muhim qadam bo‘ldi. Bu algoritm asosida shaharlarga tashrif buyuruvchi sayohatchilar uchun

kerakli manzilga boruvchi eng qisqa masofani aniqlab berishda foydalanish mumkin edi. Oradan o‘n ikki yil o‘tib 1979-yilda **Stenford** universiteti talabalari xonadagi to‘siqlarni o‘zi aylanib o‘ta oladigan “Stenford aravachasi”ni ixtiro qilishdi. 1990-yillarda mashinali o‘qitish bo‘yicha ishlar bilimga asoslangan yondashuvdan ma’lumotlarga asoslangan yondashuvga o‘tdi. Olimlar katta hajmdagi ma’lumotlarni tahlil qilish va natijalardan xulosa chiqarish uchun dasturlar yaratishni boshladilar. 2014-yilda Facebook DeepFace dasturiy ta’minot algoritmini ishlab chiqdi, u fotosuratlardagi shaxslarni odamlar bilan bir xil darajada tanib olish yoki tekshirish imkoniga ega edi. Amazon 2015-yilda esa o‘zining mashinali o‘qitish platformasini ishga tushirdi. Microsoft bir nechta kompyuterlar o‘rtasida mashinali o‘qitish muammolarini samarali taqsimlash imkonini beruvchi Distributed Machine Learning Toolkit-ni yaratdi.

Mashinali o‘qitishning fanga kirib kelish tarixi quyidagicha, bu soha sun’iy intellekt usullari sinfi bo‘lib, uning xarakterli xususiyati muammoni to‘g‘ridan to‘g‘ri hal qilish emas, balki ko‘plab shunga o‘xshash muammolarning yechimlarini qo‘llash orqali o‘rganishdir. Bunday usullarni qurish uchun matematik statistika vositalari, sonli usullar, matematik tahlil, optimallashtirish usullari, ehtimollar nazariyasi, graflar nazariyasi, raqamli shakldagi ma’lumotlar bilan ishlashning turli usullari qo‘llaniladi. Yuqorida ta’kidlab o‘tganimizdek, sun’iy intellektning asosiy tarmog‘i mashinali o‘qitish hisoblanib, uning ham turli toifalari mavjud. Keyingi mavzularda mashinali o‘qitish turlari bo‘yicha ma’lumotlar keltirib o‘tamiz.

3.1.1. Mashinali o‘qitish turlari

Ma’lumotlarning qay tarzda foydalanilishiga nisbatan mashinali o‘qitishni quyidagi toifalarga ajratishimiz mumkin.

O‘qituvchili o‘qitish (*Supervised Learning*). Ushbu turdagi o‘qitish eng keng tarqalgan usul bo‘lib, belgilangan ma’lumotlardan foydalangan holda kompyuter dasturi o‘qitiladi. O‘qituvchili o‘qitishning o‘zini ham ikkita toifaga ajratish mumkin:

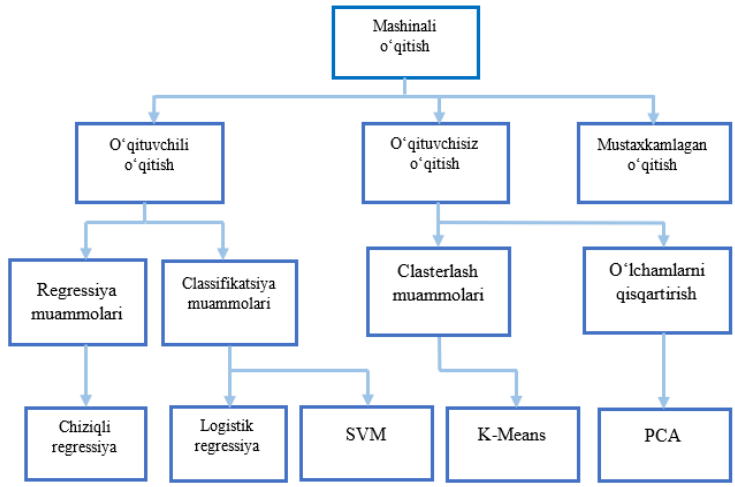
- Regressiya (Regression)
- Tasniflash (Classification)

O‘qituvchisiz o‘qitish (*Unsupervised learning*). O‘qituvchisiz o‘qitish usulidan maqsad, belgilanmagan ma’lumotlar orqali modelni o‘rgatish hisoblanadi. Masalan, bizda bir qancha meva rasmlari bor va ular belgilanmagan (ya’ni, qaysi rasmda qanday meva borligi haqida bizda ma’lumot yo‘q). Lekin o‘qituvchisiz o‘qitish usullaridan foydalanib modelni ushbu belgilanmagan rasmlar orqali o‘qitamiz, bunda model o‘rganish jarayonida ma’lumot birliklarining o‘xshashlik alomatlariga qarab (rangi, shakli, mazasi va hk.) tasniflay oladi.

Mustahkamlangan o‘qitish (*Reinforcement learning*). Mustahkamlagan o‘qitish usuli davom etayotgan jarayondagi maqbul harakatlarni mukofotlash va aksincha, nomaqbul harakatlarni jarimaga tortishga asoslanadi. Bu usulda aqlli agentdan (o‘zi mavjud bo‘lgan muhitda tajriba, foydalanuvchi kiritgan ma’lumotlar va turli shartlar asosida qaror qabul qila oladigan kompyuter dasturi elementi)

foydalanib, kerakli natijaga erishiladi. Hozirgi kunda mashinali o'qitishning ushbu usulini qo'llash barcha sohalarda ham keng tarqalmagan. Asosan, quyidagi sohalar mustahkamlangan o'qitish usulidan keng foydalanishadi:

- o'yin dasturlari
- resurslarni boshqarish
- shaxsga moslashtirilgan tavsiyalar
- robototexnika



3.1.1- rasm. Mashinali o'qitish usullari

Mashinali o'qitishda ma'lumotlarni qayta ishlash uchun ularni har xil masshtabdan bir xil masshtabga o'tkazib olish zarur. Bir xil masshtabga o'tkazib olish uchun asosan Normallashtirish va standartlashtirish funksiyalaridan foydalaniladi.

Normallashtirish - bu mashinali o'qitishning masshtablash usuli bo'lib, umumiy ma'lumotlardan foydalanish uchun, ma'lumotlar

to‘plamidagi son qiymatlarini bir xil holatga keltirishda qo‘llaniladi. Barcha ma’lumotlar bir-biriga mutanosib bo‘lsa bu usuldan foydalanish shart emas. Normallashtirish asosan kiruvchi o‘zgaruvchilarning farqini kamaytirish maqsadida foydalaniladi va quyidagi formula asosida qo‘llaniladi.

$$x_j^{(i)} = \frac{x_j^{(i)} - X_{j,min}}{X_{j,max}^{(i)} - X_{j,min}} \qquad (3.1.1)$$

$x_j^{(i)}$-ma’lum i-qatordagi va j-ustundagi o‘zgaruvchilar, $X_{j,min}$ -j ustundagi o‘zgaruvchilarning eng kichik qiymati, $X_{j,max}$ – j ustundagi eng katta qiymati. Normallashtirish funksiyasi yordamida chiziqli regressiya masalalarini yechishda foydalanish mumkin. Misol uchun oylik ish haqqi oshishini bashoratlashni ko‘rib chiqamiz. Buning uchun mavjud bo‘lgan *“salary_data”* dan foydalanamiz va u quyidagi ko‘rinishga ega:

3.1-jadval. Yillar davomida ish haqqi oshishi

№	yillar tajribasi, (X_i)	ish haqi oshishi, (Y_i)
1.	1.1	39343
2.	1.3	46205
3.	1.5	37731
:	:	:
29.	10.3	122391
30.	10.5	121872

3.1-jadvaldagi son qiymatlarni normallashtirish funksiyasi yordamida masshtablashni amalga oshiramiz. $i=0$

$$x_i = \frac{X_i - X_{min}}{X_{max} - X_{min}} \qquad y_i = \frac{Y_i - Y_{min}}{Y_{max} - Y_{min}}$$

$$x_0 = \frac{1.1-1.1}{10.5-1.1} = 0 \qquad y_0 = \frac{39343-37731}{122391-37731} = 0{,}01904$$

$$x_1 = \frac{1.3-1.1}{10.5-1.1} = 0{,}02127 \qquad y_1 = \frac{46205-37731}{122391-37731} = 0{,}10009$$

$$x_2 = \frac{1.5-1.1}{10.5-1.1} = 0{,}04255 \qquad y_2 = \frac{37731-37731}{122391-37731} = 0$$

…

$$x_{29} = \frac{10.3-1.1}{10.5-1.1} = 0{,}97872 \qquad y_{29} = \frac{122391-37731}{122391-37731} = 1$$

$$x_{30} = \frac{10.5-1.1}{10.5-1.1} = 1 \qquad y_{30} = \frac{121872-37731}{122391-37731} = 0{,}99386$$

Yuqoridagi ifoda orqali keltirilgan normallashtirish funksiyasidan foydalanib kiruvchi ma'lumotlar qiymatlarini 0 va 1 oraliqga keltirib olinadi. Bundan maqsad X_i va Y_i qiymatlarning o'rtasidagi tafovutni kamaytirish. Ya'ni: X_0 - kiruvchi ma'lumot qiymati 1.1 ga teng. Shunga mos ustundagi Y_0 - qiymati esa 39343 ga teng va birinchi o'zgaruvchiga nisbattan juda katta tafovut mavjud. Agarda kiruvchi ma'lumotlar (dataset)ni normallashtirmasdan shu holatda o'qitish amalga oshirilsa, o'zgaruvchilar o'rtasidagi tafavut va o'qitish vaqti keskin oshib ketadi. Buning oldini olish uchun normallashtirish funksiyasidan foydalanish

tavsiya etiladi. Shunga asosan, kiruvchi ma’lumotlarni normallashtirishning dasturiy algoritmini tuzib chiqamiz.

```
from numpy import *# hisoblashni tezlashtirish uchun
import matplotlib.pylab as plt
import numpy as np
n=int(input(‘n=’))# ma’lumotlar sonini kiritish uchun
f= open ('salary_data.csv', 'r')
X = []
Y = []
for i in f:
a = i.split(",")
x.append(float(a[0]))
y.append(float(a[1]))
x_min=min(X)
x_max=max(X)
y_min=min(Y)
y_max=max(Y)
x=[]
y=[]
for i in range(0, n):
x.append(((X[i])-x_min)/(x_max-x_min))
y.append(((Y[i])-y_min)/(y_max-y_min))
print(x[i], y[i])
```

Tuzilgan dasturiy algoritm asosida salary_data.csv dagi o‘zgaruvchilarni normallashtirilgan qiymatlarini jadvalga keltirib o‘tamiz.

3.2-jadval. Normallashtirish funksiyasi yordamida masshtablashtirilgan qiymatlar

№	yillar tajribasi, (X_i)	ish haqi oshishi, (Y_i)
1.	1.1	39343
2.	1.3	46205
3.	1.5	37731
:	:	:
29.	10.3	122391
30.	10.5	121872

№	(X_i)	(Y_i)
1.	0	0,01904
2.	0,02127	0,10009
3.	0,04255	0
:	:	:
29.	0,97872	1
30.	1	0,99386

Yuqoridagi jadvallarda birinchi holatda datasetning asl qiymati keltirilgan. Ikkinchi holatda esa aynan shu datasetning 0 va 1 oraliqda masshtablashtirilgan qiymatlari keltirilgan.

Standartlashtirish – odatda kiruvchi ma'lumotlar (dataset) qiymatalari to'g'ri taqsimlangan bo'lsa, standart xolatga keltirishda ishlatiladi.

$$X_j^{(i)} = \frac{x_j^{(i)} - \mu_i}{S_j} \qquad (3.1.2)$$

$x_j^{(i)}$-ma'lumot i-qatordagi va j-ustundagi o'zgaruvchilar, μ_i – j-ustundagi o'zgaruvchilarning o'rta arifmetigi, S_j – j-ustundagi o'zgaruvchilarning o'rta kvadratik og'ishi, $X_j^{(i)}$- standartlashtirilgan ma'lumotlar.

$$S_i = \sqrt{\frac{\sum(x_j^{(i)} - \mu_i)^2}{N_j}} \qquad (3.1.3)$$

j-ustun uchun o‘rta kvadratik og‘ishni hisoblaydi va bunda N_j -*j*-ustundagi o‘zgaruvchilar soni.

$$\mu_i = \frac{\sum_{j=1}^{N_i} x_j^{(i)}}{N_j} \quad (3.1.4)$$

μ_i - *j*-ustundagi o‘zgaruvchilarning o‘rta arifmetik qiymati.

3.1.2. Xatoliklarni hisoblash

Mashinali o‘qitish modellari yuqori aniqlik darajasiga erishishi uchun real qiymatlarda qanchalik yaxshi yoki yomon ishlashini tekshirish zarur. Buning uchun xatoliklarni baholovchi bir nechta funksiyasidan foydalaniladi. Bu funksiya mashinali o‘qitish modellarini aniqligini baholashda muhim parametr hisoblanadi.

Bashorat qiymatlarni baholash xatoliklari. Mashinali o‘qitishda *loss*(qiymat/yo‘qotish) funktsiyalari modelning bashorat qilgan qiymatlari haqiqiy qiymatlarga qanchalik mos kelishini baholash uchun ishlatiladi. Ular bashorat qilingan qiymatlar va haqiqiy qiymatlar o‘rtasidagi farqni aniqlaydi va bu modelni optimallashtirish uchun juda muhimdir. Turli jarayon va hodisalarni bashorat qilish uchun xatoliklarni baholashda ishlatiladigan ba’zi qiymat funksiyalari quyidagilar:

O‘rta kvadratik xatolik (Mean Squared Error MSE). Qiymat funksiyasini asosiy vazifasi xatolikni aniqlash. Bu funksiya

asosida regressiya koeffitsiyent(lar)i (w_i va b_i) qanchalik to‘g‘ri topilganini aniqlash, ya’ni ushbu koeffitsiyent(lar) orqali bashorat qilingan va haqiqiy qiymat orasidagi xatolikni topish mumkin. Buning uchun qiymat funksiyasi sifatida asosan o‘rta kvadratik xatolik funksiyasidan foydalanish tavsiya etiladi.

$$r(loss) = \frac{1}{N}\sum_{i=0}^{N}(f(x_i) - y_i)^2$$

(3.1.5)

3.1.5–formulada N-ma’lumotlar to‘plami soni y_i –ma’lumotlar to‘plami $f(x_i)$ - bashorat qilingan qiymat. Yuqoridagi keltirilgan formuladagi bashorat qiymatini $f(x_i)$ formulasi orqali yoyilsa, quyidagi formula kelib chiqadi.

$$r(kvadratik) = \frac{1}{N}\sum_{i=0}^{N}((w_i x_i + b_i) - y_i)^2$$

(3.1.6)

Bu yerda w_i - funksiya koeffitsiyenti (gradiyent), b_i -ozod had (bias) va r-xatolik (loss).

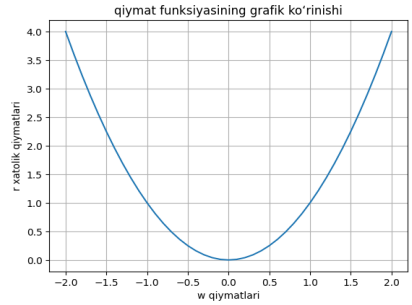
Gradiyent - bu funktsiyaning har bir parametri bo‘yicha nisbiy o‘sish yoki kamayish tezligini ko‘rsatadigan vektor kattalik.

Bias - berilgan funksiyaning ozod hadi hisoblanadi. Harbir chiqishda bitta ozod had mavjud bo‘ladi.

Variance - (farqlanish) ma’lumotlar yetishmasligi tufayli yuzaga kelgan xato.

Shovqin - ma’lumotlarni yaratadigan jarayondan kelib chiqadigan tabiiy xato. Bazi holatlarda model obyektga moslashib qolish oldini olish uchun sun’iy shovqinlar hosil qilinadi.

Yuqoridagi 3.1.6-formuladan foydalanib qiymat funksiyasini hisoblashni amalga oshirib ko‘ramiz.



3.1.2-rasm. O‘rta kvadratik xatolikni hisoblash grafigi

Yuqoridagi rasmda gradiyent qiymat o‘zgarishi bo‘yicha xatolikni hisoblash jarayoni keltirib o‘tilgan.

O‘rtacha absolyut xatolik(Mean Absolute Error MAE). Ko‘p holatlarda regressiya masalalarini yechishda qo‘llaniladi. U bashoratlar to‘plamidagi xatoliklarning o‘rtacha kattaligi hisoblanadi.

$$r(absolyut) = \frac{1}{N}\sum_{i=0}^{N}(\overline{y} - y_i)$$

(3.1.7)

Bu formulada $\overline{y}$ – bashorat qilingan qiymat. y_i – haqiqiy yorliqlangan qiymat.

Chalkashlik matritsasi(Confusion Matrix) - bu qiymat/yo‘qotish funksiyasi mashinali o‘qitishning sinflashtirish masalalarini yechishda qo‘llaniladi. Bunda bashorat qilingan qiymatlarni haqiqiy qiymat bilan taqqoslash orqali modelning ishlashi tavsiflanadi. Chalkashlik matritsasining tuzilishi quyidagicha:

- **Haqiqiy ijobiylar(True Positive TP):** Model ijobiy sinfni to‘g‘ri bashorat qilgan holatlar soni.
- **Haqiqiy negativlar (True NegativeTN):** Model salbiy sinfni to‘g‘ri bashorat qilgan holatlar soni.
- **Noto‘g‘ri pozitivlar (False Positive FP):** Model ijobiy sinfni noto‘g‘ri bashorat qilgan holatlar soni.
- **Noto‘g‘ri negativlar (False Negative FN):** Model salbiy sinfni noto‘g‘ri bashorat qilgan holatlar soni.

Chalkashlik matritsasi quyidagi formula yordamida niqlanadi.

$$\boldsymbol{Accuracy(xatolik)} = \frac{\boldsymbol{TP+TN}}{\boldsymbol{TP+TN+FP+FN}} \tag{3.1.8}$$

Misol: chalkashlik matritsasi hisoblash uchun ixtiyoriy bashorat qilingan qiymatlarni olib ko‘ramiz. $\boldsymbol{TP = 5, TP = 3, FP=1}$ va $\boldsymbol{FN=1}$ ga teng qiymatlar ustida hisoblashlarni amalga oshiramiz.

$$\boldsymbol{Accuracy(xatolik)} = \frac{\boldsymbol{5+3}}{\boldsymbol{5+3+1+1}} = \boldsymbol{0.8} \tag{3.1.9}$$

Aniqlikni(Precision) **hisoblash.** Bu modelning ijobiy bashoratlari qanchalik to‘g‘ri ekanligini hisoblash. U haqiqiy ijobiy bashoratlarning model tomonidan qilingan ijobiy bashoratlarning umumiy soniga nisbati orqali aniqlanadi.

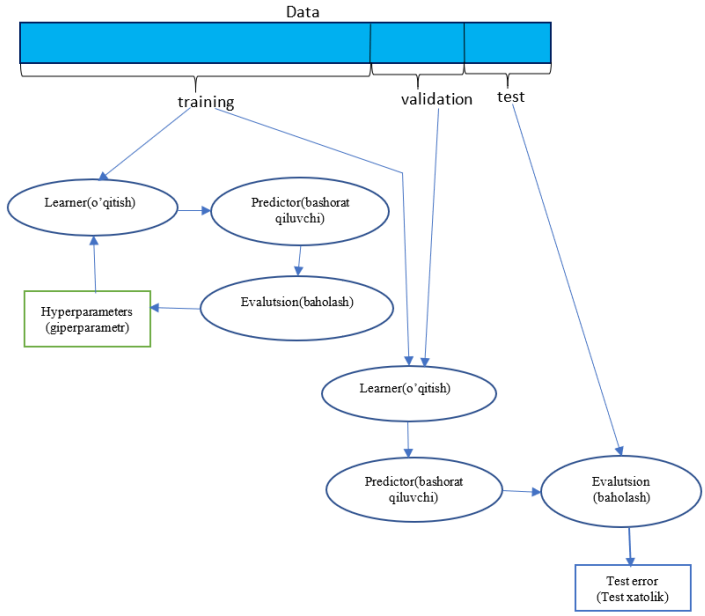
$$\boldsymbol{Aniqlik}(\text{Precision}) = \frac{\boldsymbol{TP}}{\boldsymbol{TP+FP}} \qquad (3.1.10)$$

Yuqoridagi qiymatlardan foydalanib hisoblashlarni amalga oshiramiz.

$$\boldsymbol{Aniqlik}(\text{Precision}) = \frac{\mathbf{5}}{\mathbf{5+1}} = \mathbf{0.83} \qquad (3.1.11)$$

Bu model bashoratlar asosida aniq va noaniq misollar sonini ko‘rsatish vositasidir. U ko‘pincha tasniflash modellarining ishlashini o‘lchash uchun ishlatiladi, bu esa har bir kirish namunasi uchun toifali yorliqni bashorat qilish uchun xizmat qiladi.

Cross Validation (o‘zaro tekshirish) – yangi ma’lumotlar to‘plamini, ya’ni u ilgari ko‘rmagan ma’lumotlarni qanchalik yaxshi bashoratlashini baholash uchun ishlatiladigan mashinali o‘qitishning asosiy usuli. Bu xatoliklarni baholash usuli model obyektga haddan tashqari moslashishining oldini olish va modelning ishlashi o‘quv(treaning) ma’lumotlariga ortiqcha moslashish natijasi emasligini ta’minlash uchun ishlatiladi.



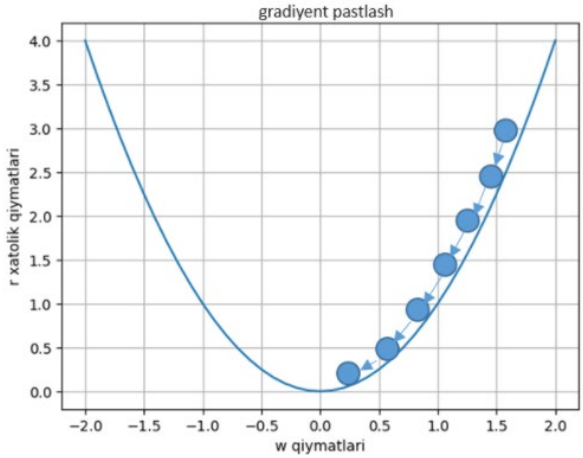
3.1.3-rasm. Modelni o'qitish va testlash jarayoni

Yuqoridagi rasmda modelni o'qitish jarayoni keltirib o'tilgan. Bunda dataset(ma'lumotlar to'plami) **o'quv (training)** to'plami va **tekshirish(test)** to'plamiga ajratilgan holati ifodalangan.

3.1.3. Gradiyent pastlash

Gradiyent pastlash – bu funksiyaning minimal qiymatini topish uchun iterativ algoritm. Uning maqsadi eng kichik yoki global minimal xato qiymatini topish uchun optimallashtirishni qo'llashdir. U asosan model parametrlarini yangilash uchun ishlatiladi. Gradiyent pastlash funksiyasi grafigi tangensining qiyaligini tavsiflovchi vektor qiymatli funksiya bo'lib, eng muhim pastlash tezligi yo'nalishini ko'rsatadi. Ushbu algoritmning asosiy maqsadi funksiyani kiritish parametrlarini

iterativ ravishda sozlash orqali minimallashtirishdir. Quydagi 3.1.5-rasmda Gradiyent pastlash grafikasini ko‘rishimiz mumkin.



3.1.5– rasm. Gradiyent pastlash grafikasi

Og‘irlik parametrini yangilashda Batch Gradiyent pastlash formulasidan foydalaniladi va u quyidagicha ifodalanadi.

$$w_{i+1} = w_i - a * \nabla_{w_i} J(w_i) i \qquad (3.1.12)$$

Bu yerda w - yangilanishi kerak bo‘lgan og‘irliklarni bildiradi. a -algoritmning o‘rganish tezligi. J-qiymat funksiyasi hisoblanadi.

Batch gradiyent-pastlash algoritmi barcha ma’lumotlar to‘plami qiymat funksiyasining gradiyentini hisoblash uchun ishlatiladi. Parametrlar gradiyentning teskari yo‘nalishi bo‘yicha yangilanib o‘rganish tezligi bilan ko‘paytiriladi. To‘plam gradiyentning tushishi hisoblash uchun murakkab bo‘lib, butun ma’lumotlar to‘plamini xotiraga yuklashni va har bir misol uchun qiymat funksiyasini baholashni talab qiladi. Batch gradiyent pastlash har bir bosqichda

barcha o'quv ma'lumotlaridan foydalangan holda gradiyentni hisoblab chiqadi. Batch Gradiyent pastlash algoritmi har bir iteratsiyada barcha o'qitish misollarini ko'rib chiqadi. Agar ma'lumotlar to'plamida ko'p sonli o'quv misollari va ko'p sonli xususiyatlar mavjud bo'lsa, Batch Gradiyent pastlash algoritmini amalga oshirish kerakli natijaga olib bormaydi.

Stoxastik Gradiyent pastlash-batch gradiyentidan farqli o'laroq, barcha namunalardan o'tmaydi. Buning o'rniga u bitta tasodifiy parametrni tanlaydi va ma'lumotlar to'plamini ma'lumotlar nuqtalarining qayd etilgan qiymatiga muvofiq optimallashtiradi, bu esa har bir yangilanishda o'rganishni amalga oshiradi. Og'irlik parametrini yangilaydigan Stoxastik Gradiyent pastlash formulasi.

$$w_{i+1} = w_i - a * \nabla_{w_i} J(x^i, y^i, w_i) \qquad (3.1.13)$$

Bu yerda y - ma'lumotlarning haqiqiy qiymati.

Mini-Batch Gradiyent pastlash-Batch Gradiyent pastlash va Stoxastik Gradiyent pastlash o'rtasidagi kesishish hisoblanadi. Ushbu yondashuvda biz butun ma'lumotlar to'plamini yoki bitta kuzatishni takrorlash o'rniga, ma'lumotlar to'plamini kichik-kichik to'plamlarga ajratamiz va har bir to'plam uchun gradiyentlarni hisoblaymiz. Og'irlik parametrini yangilaydigan Stoxastik Gradiyent pastlash formulasi.

$$w_{i+1} = w_i - a * \nabla_{w_i} J(x^{i:i+b}, y^{i:i+b}; w_i) \qquad (3.1.14)$$

Belgilar Stoxastik Gradiyent pastlash bilan bir xil, bu yerda b - bitta to'plam hajmini bildiruvchi giperparametr.

Nazorat savollari.

1) Mashinali o‘qitish qanday ma’no anglatadi?

2) Mashinali o‘qitish sohasi qaysi yillarda fanga kirib kelgan?

3)Mashinali o‘qitishning nechta usuli bor?

4) Normallashtirish nima uchun ishlatiladi?

5) O‘rta kvadratk xatolik qanday ma’no anglatadi?

6) Chalkashlik matritsasi yordamida qanday xatoliklar baholanadi?

7) Gradiyent tushush yordamida qanday muammolar bartaraf etiladi?

8) Batch gradiyent qanday holatlarda ishlatiladi?

9) Stoxastik gradiyent qanday holatlarda ishlatiladi?

10) Mini-Batch Gradiyent qanday holatlarda ishlatiladi?

3.1-bo‘lim bo‘yicha amaliy qism

3.1-bo‘lim bo‘yicha amaliy qism quyidagi mavzular bo‘yicha keltirilgan. Bunda keltirilgan havola yoki QR-kodni skaner qilish orqali GitHubga to‘g‘ridan to‘g‘ri murojat qilish imkoniyatiga ega bo‘lasiz.

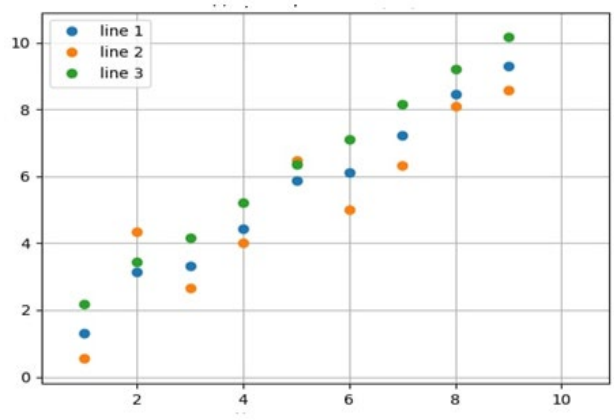
https://github.com/TUITAI/

3.2. O‘QITUVCHILI O‘QITISH ALGORITMLARI

O‘qituvchili o‘qitish – bu mashinali o‘qitish turi bo‘lib, unda mashinalar "yorliqlangan (labelled)" o‘quv ma'lumotlaridan foydalangan holda o‘qitiladi va shu ma'lumotlar asosida mashinalar natijani bashorat qiladi.

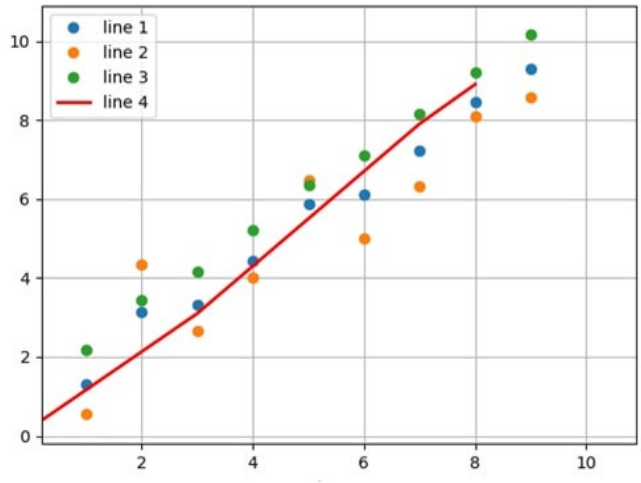
3.2.1. Chiziqli regressiya

O‘qituvchili o‘qitishning chiziqli regressiya usuli-mashinali o‘qitish usullarining eng boshlang‘ich va keng qo‘llaniladigan usullaridan biri hisoblanadi. Ushbu usul yordamida davomiy bo‘lgan qiymatlar bashorat qilinadi. Bugungi kunda bu usul yordamida signallarga raqamli ishlash jarayonlarida ham keng qo‘llanilmoqda. Buning uchun birinchi navbatda tasodifiy ma'lumotlar hosil qilib olinadi.



3.2.1-rasm. Chiziqli regressiya tasodifiy qiymatlari

Bu yerda line (1, 2 va 3) tasodifiy qiymatlar. Chiziqli regressiya usullarining asosiy muammolaridan biri, tasodifiy qiymatlardan foydalanib haqiqiy qiymatga yaqin qiymatni hosil qilish.



3.2.2-rasm. Bashorat qiymatlar yordamida haqiqiy qiymatlarga eng yaqin qiymatlarni topish

Grafikdagi 1, 2 va 3-chiziq(line) bashorat qiymatlar. 4-chiziq (line) bashorat qiymatlari asosida haqiqiy qiymatga eng yaqin qiymat. Bashorat qilingan qiymat va haqiqiy qiymat o‘rtasidagi tafovut imkon qadar kam bo‘lishi kerak. Ushbu tafovut xatolik deb qabul qilinib, qiymat funksiyasi hosil qilinadi. Xatolikni kamaytirish uchun gradiyent pastlash usullaridan foydalaniladi. Gradiyent pastlash regressiya koeffitsiyentini yangilash orqali xatolikni kamaytirishga imkon beradi. Odatda o‘qitish jarayonida dastlab koeffitsiyent(lar)ga (w_i va b_i) tasodifiy qiymat beriladi. Keyingi bosqichda qiymat funksiyasi hisoblanib (r) xatolik topiladi. Bundan asosiy maqsad xatolikning

qiymatiga qarab qiymat funksiyasini (r), chiziqli regressiya misolida esa o‘rta kvadratik funksiyasini minimallashtirish. Buning uchun gradiyent pastlash (gradiyent descent) amalga oshiriladi.

$$w_{i+1} = w_i - a \cdot \frac{2}{N} \sum_{i=0}^{N} \frac{\partial r(loss)}{\partial w}$$
$$w_{i+1} = w_i - a \cdot \frac{2}{N} \sum_{i=0}^{N} (f(x_i) - y_i) \cdot x_i \qquad (3.2.1)$$
$$w_{i+1} = w_i - a \cdot \frac{2}{N} \sum_{i=0}^{N} (w_i \cdot x_i + b_i - y_i) \cdot x_i$$

$$b_{i+1} = b_i - a \cdot \frac{2}{N} \sum_{i=0}^{N} \frac{\partial r(loss)}{\partial r}$$
$$b_{i+1} = b_i - a \cdot \frac{2}{N} \sum_{i=0}^{N} (f(x_i) - y_i) \qquad (3.2.2)$$
$$b_{i+1} = b_i - a \cdot \frac{2}{N} \sum_{i=0}^{N} (w_i \cdot x_i + b_i - y_i)$$

Yuqoridagi formulalarda $f(x_i)$-qurilgan funksiya. w_i-funksiya koeffitsiyenti(gradiyent), b_i-ozod had(bias) va r-qiymat funksiyasi xatolik(loss). (3.2.1) va (3.2.2) formulalar gradiyent pastlash qadamlari qiymatlarini topishning matematik modeli. Bu yerda a-o‘rganish qadami
(learning rate).

Misol: 3.1-jadvaldan foydalanib yillar davomida ish haqqi oshishini bashoratlashni amalga oshirib ko‘ramiz. Buning uchun 3.2(b)-normallashtirilgan jadvaldan foydalanamiz. *w=0.8*, *b=0.4 va a=0.01* qiymatlarini ixtiyoriy tanlab olamiz. 3.2.1-formula yordamida hisoblashni amalga oshiramiz.

3.3-jadval. Masshtablashtirilgan qiymatlar

№	(X_i)	(Y_i)
1.	0	0,01904
2.	0,02127	0,10009
3.	0,04255	0
:	:	:
29.	0,97872	1
30.	1	0,99386

Bu jadvaldan foydalanib gradiyent pastlashni amalga oshiramiz. Bunda datasetning 2/3 qismini o‘qitishga(trening) qolgan qismini testlashga ishlatamiz.

$$w_1 = 0,8 - 0,01 \cdot \frac{2}{20}((0,8 \cdot 0 + 0,4 - 0,01904) \cdot 0 + \\ + (0,8 \cdot 0,02127 + 0,4 - 0,10009) \cdot 0,02127 + \\ + (0,8 \cdot 0,04255 + 0,4 - 0) \cdot 0,04255 + \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ + (0,8 \cdot 0,97872 + 0,4 - 1) \cdot 0,02127 + \\ + (0,8 \cdot 1 + 0,4 - 0,99386) \cdot 0,02127)$$

$$b_1 = 0,8 - 0,01 \cdot \frac{2}{20}((0,8 \cdot 0 + 0,4 - 0,01904) + \\ + (0,8 \cdot 0,02127 + 0,4 - 0,10009) + (0,8 \cdot 0,04255 + 0,4 - 0) + \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ (0,8 \cdot 0,97872 + 0,4 - 1) + (0,8 \cdot 1 + 0,4 - 0,99386) \cdot 0,02127)$$

Hisoblash amalga oshirilgandan keyin $w_1=0.75905$ va $b_1=0.37350$ ga teng qiymat chiqadi. Bu hisoblash jarayoni bitta iteratsiya(epok) deyiladi. Bunda *gradiyent* va *bias* qiymatlarni o‘zgarishini ko‘rishimiz

mumkin. Yuqoridagi iteratsiyalarni hisoblashni osonlashtirish uchun dasturiy algoritmini tuzib chiqamiz.

```
from numpy import *
import matplotlib.pylab as plt
import numpy as np
n=int(input('n='))
f=open ('salary_data.csv','r')
x = []
y = []
for i in f:
  a = i.split(",")
  x.append(float(a[0]))
  y.append(float(a[1]))
print(max(x))
#print(x, y)
x_min=min(x)
x_max=max(x)
y_min=min(y)
y_max=max(y)
#print(x_min,x_max)
#print(y_min,y_max)
z=[]
k=[]
for i in range(0,n):
  z.append(((x[i])-x_min)/(x_max-x_min))
  k.append(((y[i])-y_min)/(y_max-y_min))
```

```
  #print(z[i], k[i])
w = []
w.append(0.8)
w_sum = 0
b_sum = 0
b = []
b.append(0.4)
for i in range(1, 20):
  w.append(w[i-1] - 0.001 * (w[i-1] * z[i-1] + b[i-1] -    k[i-1]) *
  z[i-1])
   b.append(b[i-1] - 0.001 * (w[i-1] * z[i-1] + b[i-1] -    k[i-1]))
  w_sum += w[i-1]
  b_sum += b[i-1]
new_w = w_sum/20
new_b = b_sum/20
print(new_w, new_b)
```

Natija:

new_w=0.75905 new_b=0.37350

Keyingi bosqichda hisoblab topilgan w_1 va b_1 qiymatlarini optimalligini tekshirish uchun xatolik funksiyasida tekshirib ko‘ramiz. Xatolik aniqligini dataset qiymatlariga nisbatan 0.001 dan katta 0.01 kichik qilib ixtiyoriy tanlab olamiz. Birinchi navbatda r_1 xatolikni hisoblab chiqamiz.

$$r_1(loss) = (w_1 \cdot x_1 + b_1 - y_1)^2$$
$$r_1(loss) = (0,75905 \cdot 0 + 0,3735 - 0,01904)^2 = 0,12564$$

Yuqoridagi ketma-ketlikni davom ettirib datasetning o‘qitish uchun ajratilgan barcha qiymatlari uchun xatoliklarni(r) hisoblaymiz va o‘rtacha qiymatini 0,001 dan katta 0,01 oraliqda tekshirib ko‘ramiz.

$$0,001 \le r(loss) \le 0,01$$

$$0,001 \le \frac{1}{N}\sum_{i=0}^{N}(w_i \cdot x_i + b_i - y_i)^2 \le 0,01$$

Hisoblab chiqilgan xatolikning o‘rtacha qiymati belgilab olingan oraliqqa tushsa iteratsiya jarayoni tugaydi aks holda xatolik shu oraliqqa tushguncha, ya’ni xatolik qiymati minimal qiymatga erishguncha iteratsiya davom etadi. Bu hisoblashlarning dasturiy algoritmini tuzib chiqamiz.

```
r = []
r_sum = 0
for i in range(0, 20):
  r.append(new_w * z[i] + new_b - k[i])
  r_sum += r[i]
r_new = r_sum/20
if 0.001 < r_new and r_new < 0.01:
  t = new_w
  f = new_b
```

Tuzib chiqilgan dasturiy algoritm asosida, xatolik qiymatini belgilab olingan oraliqqa tushguncha iteratsiyani davom ettiramiz. Belgilangan oraliqqa mos kelganda shart operatori yordamida shu qiymatga mos keluvchi gradiyent va biasning qiymatini boshqa o‘zgaruvchiga o‘zlashtiramiz. Topilgan gradiyent va biasning yangi

qiymatlaridan foydalanib datasetning 1/3 qismini testlashni amalga oshiramiz va xatoligini baholaymiz. Yuqoridagi dasturiy algoritmlarni birlashtirib dinamik holatga keltiramiz.

```
import matplotlib.pylab as plt
import numpy as np


n=int(input("epoklar soni n="))
m=int(input("o‘qitish sarflanadigan dataset soni m="))
count = 0
w0 = 2
b0 = 1
while True:
 if count == n:
  break
 w = []
 w.append(w0)
 w_sum = 0
 b_sum = 0
 b = []
 b.append(b0)
 for i in range(1, m):
   w.append(w[i-1] - 0.001 * (w[i-1] * z[i-1] + b[i-1] -   k[i-1]) *
   z[i-1])
  b.append(b[i-1] - 0.001 * (w[i-1] * z[i-1] + b[i-1] -  k[i-1]))
  w_sum += w[i-1]
  b_sum += b[i-1]
```

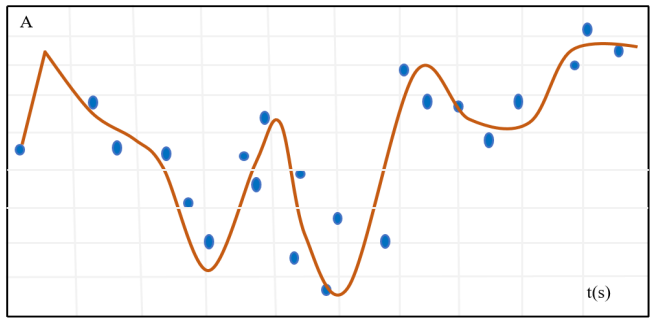
```
    new_w = w_sum/20
    new_b = b_sum/20
    w0 = new_w
    b0 = new_b
    r = []
    r_sum = 0
    for i in range(0, 20):
      r.append(new_w * z[i] + new_b - k[i])
      r_sum += r[i]
    r_new = r_sum/20
    if 0.001 < abs(r_new) and abs(r_new) < 0.01:
      t = new_w
      f = new_b
      print(t, f)
    count += 1
```

Natija: epoklar soni n=35
o‘qitish sarflanadigan dataset soni m=20
new_w1=0.36167 new_b1=0.13979
new_w2=0.34357 new_b2=0.13230

3.2.2. Polinomial regressiya

Polinomial regressiya-kiruvchi ma'lumotlari xaotik ko‘rinishda bo‘ladi. Ma'lumotlar to‘plamini taxminan ifoda etadigan chiziq o‘z-o‘zidan to‘g‘ri chiziq bo‘la olmaydi. Bunday murakkab ma'lumotlar

to‘plamini ifoda etish uchun hadlari murakkab kombinatsiyaga ega funksiyalardan foydalanish talab etiladi.



3.2.3-rasm. Polinomial regressiyaning tasodifiy qiymatlar grafigi

Kiruvchi qiymatlar murakkab xususiyatga ega ma’lumotlar to‘plami bo‘lsa, unda kvadratik, kubik va boshqa yuqori darajali funksiyalardan foydalanib murakkab model tuzib olinadi.

$$f(x) = w_0 + w_1 \cdot x_1 + w_1 \cdot x_1^{(2)} + w_2 \cdot x_2^{(3)} + w_3 \cdot x_3^{(4)} \cdots + w_n \cdot x_n^{(n)} \qquad (3.2.3)$$

Murakkab jarayonni osonlashtirish maqsadida darajaga oshirilgan ifodalarga belgilash kiritib olish kerak.

$$x_2 = x_1^2; \quad x_3 = x_1^3 \cdots x_n = x_1^n \qquad (3.2.4)$$

Belgilash amalga oshirilgandan keyin ko‘p o‘zgaruvchili chiziqli regressiya formulasiga mos tushadi va shu kabi yechiladi.

$$f(x) = w_0 + w_1 \cdot x_1 + w_2 \cdot x_2 + \cdots + w_n \cdot x_n \qquad (3.2.5)$$

Ikkinchi darajali polinomial funksiyalarning noma’lum koeffitsiyentlari 3.2.3-formuladan foydalanib topilganda, ma’lumotlarni qayta ishlash jarayoni aniqligi oshadi ammo tezligi

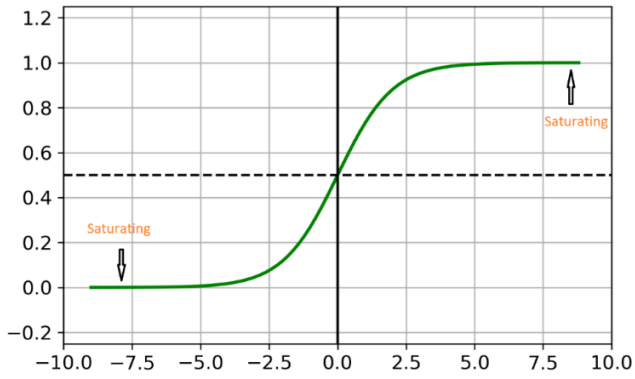
kamayib ketadi. Shunga asosan xususiyatlari ikki va undan ortiq masalalarni yechishda, o'qituvchili o'qitish usulning ikkinchi va undan yuqori darajali polinomial modellaridan foydalanish maqbul yechim hisoblanadi.

3.2.3. Logistik regressiya

Logistik regressiya - Sigmoid funksiya logistika funksiyasi deb ham ataladi va uning asosiy vazifasi kiruvchi qiymatlarning (0 va 1) oraliqdagi qiymatini aniqlashdir. Mashinali o'qitishda asosiy sigmoid funksiya yordamida bashorat qilingan qiymatlarni ehtimoliysini aniqlashda foydalanilsa, neyron tarmoqlarda bu funksiyadan faollashtiruvchi funksiya sifatida foydalaniladi.

$$\sigma(z) = \frac{1}{1+e^{-z}} \tag{3.2.6}$$

Bu formulada $\sigma(z)$ - sigmoid funksiya. Z - sigmoid funksiyaga kiruvchi qiymatlar. Sigmoid funksiya grafik ko'rinishida quyidagicha tasvirlanadi.



3.2.4 – rasm. Sigmoid funksiya grafik ko'rinishi

Grafikdan ko‘rinib turibdiki sigmoid funksiyaning minimal qiymati
0 ga cheksiz yaqinlashsa maksimal qiymati 1 ga cheksiz yaqinlashadi. Ya’ni ushbu funksiya (0,1) oraliqda yotadi.

Agar kiruvchi qiymatlar musbat sonlardan tashkil topgan bo‘lsa funksiyaning qiymati 1 ga intiladi, manfiy sonlardan tashkil topgan bo‘lsa 0 ga intiladi.

Bu funksiya neyron tarmoqlarga tadbiq qilinsa quyidagicha ko‘rinishga ega bo‘ladi. (3.2.7- formula)

$$\sigma(z) = \sigma(X \cdot A^T) \tag{3.2.7}$$

To‘liqroq ko‘rinishda quyidagicha yozish mumkin.

$$\sigma(z) = \frac{1}{1 + e^{-(X \cdot A^T)}} \tag{3.2.8}$$

Bu yerda $Z = X \cdot A^T$ qabul qilingan.

Misol: Ijtimoiy tarmoq saytlaridan olingan turli foydalanuvchilarning ma’lumotlarini o‘z ichiga olgan ma’lumotlar to‘plami mavjud. Bunda yaqinda yangi avtomobilini ishlab chiqargan avtomobil ishlab chiqaruvchi kompaniyaning sotuv ma’lumotlari bor. Ma’lumotlar to‘plamidan qancha foydalanuvchi mashinani sotib olishni xohlayotganini oldindan bashoratlashni amalga oshiramiz.

Ushbu masalani yechish uchun Logistik regressiya algoritmidan foydalanamiz.

3.4-jadval. Avtomobil sotuvi to‘g‘risida olingan dataset

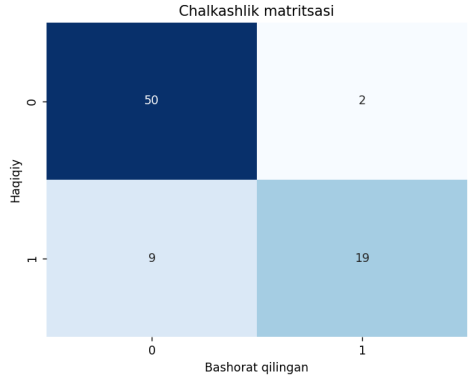
User ID	Gender	Age	EstimatedSalary	Purchased
15624510	Male	19	19000	0
15810944	Male	35	20000	0
15668575	Female	26	43000	0
15603246	Female	27	57000	0
15804002	Male	19	76000	0
15728773	Male	27	58000	0
15598044	Female	27	84000	0
15694829	Female	32	150000	1
15600575	Male	25	33000	0
15727311	Female	35	65000	0
15570769	Female	26	80000	0
15606274	Female	26	52000	0
15746139	Male	20	86000	0
15704987	Male	32	18000	0
15628972	Male	18	82000	0
15697686	Male	29	80000	0
15733883	Male	47	25000	1
15617482	Male	45	26000	1
15704583	Male	46	28000	1
15621083	Female	48	29000	1
15649487	Male	45	22000	1
15736760	Female	47	49000	1

Ma'lumotlar to'plami quyidagi 3.4-jadvalda keltirilgan. Ushbu masalada biz sotib olingan avtomobil (bog'liq o'zgaruvchi), yosh va ish haqi (mustaqil o'zgaruvchilar) yordamida bashorat qilamiz.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix, accuracy_score,
precision_score, recall_score, roc_curve, auc
```

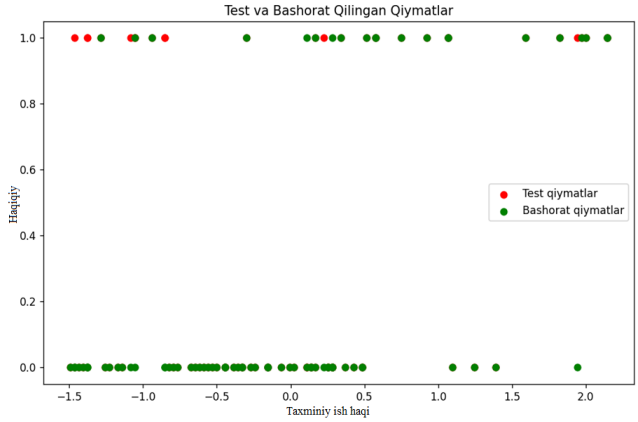
```
# Ma’lumotlarni yuklash
df = pd.read_csv("./User_Data.csv")
# X va Y ni tanlash
x = df.iloc[:, [2, 3]].values
y = df.iloc[:, 4].values
# Datasetni o‘qish va sinov qatorlarga bo‘lish
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2)
# X larni standartlash
standard_x = StandardScaler()
x_train = standard_x.fit_transform(x_train)
x_test = standard_x.transform(x_test)
# Logistik regressiya modelini yaratish
model = LogisticRegression()
# Modelni o‘qitish
model.fit(x_train, y_train)
# Test qatorida bashorat qilish
y_pred = model.predict(x_test)
# Bashorat qiymatlarini ko‘rsatish
print("y_pred:", y_pred)
# Confusion matrixni hisoblash
cfm = confusion_matrix(y_test, y_pred)
# Confusion matrixni ko‘rsatish
print("Confusion Matrix:")
print(cfm)
# Qo‘shimcha metrikalarni hisoblash
accuracy = accuracy_score(y_test, y_pred)
```

```
precision = precision_score(y_test, y_pred)
recall = recall_score(y_test, y_pred)
# Qo‘shimcha metrikalarni ko‘rsatish
print("Accuracy:", accuracy)
print("Precision:", precision)
print("Recall:", recall)
# Confusion matrixni chizish
sns.heatmap(cfm, annot=True, fmt="d", cmap="Blues", cbar=False)
plt.xlabel('Predicted')
plt.ylabel('True')
plt.title('Confusion Matrix')
plt.show()
# ROC kurvani chizish
y_prob = model.predict_proba(x_test)[:, 1]
fpr, tpr, thresholds = roc_curve(y_test, y_prob)
roc_auc = auc(fpr, tpr)
# Scatter plot chizish
plt.figure(figsize=(10, 6))
plt.scatter(X_test[:, 1], Y_test, label='Test qiymatlar', color='red')
plt.scatter(X_test[:, 1], Y_pred, label='Bashorat qiymatlar',
color='green')
plt.xlabel(' Taxminiy ish haqi')
plt.ylabel('Haqiqiy')
plt.title('Test va Bashorat Qilingan Qiymatlar')
plt.legend()
plt.show()
```



3.2.5 – Logistik regressiya algoritmi uchun chalkashlik matritsasi.

Chalkashlik matritsasidan ko‘rishimiz mumkin, modelimiz 0 sinfga tegishli jami 52 ta qiymatdan 50 tasini to‘g‘ri aniqlagan, 2 tasini 1 sinfga tegishli deb adashgan. Hamda 1-sinfga tegishli 28 ta qiymatimizdan 19 tasini to‘g‘ri aniqlagan va 9 tasini 0 sinf deb adashganligini ko‘rishimiz mumkin.

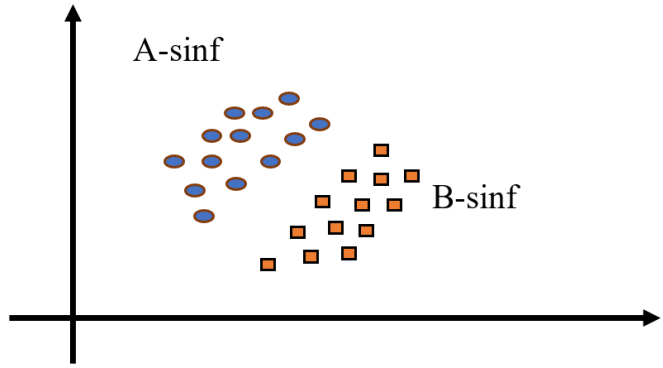


3.2.6 –rasm. Test va boshorat nuqtalar

Bu grafikdan test qiymatlari va bashorat qiymatlari qanchalik aniq topilganligini ko‘rishimiz mumkin.

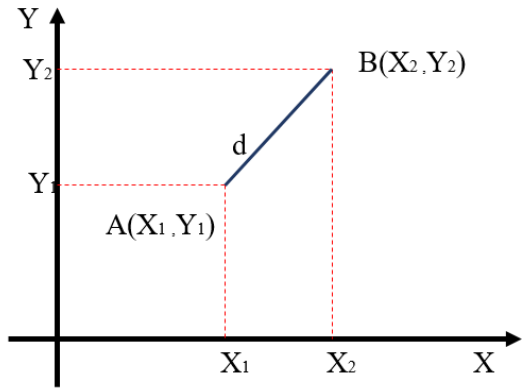
3.2.4. K-NN algoritmi

Bu algoritm mavjud ma'lumotlar yordamida eng yaqin masofani aniqlab oladi va o'xshashlik asosida yangi ma'lumotlarni tasniflaydi. K-NN algoritmi regressiya uchun ham, tasniflash uchun ham qo'llanilishi mumkin, lekin u asosan tasniflash muammolarini hal qilishda ishlatiladi. Misol uchun kiruvchi tasvirlardan aylana yoki aylana emasligini ajratish mumkin.



3.2.7-rasm. Ma'lumotlarni tasniflash

Tasniflashni amalga oshirish uchun birinchi qo'shnilar soni tanlab olinadi, odatda k=5 olinadi. Keyinchalik, ma'lumotlar nuqtalari orasidagi *Evklid masofasi* hisoblanadi.

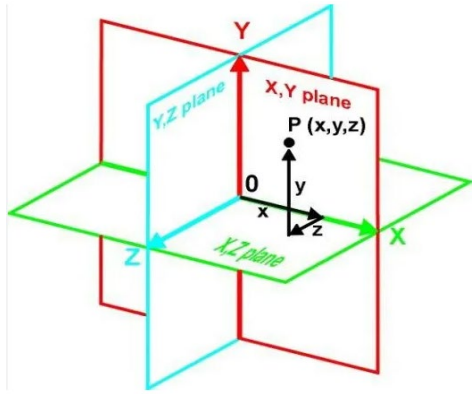


3.2.8-rasm. Eng qisqa masofani topish grafigi

Evklid masofasi ikki nuqta orasidagi masofa bo‘lib, uni quyidagicha hisoblash mumkin.

$$d = \sqrt{(X_2 - X_1)^2 + (Y_2 - Y_1)^2} \qquad (3.2.9)$$

Yuqoridagi formulada d-evklid masofasi. Agarda xususiyatlar soni ikki va undan yuqori bo‘lsa, kiruvchi parametr qiymatlari soni o‘zgaradi.



3.2.9-rasm. Uch o‘lchovli fazo

Analitik geometriyada uch o‘lchamli fazodagi har bir nuqta uchta kattalik koordinatalar to‘plami sifatida ifodalanadi. Koordinata tekisligida kesishuvchi uchta o‘zaro perpendikulyar koordinata o‘qi beriladi. Nuqtaning o‘rni shu uch o‘qqa nisbatan tartiblangan uchlik sonlarni ko‘rsatish orqali beriladi. Ushbu raqamlarning har biri tegishli o‘q bo‘ylab o‘lchanadigan boshlang‘ich nuqtadan keyingi nuqtagacha bo‘lgan masofani ko‘rsatadi, bu nuqtadan boshqa ikkita o‘q hosil qilgan tekislikkacha bo‘lgan masofaga teng bo‘ladi.

$$d = \sqrt{(X_2 - X_1)^2 + (Y_2 - Y_1)^2\,(Z_2 - Z_1)^2} \qquad (3.2.10)$$

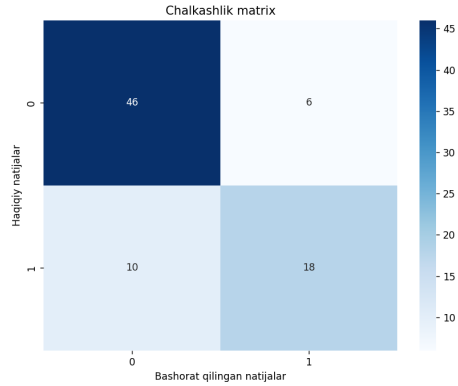
Yuqoridagi formulada d-evklid masofasini uch xil parametr asosida aniqlash keltirilgan.

Misol: Ushbu misolda ma’lumot sifatida 3.4-jadvaldagi datasetdan foydalanib sotib oluvchilarning xarid qobiliyatini bashorat qilish uchun KNN algoritmidan foydalanamiz va aniqlik darajasini tekshiramiz.

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score, confusion_matrix,
recall_score, precision_score, f1_score
import matplotlib.pyplot as plt
import seaborn as sns
# Ma’lumotlarni yuklab olish.
data = pd.read_csv('./User_Data.csv')
X = data[['Age', 'EstimatedSalary']]
```

```
y = data['Purchased']
# Ma’lumotlarni trenirovkasi va test qismiga bo‘lib ajratib olish
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)
# KNN modelini yaratib, ma’lumotlarni o‘qitish
knn = KNeighborsClassifier(n_neighbors=3)
knn.fit(X_train, y_train)
# Test ma’lumotlarini o‘qib, natijalarni bashorat qilish
y_pred = knn.predict(X_test)
# Aniqlik darajasi (accuracy)ni hisoblash
accuracy = accuracy_score(y_test, y_pred)
f1 = f1_score(y_test,y_pred)
ps = precision_score(y_test,y_pred)
rs = recall_score(y_test,y_pred)
print(f"Aniqlik darajasi: {accuracy:.2f}")
print(f"f1_score: {f1:.2f}")
print(f"precision_score: {ps:.2f}")
print(f"recall_score: {rs:.2f}")
# Confusion matrixni hisoblash
cm = confusion_matrix(y_test, y_pred)
print("Confusion matrix:")
print(cm)
# Confusion matrixni vizualizatsiya qilish
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='g‘, cmap='Blues',
xticklabels=data['Purchased'].unique(),
yticklabels=data['Purchased'].unique())
```

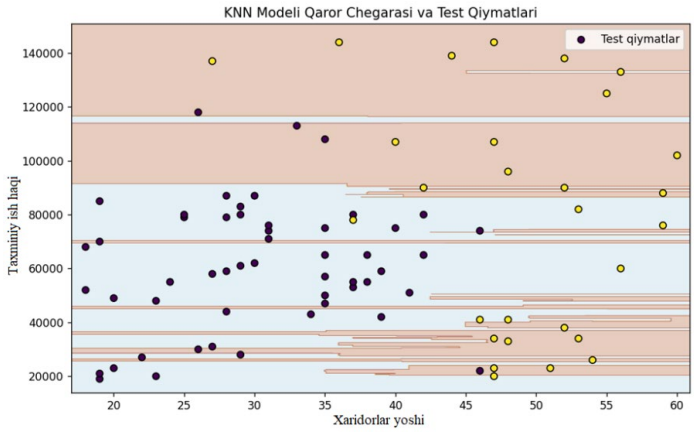
```
plt.title('Confusion matrix')
plt.xlabel('Bashorat qilingan natijalar')
plt.ylabel('Haqiqiy natijalar')
plt.show()
plt.figure(figsize=(10, 6))
plt.scatter(X_test[:, 1], Y_test, label='Test qiymatlar', color='red')
plt.scatter(X_test[:, 1], Y_pred, label='Bashorat qiymatlar',
color='green')
plt.xlabel('Tahminiy ish haqi')
plt.ylabel('Haqiqiy')
plt.title('Test va Bashorat Qilingan Qiymatlar')
plt.legend()
plt.show()
```



3.2.10 – KNN algoritmi uchun chalkashlik matritsasi

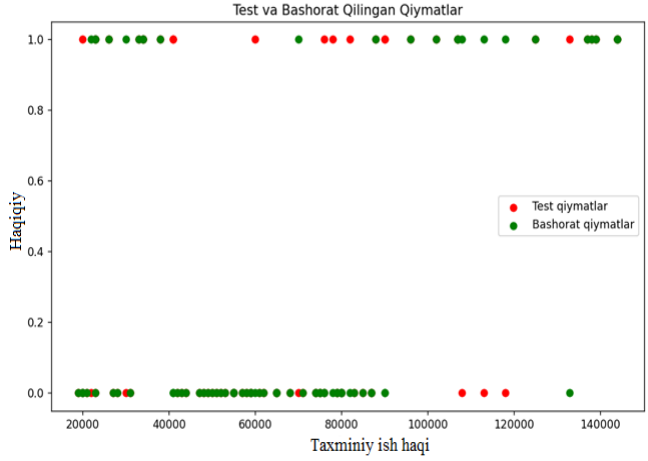
Chalkashlik matritsasidan ko‘rishimiz mumkinki, algoritmimiz 0 sinfga tegishli jami 52 ta qiymatdan 46 tasini to‘g‘ri aniqlagan 6 tasini

1 sinf deb xato qilgan. Hamda 28 ta 1-sinfga tegishli qiymatlarimizning 18 tasini to‘g‘ri aniqlagan va 10 tasida xato qilgan.



3.2.11 – KNN modeli qaror chegarasi va test qiymatlari

Bu grafikdan ko‘rishimiz mumkin, taxminiy maosh va yoshning bog‘liqlik grafigini ko‘rishimiz mumkin.

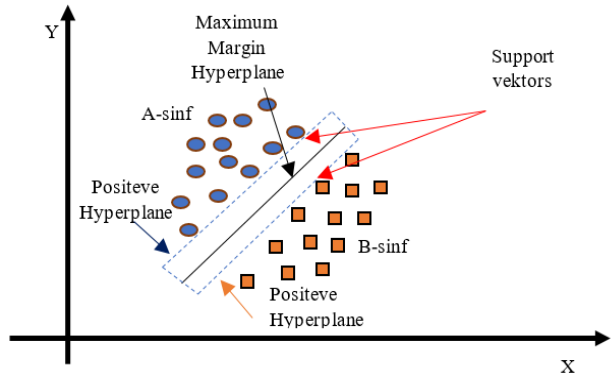


3.2.12 – Test va boshorat qiymatlar.

Bu grafikdan ko‘rishimiz mumkin, test qiymatlar va boshorat qiymatlarnig bog‘liqligini va aniq topilgan qiymatlarimizning ustma-ust tushganini ko‘rishimiz mumkin.

3.2.5. Support vektor mashinasi (SVM)

Support vektor mashinasi yoki SVM mashinali o‘qitish algoritmlaridan biri bo‘lib, u tasniflash va regressiya muammolari uchun ishlatiladi. Bugungi kunda asosan tasniflash muammolarini yechish uchun ishlatilmoqda. SVM algoritmi yordamida n o‘lchovli fazoda ma’lumotlarni sinflarga ajrata oladigan eng yaxshi chiziq yoki qaror chegarasini aniqlash mumkin. Bunda eng yaxshi qaror chegarasi giperplan deb ataladi. SVM giperplanni yaratishda yordam beradigan ekstremal nuqtalarni/vektorlarni tanlaydi. Quyidagi diagrammada qaror chegarasi yoki giperplan yordamida tasniflangan ikki xil toifa ko‘rsatilgan.



3.2.13-rasm. SVM qaror chegarasi va geperplan

SVM algoritmi ikkala sinfdagi chiziqlarning eng yaqin nuqtasini topadi. Ushbu nuqtalar qo‘llab-quvvatlash vektorlari deb ataladi. Vektorlar va giper tekislik orasidagi masofa *chekka chegara* deb ataladi. SVM ning maqsadi bu farqni (marge)ni maksimal darajada oshirishdir.

Misol: $A{\cdot}x+B{\cdot}y+C=0$ chiziqli tenglamaga qaraydigan bo‘lsak tenglamani quyidagi holatga keltirib olamiz.

$$\frac{A{\cdot}x}{A}+\frac{B{\cdot}y}{A}+\frac{C}{A}=0 \qquad (3.2.11)$$

Yuqoridagi tenglama asosida

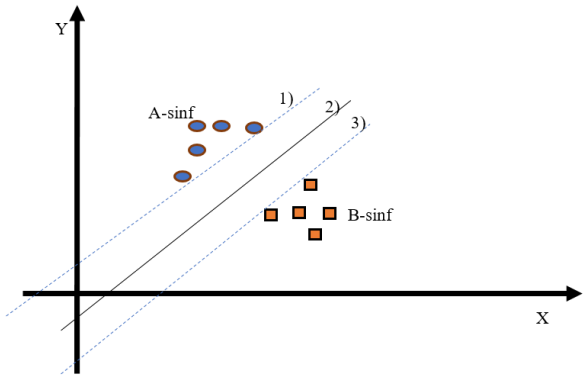
$$y=m{\cdot}x+1 \qquad (3.2.12)$$

Hisoblashni amalga oshirish uchun quyidagi jadvaldan foydalanamiz.

3.5-jadval

guruh	**x**	**y**
A	1	4
A	2	5
A	3	5
A	3	4
A	4	2
B	6	1
B	4	0
B	5	2
B	3	1
B	3	2

Yuqoridagi jadval asosida quyidagi grafikni hosil qilib olamiz



4.2.14-rasm. SVM qaror chegarasi va geperplan

1) Grafikdagi 1-to‘g‘ri chiziq tenglamasini quyidagi ko‘rinishga keltirib olamiz $w^T \cdot x_i + b = +1$

2) Grafikdagi 2-to‘g‘ri chiziq tenglamasini quyidagi ko‘rinishga keltirib olamiz $w^T \cdot x_i + b = 0$

3) Grafikdagi 3-to‘g‘ri chiziq tenglamasini quyidagi ko‘rinishga keltirib olamiz $w^T \cdot x_i + b = -1$. Tenglamalardan A va B sinflar uchun chegraviy shartlarni hosil qilamiz

$$y_i = \begin{cases} -1, & w^T \cdot x_i + b \leq -1 \\ -1, & w^T \cdot x_i + b \geq 1 \end{cases} \qquad (3.2.12)$$

Misol: Yuqoridagi ma’lumotlar uchun SVM algoritmini ishlatib ko‘ramiz va aniqlik darajalarni solishtiramiz.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
```

```
from sklearn.metrics import accuracy_score, precision_score,
f1_score, recall_score
from sklearn.preprocessing import StandardScaler
# CSV faylini Pandas DataFrame ga o‘qish
df = pd.read_csv("User_Data.csv")  # "User_Data.csv" faylini o‘z
manzilingiz bilan almashtiring


# Xususiyatlar (X) va maqsad o‘zgaruvchisi (y) ni ajratish
X = df[['Age', 'EstimatedSalary']]
y = df['Purchased']

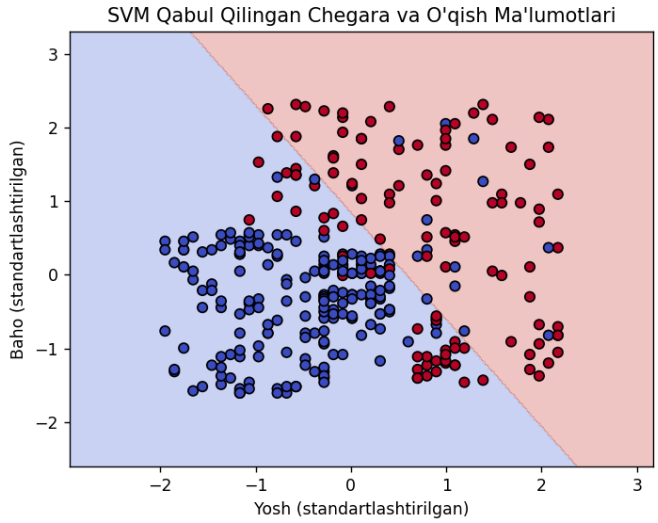

# Ma’lumotlarni o‘qish va test qismi uchun ajratish
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)
# Xususiyatlar miqdorini tekshirish - Xususiyatlar o‘zgaruvchilarini
standartlash


scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
# Linear kernel'li SVM model yaratish
model = SVC(kernel='linear')
# Modelni standartlangan o‘qish ma’lumotlari bilan o‘qitish
model.fit(X_train_scaled, y_train)
# Standartlangan test ma’lumotlariga qarab bashorat qilish
y_pred = model.predict(X_test_scaled)
# Bashoratni chiqarish
print("Aniqlik:", accuracy_score(y_test, y_pred))
```

```
print("f1_score:",f1_score(y_test,y_pred))
print("recall_score:",recall_score(y_test,y_pred))
print("precision_score:",precision_score(y_test,y_pred))
# Qabul qilingan qaror chegarasini chizish
h = .02  # chiziqlik

x_min, x_max = X_train_scaled[:, 0].min() - 1, X_train_scaled[:, 0].max() + 1
y_min, y_max = X_train_scaled[:, 1].min() - 1, X_train_scaled[:, 1].max() + 1
xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min, y_max, h))
Z = model.predict(np.c_[xx.ravel(), yy.ravel()])
# Natijani rangli grafikga joylash
Z = Z.reshape(xx.shape)
plt.contourf(xx, yy, Z, cmap=plt.cm.coolwarm, alpha=0.3)
# O‘qish nuqtalarini chizish
plt.scatter(X_train_scaled[:, 0], X_train_scaled[:, 1], c=y_train, cmap=plt.cm.coolwarm, edgecolors='k')
plt.xlabel('Yosh (standartlashtirilgan)')
plt.ylabel('Baho (standartlashtirilgan)')
plt.title("SVM Qabul Qilingan Chegara va O‘qish Ma\'lumotlari")
plt.show()
```

Ishlab chiqilgan dasturiy algoritmdan olingan natija quyidagi 3.2.15-rasmda keltirilgan.



3.2.15-rasm. SVM algoritmi asosida sinflashtirish

Bu grafikdan ko‘rishimiz mumkin, yoshning uni bahosiga bog‘liqligini va shunday to‘g‘ri chiziq o‘tkazilganki, u nuqtalarni 2 ta sinfga ajratadi.

3.2.5. Regulyarizatsiya

Ockhamning ustarasi murakkabroq bo‘lgan modellardan ko‘ra oddiyroq farazlarni afzal ko‘rish kerakligini ko‘rsatadi.

Ingliz olimi Uilyam Ockham taxminan 1285 yilda tug‘ilgan va 1349 yilda vafot etgan faylasuf.

Regulyarizatsiya - mashinali o‘qitishning eng muhim tushunchalaridan biridir. Bu usul, modelga qo‘shimcha ma’lumot qo‘shish orqali bir xil ma’lumotga haddan tashqari moslashib qolishining oldini oladi. Ko‘p holatlarda mashinali o‘qitish modellari

yordamida ma'lumotlarni o'qitganda o'quv xatolik natijasi past chiqadi, ammo test ma'lumotlarini o'qitganda xatolik yuqori chiqadi. Modelning ko'rmagan ma'lumotlar bilan testlanganda xatolik yuqori chiqishi model haddan tashqari moslashib qolganligini anglatadi. Ushbu muammoni regulyarizatsiya usuli yordamida hal qilish mumkin. Bu usuldan shunday foydalanish mumkinki, u o'zgaruvchilarning kattaligini kamaytirish orqali modeldagi barcha o'zgaruvchilar yoki xususiyatlarni saqlab qolishga imkon beradi. Demak, u modelni umumlashtirish bilan bir qatorda aniqlikni ham saqlaydi. U asosan xususiyatlar koeffitsiyentini nolga tenglashtiradi yoki kamaytiradi. Oddiy so'zlar bilan aytganda, "*Normallashtirish usulida biz bir xil sonli xususiyatlarni saqlab, xususiyatlarning hajmini kamaytirar edik"*normallashtirish usulining vazifasini bajaradi.

Regulyarizatsiya ishlash prinsipi. Regulyarizatsiya murakkab modelni optimallashtirish uchun qo'shimcha ma'lumot qo'shish orqali ishlaydi. Asosan u ikki turga bo'linadi.

- **Ridge regressiyasi**
- **Lasso regressiyasi**

Ridge regressiya - bu chiziqli regressiya turlaridan biri bo'lib, unda biz uzoq muddatli bashoratlarni yaxshiroq olishimiz uchun oz miqdordagi tarafkashlik(bias) kiritiladi. Ridge regressiyasi modelning murakkabligini kamaytirish uchun ishlatiladigan tartibga solish usuli. U **L2 regulyarizatsiyasi** deb ham ataladi . Ushbu usul xarajat funktsiyasiga jarima muddatini qo'shish orqali o'zgartiriladi. Modelga

qo‘shilgan qiymat miqdori **Ridge Regression penalty(Ridge regressiya uchun jarima)** deb ataladi. Biz uni gamma bilan belgilab alohida xususiyatning kvadrat og‘irligiga ko‘paytirish orqali hisoblashimiz mumkin. Ridge regressiyasi uchun qiymat funktsiya tenglamasi quyidagicha hisoblanadi.

$$\sum_{i=1}^{M}(y_i - \overline{y_i})^2 = \sum_{i=1}^{M}(y_i - \sum_{j=0}^{n} w_j \cdot x_{ij})^2 + \gamma \sum_{j=0}^{n} {w_j}^2$$

(3.2.13)

(3.2.13)-formulada γ-gamma regulyarizatsiya koeffitsiyenti. Yuqoridagi tenglamada jarima muddati modelning koeffitsyientlarini tartibga soladi va shuning uchun regressiyasi koeffitsiyentlarning amplitudalarini kamaytiradi, bu esa modelning murakkabligiga ta’sir qiladi. Yuqoridagi tenglamadan ko‘rinib turibdiki, agar γ **ning qiymatlari nolga yaqin bo‘lsa, tenglama chiziqli regressiya modelining qiymat funksiyasiga aylanadi.** Demak, γ ning minimal qiymatida model chiziqli regressiya modeliga o‘xshaydi. Agar qat’iy o‘zgaruvchilar o‘rtasida yuqori kollinearlik mavjud bo‘lsa, chiziqli yoki polinomli regressiya modellari obyektga moslashib qolish ehtimolligi yuqori bo‘ladi, shuning uchun bunday muammolarni hal qilishda **Ridge regressiyasidan** foydalanish mumkin. Namunalardan ko‘ra ko‘proq parametrlarga ega bo‘lsak, muammolarni hal qilishga yordam beradi.

Lasso regressiyasi - bu modelning murakkabligini kamaytirish uchun yana bir tartibga solish usuli. Bu **eng kam mutlaq va tanlash operatori** degan ma’noni anglatadi. Bu Ridge regressiyasiga

o‘xshaydi, faqat jarima atamasi og‘irliklar kvadrati o‘rniga faqat og‘irliklarni o‘z ichiga oladi. Mutlaq qiymatlarni qabul qilganligi sababli, uni 0 ga tenglashtirish mumkin, Ridge regressiyasini esa faqat 0 ga yaqinlashtirishi mumkin. **U L1 tartibga solish** deb ham ataladi . Lasso regressiyasining qiymat funktsiyasi uchun tenglama quyidagicha bo‘ladi.

$$\sum_{i=1}^{M}(y_i - \overline{y_i})^2 = \sum_{i=1}^{M}(y_i - \sum_{j=0}^{n} w_j \cdot x_{ij})^2 + \gamma \sum_{j=0}^{n} |w_j|$$

(3.2.13)

Shunday qilib, Lasso regressiyasi bizga modeldagi haddan tashqari moslashish va xususiyatlarni tanlashni kamaytirishga yordam beradi.

Ridge regressiyasi va Lasso regressiyasi o‘rtasidagi asosiy farq

- **Ridge regressiyasi** asosan modeldagi ortiqcha moslashishni kamaytirish uchun ishlatiladi va u modeldagi barcha xususiyatlarni o‘z ichiga oladi. Koeffitsyientlarni qisqartirish orqali modelning murakkabligini kamaytiradi.
- **Lasso regressiyasi** modeldagi haddan tashqari moslashishni, shuningdek xususiyatlarni tanlashni kamaytirishga yordam beradi.

Hozirgi vaqtga kelib bir nechta modellar birlashmasi orqali ma’lumotlarni qayta ishlash va ular asosida sinflashtirish hamda bashoratlash masalalari yechilmoqda. Bu modellardan biri Ensemble learning:

Ensemble learning - bu bir nechta modellarini birlashtirish orqali ularning bashorat qobiliyatlarini yaxshilashga asoslangan yondashuv.

Masalan, Random Forests va Gradient Boosting Machines (GBM) **ensemble learningga** misol bo‘la oladi. Ensemble learning usullari, alohida modellarga qaraganda ko‘pincha yaxshiroq natijalar beradi, chunki ular bir nechta bashoratchilarning “o‘rtacha fikri”ni olishadi, bu esa haddan tashqari moslashuv (overfitting) va boshqa muammolarni kamaytiradi.

Ansamblni o‘rganish - mashinali o‘qitishda kuchli texnika sifatida ajralib turadi, bu modelning ishlashi va bashorat qilish aniqligini yaxshilash uchun xizmat qiladi. Bir nechta modellarning kuchli tomonlarini birlashtirgan holda, ansambl usullari ko‘pincha har qanday bitta modeldan ustun bo‘lib, ularni mashinali o‘qitishda samarali hisoblanadi.

Nazorat savollari.

1) Chiziqli regressiya va logistik regressiya o‘rtasidagi farqni amaliy misollar bilan tushuntiring?

2) O‘rta kvadratik xatolik funksiyasi qanday ishlaydi va vazifasini tushuntiring?

3) Gradiyent tushish algoritmining asosiy vazifasi nimada to‘liqroq yoritib bering?

4) Regulyarizatsiyaning nechta turi mavjud?

3.2 bo‘lim bo‘yicha amaliy qism

3.2 bo‘lim bo‘yicha amaliy qism quyidagi mavzular bo‘yicha keltirilgan. Bunda keltirilgan havola yoki QR-kodni skaner qilish orqali GitHubga to‘g‘ridan to‘g‘ri murojat qilib dasturiy kodidan foydalanish imkoniyatiga ega bo‘lasiz.

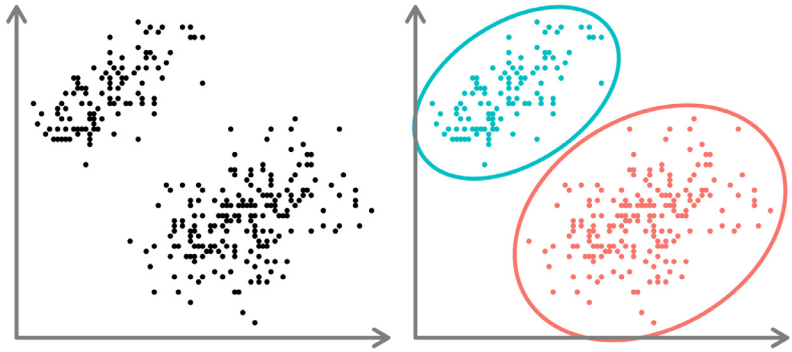
https://github.com/TUITAI/

3.3. O‘QITUVCHISIZ O‘QITISH ALGORITMLARI

O‘qituvchili o‘qitish mashinali o‘qitish algoritmlarining keng tarqalgan usuli hisoblanadi. Bu usulda mashinani o‘qitish uchun yorliqlangan ma’lumotlar to‘plamiga asoslanadi. Ammo yorliqlanmagan ma’lumotlar to‘plami asosida sinflashtirish masalalarini yechish uchun, o‘qituvchisiz o‘qitish algoritmlaridan foydalaniladi.

3.3.1. K-Means Clustering

K-Means Clustering – bu o‘qituvchisiz o‘qitish algoritmi bo‘lib, yorliqsiz ma’lumotlar to‘plamini turli klasterlarga ajratish. Bu yerda K jarayonda yaratilishi kerak bo‘lgan oldindan belgilangan klasterlar yoki guruhlar sonini belgilaydi, agar K=5 bo‘lsa, klasterlar soni beshta, K=10 uchun esa klasterlar soni o‘nta bo‘ladi va hokazo.



3.3.1-rasm. Klasterlash grafigi

K-means klasterlash algoritmi asosan ikkita vazifani bajaradi:

• K markaz nuqtalari uchun eng yaxshi qiymatni aniqlaydi.

• Har bir ma’lumot nuqtasini eng yaqin k-markaziga bog‘laydi. Guruhlar K nuqta va ma’lumot nuqtalari orasidagi masofani o‘lchash orqali K markaz nuqtalari asosida aniqlanadi.

K-Means Clusteringni qanday amalga oshirishni ko‘rib chiqamiz.

K-means klasterlash algoritmi quyidagi bosqichlarni o‘z ichiga oladi:

1-qadam: K (Klasterlar) sonini hisoblash.

2-qadam: Klaster markazi sifatida tasodifiy ravishda K ma’lumotlar nuqtalarini tanlash.

3-qadam: Evklid masofa formulasidan foydalanib, har bir ma’lumot nuqtasi va har bir klaster markazi orasidagi masofani o‘lchash amalga oshiriladi.

$$D_{masofa} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

4-qadam: Har bir ma’lumot nuqtasini markazi ushbu ma’lumotlar nuqtasiga eng yaqin bo‘lgan klasterga biriktiradi.

5-qadam: Yangi tuzilgan klasterlar markazlari qayta hisoblanadi. Klaster markazi ushbu klasterdagi barcha ma’lumotlar nuqtalarining o‘rtacha qiymatini olish orqali hisoblanadi.

6-qadam: Quyidagi yakunlash mezonlaridan birortasi bajarilmaguncha protsedurani 3-bosqichdan 5-bosqichgacha takrorlashni davom eting:

- Agar ma’lumotlar nuqtalari bir xil klasterga tushsa
- Maksimal takrorlanishga erishguncha
- Yangi tashkil etilgan klaster markaz nuqtalarida o‘zgarmaydi

Misol: Quyidagi berilgan A nuqtalar to‘plamini K=2 ga teng bo‘lgan xol uchun klasterlashni ko‘rib chiqamiz;

3.6-jadval. Klasterlash uchun ma’lumotlar to‘plami

A	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
x	1	2	5	8	3	1	3	9
y	3	2	8	5	9	7	3	4

Dastlab boshlang‘ich klaster markazlarimizni C_1=(3,3); C_2=(9,4) sifatida qabul qilsak. Yuqoridagi ma’lumotlar nuqtalari uchun 1-iteratsiyadan so‘ng yangi markazlarni topamiz.

1-qadam. Ma’lumot nuqtalari va Centroid (markas)lar orasidagi masofani toping. Qaysi ma’lumotlar nuqtalari eng yaqin klaster markaziga ko‘chiriladigan minimal masofaga ega.

$$D_{j,(masofa)} = \sqrt{\left(x - x_j\right)^2 + \left(y - y_j\right)^2}$$

1-iteratsiya.

Ma’lumot (A) nuqtalari va K (C_1, C_2) orasidagi masofani hisoblaymiz.

C_1 uchun:

$$D_1(C_1)=\sqrt{(3-1)^2+(3-3)^2}=2; \qquad D_2(C_1)=\sqrt{(3-2)^2+(3-2)^2}\approx 1,41;$$
$$D_3(C_1)=\sqrt{(3-5)^2+(3-8)^2}\approx 5,38; \qquad D_4(C_1)=\sqrt{(3-8)^2+(3-5)^2}\approx 5,38;$$
$$D_5(C_1)=\sqrt{(3-3)^2+(3-9)^2}=6; \qquad D_6(C_1)=\sqrt{(3-10)^2+(3-7)^2}\approx 8,06;$$
$$D_7(C_1)=\sqrt{(3-3)^2+(3-3)^2}=0; \qquad D_8(C_1)=\sqrt{(3-9)^2+(3-4)^2}\approx 6,08.$$

C_2 uchun:

$$D_1(C_1)=\sqrt{(9-1)^2+(4-3)^2}\approx 8,06; \qquad D_2(C_1)=\sqrt{(9-2)^2+(4-2)^2}\approx 7,28;$$
$$D_3(C_1)=\sqrt{(9-5)^2+(4-8)^2}\approx 5,65; \qquad D_4(C_1)=\sqrt{(9-8)^2+(4-5)^2}\approx 1,41;$$
$$D_5(C_1)=\sqrt{(9-3)^2+(4-9)^2}\approx 7,81; \qquad D_6(C_1)=\sqrt{(9-10)^2+(4-7)^2}\approx 3,16;$$
$$D_7(C_1)=\sqrt{(9-3)^2+(4-3)^2}\approx 6,08; \qquad D_8(C_1)=\sqrt{(9-9)^2+(4-4)^2}=0.$$

3.7-jadval. Klaster markazlarini hisoblash

Nuqtalar	**1-markaz (3,3) masofasi**	**2-markaz (9,4) masofasi**	**Klasterlash**
A(1,3)	2	8,06	C_1
A(2,2)	1,41	7,28	C_1
A(5,8)	5,38	6,65	C_1
A(8,5)	5,38	1,41	C_2
A(3,9)	6	7,81	C_1
A(10,7)	8,06	3,16	C_2
A(3,3)	0	6,08	C_1
A(9,4)	6,08	0	C_2

1-klaster (C_1) => C_1(1,3), C_1(2,2), C_1(5,8), C_1(3,9), C_1(3,3)

2-klaster (C_2) => C_2(8,5), C_1(10,7), C_1(9,4)

Endi biz yangi klasterlarni qayta hisoblaymiz va yangi klaster markazi ushbu klasterdagi barcha nuqtalarning o‘rtacha qiymatini olish orqali hisoblanadi.

1-klasterning yangi markazi =>

$$C_1 = \frac{1}{5}\left[(1+2+5+3+3),(3+2+8+9+3)\right] = \frac{1}{5}(13,25) = (2.6;5)$$

2-klasterning yangi markazi =>

$$C_1 = \frac{1}{3}\left[(8+10+9),(5+7+4)\right] = \frac{1}{3}(27,16) = (9;\ 5.3)$$

1-iteratsiya tugadi. Keyingi jarayon ham xuddi 1-itiratsiya jarayoni singari davom etadi va yangi markaz nuqtalarimizni Evklid formulasi yordamida ma’lumotlar nuqtalari va yangi markaz nuqtalari orasidagi masofani hisoblash va klaster guruhlarini topish uchun takrorlaymiz.

Nazorat savollari

1) Boshlang‘ich klaster markazlari qanday hisoblanadi?

2) K-Means Clustering qaysi turdagi algoritm ?

3) Support Vector Machine (SVM) algoritmi logistik regressiya chiziqli regressiyadan qanday farq qiladi amaliy misollar bilan tushuntiring?

3.3-bo‘lim bo‘yicha amaliy qism

3.3-bo‘lim bo‘yicha amaliy qism quyidagi mavzular bo‘yicha keltirilgan. Bunda keltirilgan havola yoki QR kodni skaner qilish orqali GitHubga to‘g‘ridan to‘g‘ri murojat qilib dasturiy kodidan foydalanish imkoniyatiga ega bo‘lasiz.

https://github.com/TUITAI/

IV BOB

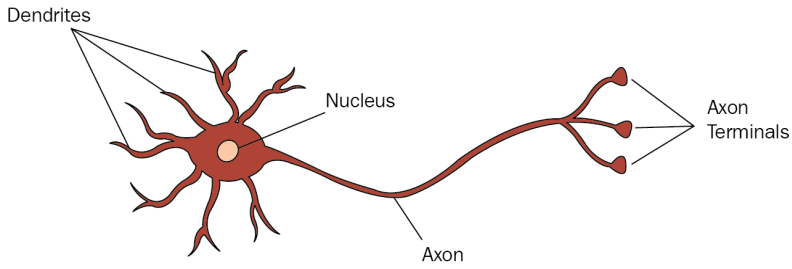
NEYRON TARMOQLAR

NEYRON TARMOQLAR

Sun’iy neyron biologik neyronlardan ilhomlanib ishlab chiqilgan matematik model va algoritmlar to‘plamidir. Sun’iy neyron bir nechta kiruvchi qiymatlarni qabul qilish, ushbu kiruvchi qiymatlarga mos chiquvchi qiymatlarni ishlab chiqish uchun ularni jamlaydigan funksiyadir.

4.1. SUN’IY NEYRON TARMOQLARGA KIRISH

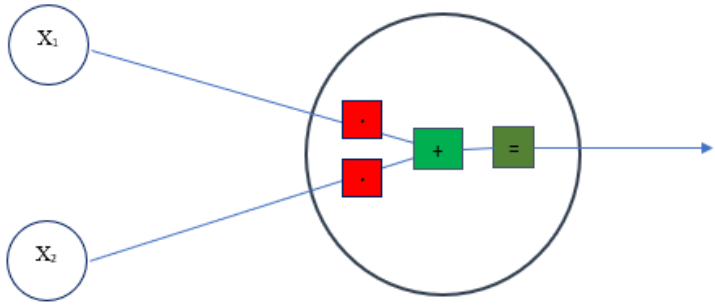
Neyron tarmoqlar – mashinali o‘qitish algoritmlarining bir turi bo‘lib, kiruvchi qiymatlar ustida turli xil hisoblashlarni va funksiyalar kombinatsiyasini qo‘llab, kerakli natijani hisoblaydi. Neyron tarmoqlar matematik modeli inson miyasidagi asab tolalarining ishlashidan ilhomlanib ishlab chiqilgan bo‘lib, bir necha qatlamlardan iborat bo‘ladi. Neyron tarmoq va neyrokompyuter texnologiyalari asoschilari strukturaviy va funksional jihatdan inson miyasiga mos keladigan elektron qurilmalarni yaratishni maqsad qilgan. Neyron deb ataladigan elementlardan tashkil topgan inson asab tizimi nihoyatda murakkab. Taxminan 10^{11} ta neyron bo‘lib, bir metr yoki undan ortiq uzunlikdagi 10^{15} ta uzatish va ulanishlarida ishtirok etadi. Har bir neyronning o‘ziga xos qobiliyati va xususiyatlari bor bo‘lib ularda qabul qilish, qayta ishlash va uzatish imkoniyatlari mavjud. Xususan, 4.1.1 - rasmda inson miyasidagi asab tolalarining bir - biriga ma’lumot almashishi ifodalovchi grafik keltirilgan.



4.1.1 – rasm. Inson asab tolalari ishlashi

Neyron uch qismdan iborat: **hujayra tanasi**(nucleus), **dendritlar**(dendrites) va **akson**(axon). Dendritlar nerv hujayrasi tanasidan boshqa neyronlarga o‘tadi va u yerda (sinapsislar) ulanish nuqtalarida kiruvchi qiymatlarni qabul qiladilar. Qabul qilingan kiruvchi qiymatlar neyron tanasiga uzatiladi. Bu yerda ular umumlashtiriladi va ba’zi kirishlar neyronni qo‘zg‘atishga intiladi, boshqalari uning qo‘zg‘alishini oldini olish uchun harakat qiladi. Neyron tanasidagi umumiy qo‘zg‘alish ma’lum chegaradan oshib ketganda, neyron qo‘zg‘aladi va akson bo‘ylab boshqa neyronlarga signal yuboradi.

Neyron kirish o‘zgaruvchilarning ustida matematik amallar bajaradi va natijani chiqaradi, bu neyron elementi deb ataladi.



4.1.2 – rasm. Neyron elementi

Ko‘rinib turibdiki neyronga o‘zgaruvchilari (x_1,x_2 va x_3) ko‘rinishda kiritiladi va ular ustida turli matematik amallar bajariladi. Keyin esa natija neyron tarmog‘ining boshqa qatlamlariga uzatiladi (ushbu element oxirgi qatlam bo‘lsa, u holda natija bashorat qilingan natija sifatida tanlanadi). Neyronning kirishi ma’lum bir ma’lumot

to‘plamini oladi, ularning har biri boshqa neyronning chiqishidir. Har bir kirish sinaptik quvvatga o‘xshash mos keladigan og‘irlik bilan ko‘paytiriladi va barcha ma’lumotlar yig‘iladi hamda faollashuv darajasi aniqlanadi. Kirish o‘zgaruvchilari tegishli “og‘irlik” w_1, w_2, ..., w_n bilan ko‘paytiriladi. O‘zgaruvchilari ustida matematik amallar, chiziqli regressiyada bo‘lganidek *f(x)* funksiya hisoblab olinadi.

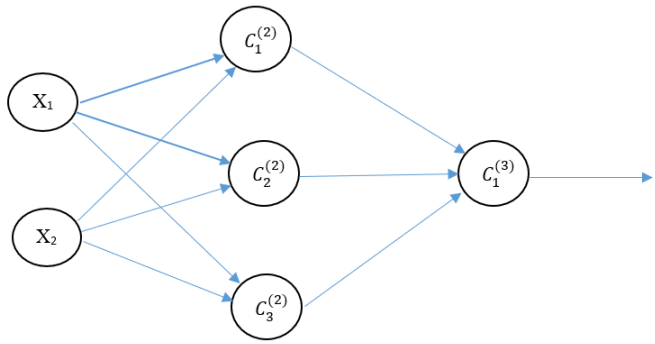
$$f(x) = w_1 \cdot x_1 + w_2 \cdot x_2 + w_3 \cdot x_3 \cdots\cdots w_n \cdot x_n + b \qquad (4.1.1)$$

w_0 va x_0 o‘rniga b ya’ni bias qo‘llanilmoqda, chunki amallarni bajarish jarayoni uzluksiz. Hisob kitoblarda, ayniqsa, gradiyent tushish amalida ushbu usul qulayroq bo‘ladi. Formulaning vektor ko‘rinishi

$$Z = f(x) = X \cdot A^T \qquad (4.1.2)$$

w_1, w_2…… w_n va b - neyronning parametrlari bo‘lib, n - kiruvchi elementlar soniga bog‘liqdir. f(x) funksiyani hisoblab olganimizdan so‘ng aktivlashtirish (h_a,(x)) funksiyasini tanlab olishimiz zarur. Aktivlashtirish funksiyasi sifatida Sigmoid, Relu, Softmax, Radial bazis va boshqa funksiyalar ishlatiladi.

Neyron tarmoqlar arxitekturasi – bir nechta neyronning o‘zaro tartiblangan holda bir – biri bilan bog‘lanishi *neyron tarmoq* deb ataladi. Neyron tarmoqning kirish qatlamida kirish o‘zgaruvchilari joylashadi, yashirin qatlamda esa bir qancha neyron qatlamlari bo‘ladi. Yashirin qatlamning o‘zi ham bir nechta qatlamlardan iborat bo‘lishi mumkin. Chiqish qatlamida esa tarmoqdan chiquvchi natijalarni hisoblovchi neyronlar joylashadi.



4.1.3-rasm. Eng sodda neyron tarmoq arxitekturasi

Yuqoridagi rasmda 3 ta qatlamdan iborat bo'lgan neyron tasvirlangan. Birinchi qatlam ***kirish qatlami***, ikkinchi qatlam ***yashirin qatlam*** va uchinchi qatlam ***chiqish qatlami***. Bu yerda x_i - kiuvchi o'zgaruvchilar. $C_i^{(2)}$- yashirin qatlam natijalari. $C_i^{(3)}$ – chiqish qatlami natijalari. Neyron tarmoqlarda oldinga va orqaga siljish alohida ahamiyat kasb etadi. Ikki holat uchun ham alohida algoritmlar ishlab chiqilgan.

4.1.1. Eng sodda perseptron

Eng sodda perseptron — bu sun'iy neyronlarning eng oddiy va asosiy turi bo'lib, u binar (ikkilik holat) tasniflash vazifalarini bajarish uchun mo'ljallangan. Perseptron 1943-yilda Uorren Makkaloch va Uolter Pits tomonidan ixtiro qilingan .

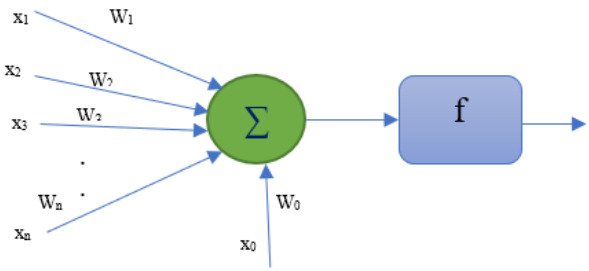
$$W_k = \{w_1, w_2, w_3 + \ldots\ldots + w_N\} \qquad (4.1.3)$$

$$x(t) = \{x_1, x_2 + \ldots\ldots + x_N\} \qquad (4.1.4)$$

Boshqacha qilib aytganda $x(t)$ vektor komponentlari w_i vazn koeffitsiyentlari bilan quyidagi keltirilgan 4.1.5-formula asosida hisoblanadi.

$$u_i = \sum_{i=1}^{N} w_i \cdot x_i \qquad (4.1.5)$$

Bu yerda: u_i - bashoratlashdan hosil bo'lgan qiymatlar yig'indisi.



4.1.4-rasm. Eng sodda neyron modeli

Neyrondan chiquvchi o'zgaruvchilar qiymati quyidagi formula orqali aniqlanadi.

$$y_i = f(u_i) \qquad (4.1.6)$$

Bu yerda y_i - neyronning aktivlash funksiyasi. Odatda aktivlash funksiyasi sifatida uzluksiz funksiyalar ishlatiladi.

Misol: Eng sodda perseptron tarmog'idan foydalanib hisoblashlarni amalga oshiramiz. Model tarkibiy qismlari quyidagilardan iborat.

1. Input – x_i

2. Vazn - w_i
3. Bias - b
4. Aktivlash funksiyasi $y_i = \frac{1}{1+e^{-f(u_i)}}$
5. Sigmoid funksiya
6. Chiziqli model $u_i = w_i \cdot x_i + b$
7. Kiruvchi qiymatlar:

$x_0=2$, $w=0.5$, $b=1$ bu qiymatlar ixtiyoriy tanlab olingan.

1-qadam. Chiziqli modelni hisoblash.

$$u_0 = w_0 \cdot x_0 + b$$

$$u_0 = 0.5 \cdot 2 + 1 = 2$$

2- qadam. Aktivlash funksiyasini hisoblash.

$$y_0 = \frac{1}{1+e^{-f(u_0)}}$$

$$y_0 = \frac{1}{1+e^{-2}} \approx 0.8808$$

Yuqoridagi neyron tarmoq qatlamidan birinchi chiquvchi qiymat natijasini hisoblashni amalga oshirdik.

Neyron tarmoqni o'rganish, vaqt o'tishi bilan tuzilishiga tuzatishlar kiritish qobiliyati uni mashinani o'rganish sohasida juda foydali qiladi . Bu atamani 1959-yilda chop etilgan "Shashka o'yinidan foydalangan holda mashinani o'rganish bo'yicha ba'zi tadqiqotlar" nomli maqoladan kelib chiqish mumkin, unda kompyuter olimi Artur Samuel shashka o'ynash uchun "o'z-o'zini o'rganish" dasturini tavsiflaydi. Kompyuterga aniq dasturlashsiz o'rganish imkonini beruvchi algoritm tushunchasi mashinali o'qitishning asosidir.

4.2. CHUQUR O‘QITISH ALGORITMLARI

Chuqur o‘qitish - sun’iy neyron tarmoqlarga asoslangan mashinali o‘qitish va sun’iy intellektning kichik bir bo‘lagi hisoblanadi. Chuqur o‘qitishda *chuqur* so‘zi neyron tarmog‘idagi qatlamlar sonini anglatadi.

Chuqur o‘qitish(Deep Learning)-bu inson miyasining tuzilishi va funksiyasi asosida ishlab chiqilgan algoritmlar to‘plami. U mashinani o‘rgatish va aniq natijalarni bashorat qilish uchun juda ko‘p tuzilgan ma’lumotlardan foydalanadi. Mashinali o‘qitish va chuqur o‘qitish texnologiyalari o‘rtasidagi asosiy farq ma’lumotlarni qayta ishlash jarayoni hisoblanadi. Mashinali o‘qitishda o‘rganish uchun saralangan va saralanmagan ma’lumotlardan foydalaniladi, chuqur o‘qitishda esa modellarni o‘rgatish uchun neyron tarmoqlardan foydalaniladi. Mashinali o‘qitishda, agar model noto‘g‘ri natijalarni bashorat qilsa, biz uni nazorat ostida to‘g‘irlashimiz mumkin. Chuqur o‘qitish usullarida esa bu muammolar avtomatik ravishda hal qilinadi va biror o‘zgartirish kiritish shart emas. O‘z-o‘zidan boshqariladigan avtomobillar chuqur o‘qitish tushunchasiga eng yaxshi misol bo‘la oladi. *Chuqur o‘qitish algoritmlari yordamida tasvirni tanish, ovozni aniqlash, mahsulotni tavsiya qilish tizimlari, tabiiy tilni qayta ishlash* va aniqroq bashoratlash muammolarini hal qilish mumkin. Chuqur o‘qitish kirish qatlamlari, yashirin qatlamlar va chiqish qatlamlari kabi turli qatlamlarga ega bo‘lgan neyron tarmoqlarni o‘z ichiga oladi. Kirish qatlami kiritilgan ma’lumotlarni qabul qiladi. Yashirin qatlamlar ma’lumotlardan har

qanday yashirin shakl va xususiyatni ajratib olish uchun ishlatiladi va chiqish qatlamlari kutilgan natijalarni chiqarib beradi. Chuqur o'qitish tarmoqlarining bir nechta turlari mavjud. Bular quyidagilar:

Oldinga harakatlanuvchi neyron tarmoqlar(Feedforward Neural Networks)

Ko'p qatlamli perseptron

Konvolyutsion neyron tarmoq (CNN)

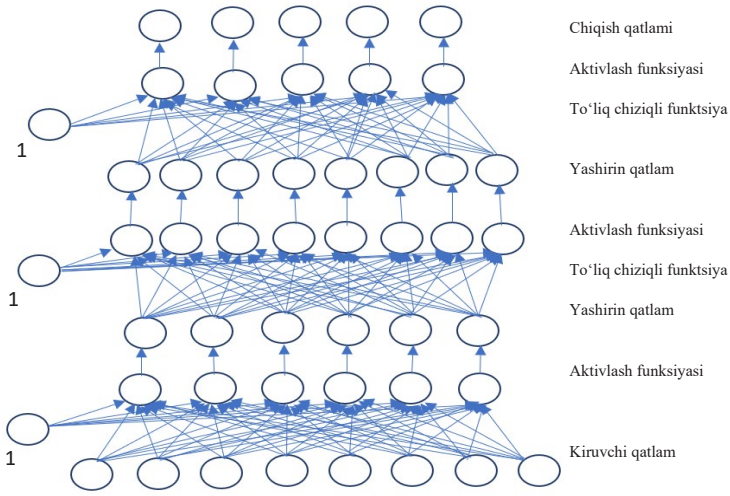
Takroriy neyron tarmoq (RNN)

Neyron tarmoqlarining ko'plab turlari mavjud ulardan biri **Oldinga harakatlanuvchi(feedforward)** neyron tarmoq berilgan x kirishlari bo'yicha bashoratlash funktsiyasini quyidagicha amalga oshiradi:

$$f(x) = f_n(f_{n-1}(\ldots f_2(f_1(x)))) \qquad 4.1.7$$

Har bir f_i funktsiya qiymatlar **vektorini** (massiv yoki ro'yxat) boshqa qiymatlar vektoriga biriktiradi. f_i funktsiya i- **qatlam** bo'lib, kompozitsiyaga kiritilgan funksiyalar soni, n neyron tarmog'ining **chuqurligini** bildiradi. f_n oxirgi qatlam **chiqish qatlami** hisoblanadi. Boshqa qatlamlar esa **yashirin qatlamlar** deb ataladi. Ba'zida x **kirish qatlami** deb ataladi. Qatlamning chiqish vektorining har bir komponenti **birlik** deb ataladi. **"Chuqurlik"** (deep) atamasi chuqur o'rganishda tarmoq chuqurligini ko'rsatadi. Oldinga harakatlanuvchi neyron tarmoqlar har bir f_i qatlam kirishi berilgan holda chiqariladigan

qiymatlarning o‘rganiladigan parametrlari bilan chiziqli funksiya bo‘lib, keyin chiziqli bo‘lmagan aktivatsiya funksiyasi qo‘llaniladi.



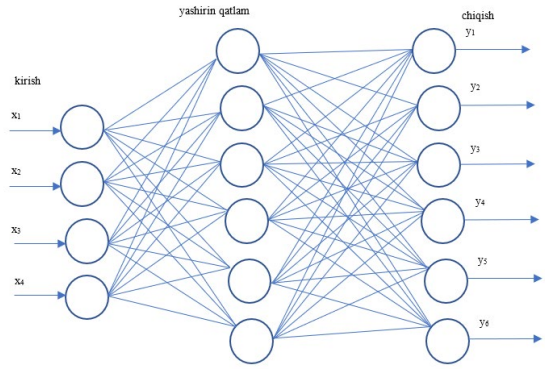
4.1.5-rasm.Uchtalik chuqurlikka ega oldinga harakatlanuvchi neyron tarmoq

Pastda kirish xususiyatlari berilgan kirish qatlami joylashgan. Yuqorida esa kerakli xususiyatlar uchun bashoratlar qiladigan chiqish qatlami mavjud. Har bir zich chiziqli funktsiyada chap tomondagi kirish qiymati 1 ga teng bo‘lib, har bir yoy uchun parametr mavjud. Har bir qiymatga aktivatsiya funktsiyasi qo‘llaniladi. Neyron tarmoqlar vaznli summalarni hisoblab chiqadi. Og‘irliklarning hisoblangan yig‘indisi yashirin qatlamlarda faollashtirish funktsiyasiga kirish sifatida uzatiladi. Faollashtirish funktsiyasi kirishlarni kerakli chiqishga moslashtirish funktsiyasidir. Bu funktsiyaga kirish sifatida “kirishning vaznli yig‘indisi” ni oladi, egilish qo‘shadi va neyronni yoqish kerakmi yoki yo‘qligini hal qiladi. Chiqish qatlami prognoz qilingan natijani beradi

va model chiqishi haqiqiy chiqish bilan taqqoslanadi. Neyron tarmoqni o‘rgatgandan so‘ng, model tarmoqning ishlashini yaxshilash uchun orqaga tarqalish usulidan foydalanadi. Qiymat funksiyasi xatolik darajasini kamaytirishga yordam beradi. Kirish qatlamidagi neyronlar soni ma'lumotlardagi xususiyatlar soniga teng va juda kam hollarda, tarafkashlik uchun bitta kirish qatlami bo‘ladi. Holbuki, chiqishdagi neyronlar soni modelning regressor yoki tasniflagich sifatida ishlatilishiga bog‘liq. Agar model regressor bo‘lsa, chiqish qatlami faqat bitta neyronga ega bo‘ladi, lekin agar model klassifikator bo‘lsa, u modelning sinf yorlig‘iga qarab bitta neyron yoki bir nechta neyronga ega bo‘ladi.

4.2.1. Ko‘p qatlamli perseptron

Eng ko‘p tarqalgan neyron tarmoq modeliga misol qilib ko‘p qatlamli perseptronni (KQP) keltirish mumkin. KQP ning strukturasi sxemasi quyidagi 4.2.1-rasmda keltirilgan.



4.2.1 – rasmda. Uch qatlamli perseptron

Neyron tarmoq optimal bashoratlashni amalga oshirish uchun o‘qitish (Back Propagation Error (BP))ni amalga oshirish zarur. BP – algoritm talab qilingan va chiquvchi tarmoq tomonidan generatsiyalangan qiymat funksiyasini minimallashtirilgan optimizatsiyalashning gradiyentga asoslangan algoritmi sanaladi.

O‘qitish(Trening)ning maqsadi-neyronlar orasidagi bog‘lanishlarning vaznlarini to‘g‘irlash orqali chiqish va kirishning kerakli funksional nisbatlarini aniqlashdir. Ko‘p qatlamli perseptrondan foydalanishda asosiy maqsad omillari sifatida quyidagilarni keltirish mumkin. Ko‘p o‘zgaruvchili ma’lumotlarni sinflashtirishda perseptronning chiquvchi sinflarida taqdim etishni amalga oshiradi va unga tegishli har bir sinf ma’lumotlari uchun parametrlarini optimallashtiradi. Perseptron sinflashtirish uchun cheklovlarning optimal kombinatsiyasini topishi mumkin. Buning uchun kirish belgilarini taqsimlash to‘g‘risida qat’iy taxminlarga hojat yo‘q.

Perseptron–bu parallel tizimlardan foydalanishga imkon beradigan yuqori darajadagi tezkorlikka ega struktura.

Ko‘p qatlamli perseptronning kamchiligi vaqt o‘zgarishiga bog‘liq bo‘lmagan ketma-ketliklar uchun emas, balki statik ma’lumotlarni tanish uchun ishlab chiqilgan.

4.2.2. ReLU funksiyasi

Faollashtirish funksiyalari neyron tarmoqlarida (NN) ajralmas rol o‘ynaydi, chunki ular tarmoqqa chiziqli regressiyadan ko‘ra murakkabroq xususiyatlar va funksiyalarni o‘rganish imkonini

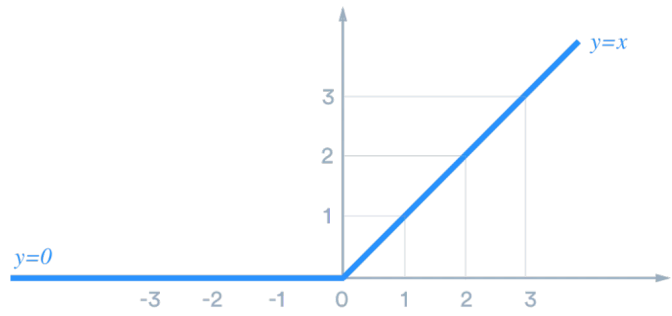
beruvchi chiziqli bo‘lmagan jarayonni hosil qiladi. Eng ko‘p qo‘llaniladigan faollashtirish funksiyalaridan biri bu Rectified Linear Unit (ReLU) bo‘lib, u nazariy jihatdan NN larga keng ko‘lamli uzluksiz funksiyalarni yaqinlashtirish imkonini beradi.

ReLU neyron tarmoqlarni o‘qitishda keng tarqalgan muammolardan biri bo‘lgan, gradiyent yo‘qolish muammosining oldini olish qobiliyati tufayli boshqa aktivlash funksiyalaridan ajralib turadi. Yo‘qolgan gradiyent muammosi, yo‘qotish funksiyasining gradiyentlari juda kichik bo‘lib, o‘qitish jarayonini sekinlashtirganda yoki uning butunlay to‘xtab qolishiga olib kelganda paydo bo‘ladi. ReLU musbat kirishlar uchun doimiy gradiyent 1 ga ega bo‘lgani uchun bu muammoni bartaraf etishga yordam beradi.

Matematik jihatdan u

$$f(x) = x^+ = \max(0, x) = \frac{x + |x|}{2} = \begin{cases} x & \text{if } x > 0, \\ 0 & \text{otherwise,} \end{cases} \qquad (4.2.1.)$$

sifatida aniqlanadi. Vizual ravishda, u quyidagicha ko‘rinadi:

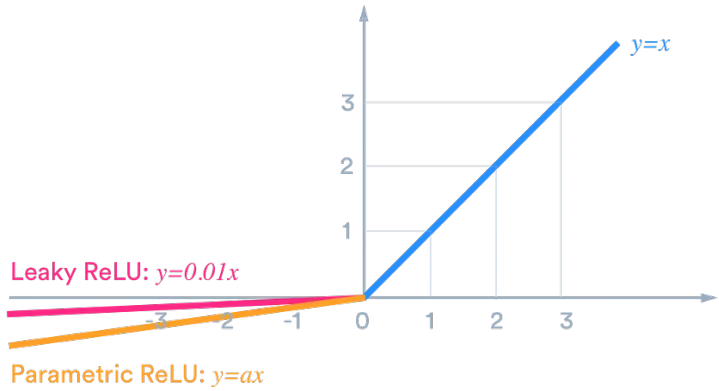


4.2.2-rasm. ReLU aktivlash funksiyasining grafik ko‘rinishi

ReLU neyron tarmoqlarda, ayniqsa CNNlarda eng ko‘p ishlatiladigan faollashtirish funksiyasidir. Uning quyidagi ikkita turi mavjud.

Leaky ReLU va Parametrik ReLU (PReLU)

Leaky ReLU umuman nol o‘rniga manfiy qiymatlar uchun kichik nishabga ega. Masalan, Leaky ReLU x < 0 bo‘lganda $y = 0{,}01 * x$ bo‘lishi mumkin.



4.2.3-rasm. Leaky ReLU va Parametrik ReLU (PReLU) aktivlash funksiyasining grafik ko‘rinishi

Parametrik ReLU (PReLU) Leaky ReLU ning bir turi bo‘lib, u 0,01 kabi oldindan belgilangan og‘ishga ega bo‘lish o‘rniga, uni neyron tarmoq uchun o‘zini aniqlash parametriga aylantiradi: *x<0* bo‘lganda *y=ax*.

Leaky ReLU

$$f(x) = \begin{cases} x & \text{if } x > 0, \\ 0.01x & \text{otherwise.} \end{cases} \qquad f'(x) = \begin{cases} 1 & \text{if } x > 0, \\ 0.01 & \text{otherwise.} \end{cases} \tag{4.2.2}$$

Parametrik ReLU (PReLU)

$$f(x) = \begin{cases} x & \text{if } x > 0, \\ a \cdot x & \text{otherwise.} \end{cases} \qquad f'(x) = \begin{cases} 1 & \text{if } x > 0, \\ a & \text{otherwise.} \end{cases} \quad (4.2.3)$$

$a \leq 1$ uchun bu

$$f(x) = \max(x, ax)$$

ga ekvivalent ekanligini unutmaslik kerak va shuning uchun "maxout" tarmoqlari bilan aloqasi bor.

Biroq, ReLU kamchiliklardan xoli emas. Sezilarli muammolardan biri bu "Dying ReLU" muammosi. Trening jarayonida ba'zi neyronlar faqat nolga teng bo'lgan holatda qolishi mumkin. Neyron bu holatga tushgandan so'ng, uning qiymatini tiklash imkoniyati mavjud emas va bu holat modelning imkoniyatlarini sezilarli ravishda kamaytiradi.

4.2.3. Softmax funksiyasi

Softmax aktivlash funksiyasi neyron tarmoqlarda ko'p turdagi ma'lumotlarni tasniflash uchun qo'llaniladi. Softmax funksiyasi berilgan sonlar to'plamidagi har bir elementlarni (0, 1) oraliqda yotuvchi qiymatlarni ifoda etuvchi sonlar to'plamiga aylantiradi. Masalan, tasavvur qilaylik, bizda sonlar to'plami mavjud:

$$Y=\begin{bmatrix} 2 \\ 1 \\ 0 \end{bmatrix} \quad (4.2.4)$$

Yuqoridagi to'plamni ma'lum bir element bilan bog'laydi. Asosiy, e-mail xabarlarni kategoriyalarga ajratish muammosini olsak, asosiy kategoriyaning 1-sinf elementi (qiymati 2), ijtimoiy tarmoqlar

kategoriyasining 2-sinf elementi (qiymati 1) va nihoyat, reklama kategoriyasining 3-sinf elementi sifatida (qiymati 0). Ushbu sonlar to‘plamiga softmax funksiyasini tadbiq qilganimizdan so‘ng natija sifatida biz quyidagi ehtimolliklar to‘plamini olamiz.

$$Y=\begin{bmatrix}2\\1\\0\end{bmatrix} \tag{4.2.5}$$

Yuqoridagi ifodada sonlar to‘plami ustida matematik amallarni bajarish ancha qulay. Sababi softmax funksiyasi tatbiq etilgan sonlar to‘plami ma’lum bir oraliqda yotadi (ya’ni[0,1]). Bu esa bizga ushbu to‘plamning elementlari qiymatlarini klasslar ehtimolligi sifatida qabul qilish imkonini beradi. Bundan tashqari, ma’lum bir qaror qabul qilish chegarasini tanlashda qulaylik yaratadi. Masalan, qaror qabul qilish chegarasi sifatida 0,5 qiymat olingan bo‘lsa, u holda ushbu qabul qilingan chegaradan yuqori qiymat elementni ushbu to‘plamning asosiy sinfi sifatida e’tirof etish mumkin.

Ushbu to‘plam 70% ehtimollik bilan asosiy kategoriyani, 20% ehtimollik bilan ijtimoiy tarmoqlar kategoriyasini va 10% ehtimollik bilan reklama kategoriyasini anglatadi. Qaror qabul qilish chegarasini (ya’ni 0.5) inobatga olib aytganda, ushbu to‘plam asosiy kategoriyasini anglatadi.

$$softmax(Z)_i = \frac{e^{z_i}}{\sum_{j=1}^{k} e^{z_i}} \tag{4.2.6}$$

Yuqoridagi formulada i=[1....K] hamda z=[z_i.....z_k] oraliqlarida joylashadi va K sinfi sonini, z_i (yoki z_i) esa oxirgi qatlamning

neyronlaridan chiquvchi qiymatlarini ifoda etadi (bunda $z=a_1*x_1+a_2*x_2+a_i*x_i$) ifodasining natijasidir.

4.2.4. Oldinga va ortga siljish algoritmi

Oldinga siljish algoritmi - neyron tarmoqdagi har bir qatlam neyronlarining chiqish qiymatlarini hisoblab, ushbu qiymatlarni keyingi qatlamga uzatish jarayoniga aytiladi. Hisob-kitoblar 2 - qatlamdan boshlanadi: 1 - qatlam - kirish qatlami, ya'ni kiruvchi o'zgaruvchilar bo'lgani uchun 1 - qatlamning qiymati bevosita 2 - qatlamga uzatiladi. Hisob-kitoblarni tezlashtirish uchun asosan vektorlangan hisob - kitoblardan foydalaniladi, ya'ni bir qadamning o'zida o'qitish to'plamining bir - nechta qatori birdaniga neyron tarmoqqa kiritiladi. Ammo odatda o'qitish to'plamlari katta hajmdagi ma'lumotlardan iborat bo'lganganligi sababli hisob - kitoblar juda katta xotira talab qilishi mumkin. Shuning uchun ma'lumotlar to'plami qatorlarini bir nechta bo'lakka bo'lib, neyron tarmoqqa uzatiladi. Ushbu bo'laklar **batch**, deb ataladi hamda giperparametr hisoblanib, neyron tarmoq amallarini bajarishdan oldin tanlanadi. Aytaylik, mashq to'plamida 1250 ta qator mavjud va biz **batch** hajmida (batch_size) 100 ni tanladik (ya'ni batch _size = 100). Bunda neyron tarmoqqa birinchi 100 ta qatorni (1-100- qatorlarni) kiritamiz va neyron tarmoqni mashq qildiramiz (o'qitish jarayoni). Keyin 100 talikni (101-200 - qatorlarni) neyron tarmoqqa kiritamiz va mashq jarayonini amalga oshiramiz. Ushbu sikl oxirida ma'lumotlar to'plamining oxirida 50 ta qatorini (1201-1250 -qatorlarni) neyron tarmoqqa kiritamiz va mashqni amalga

oshiramiz. Ushbu jarayon o‘z navbatida davrlar sonicha (ya’ni epochs) takrorlanadi. Shunda biz har bir davrda 13 ta **batches** yoki bo‘laklardan iborat bo‘lgan ichki siklga ega bo‘lamiz. Ma’lumotlar to‘plamini bo‘laklarga bo‘lib, ya’ni batchlar orqali neyronni o‘qitish quyidagi afzalliklarga ega.

Neyron tarmog‘i optimal parametrlarga tezroq ega bo‘ladi chunki har bir batch siklida neyron tarmoqning parametrlarini o‘zgartirish amali bajariladi. Albatta, bu batch_size parametri, ya’ni har bir bo‘lak soni **giperparametr** hisoblanadi va uning qiymati vaziyatdan kelib chiqib tanlanadi. Buning sababi qiymat sifatida juda kichik son tanlansa, neyron tarmoqning tezligini va aniqligini tushirib yuboradi.

Endi oldinga siljish algoritmiga qaytsak, birinchi navbatda yashirin qatlamning chiqish qiymatlari hisoblab olinadi.

$$c_1^2 := f(w_{1,1}^{(2)} \cdot x_1 + w_{1,2}^{(2)} \cdot x_2 + ... + b_1^{(2)})$$

$$c_2^2 := f(w_{2,1}^{(2)} \cdot x_1 + w_{2,2}^{(2)} \cdot x_2 + ... + b_2^{(2)}) \qquad (4.2.7)$$

$$\vdots$$

$$c_n^2 := f(w_{n,1}^{(2)} \cdot x_1 + w_{n,2}^{(2)} \cdot x_2 + ... + b_n^{(2)})$$

4.2.7-formula yashirin qatlamdagi hisoblashlarni amalga oshirishni ifodalaydi. *f(x)* – aktivlash funksiyasi bo‘lib, obyekt muammosidan kelib chiqib tanlanadi. Keyingi bosqichda 3-qatlamdagi neyronlarning chiqish qiymati hisoblanadi va bu oxirgi natija.

$$z_1^3 := w_{1,1}^{(3)} \cdot c_1^{(2)} + w_{1,2}^{(3)} \cdot c_2^{(2)} + w_{1,3}^{(3)} \cdot c_3^{(2)} + ... + b_1^{(2)})$$

$$z_2^3 := w_{2,1}^{(3)} \cdot c_1^{(2)} + w_{2,2}^{(3)} \cdot c_2^{(2)} + w_{2,3}^{(3)} \cdot c_3^{(2)} + \ldots\ldots + b_2^{(2)}) \tag{4.2.8}$$

$$\vdots$$

$$z_n^3 := w_{n,1}^{(3)} \cdot c_1^{(2)} + w_{n,2}^{(3)} \cdot c_2^{(2)} + w_{n,3}^{(3)} \cdot c_3^{(2)} + \ldots\ldots + b_n^{(2)})$$

Oxirgi qatlamda natija hisoblanadi. z_i^3-natijani hisoblash formulasi. Hisoblash jarayonlari nihoyasiga yetgach qiymat funksiyasi yordamida neyronlarning qanchalik to‘g‘ri natija berganini baholash talab etiladi. Qiymat funksiyasi neyron tarmoq qo‘llanilayotgan muammoga qarab tanlanadi. Masalan neyron tarmoq chiziqli regressiya muammosiga o‘xshab davomiy qiymatlarni bashorat qilsa, unda o‘rta kvadratik xatolik funksiyadan foydalanish mumkin.

$$J = \frac{1}{2N}\sum_{i=1}^{N}\left(f(x^{(i)}) - y^{(i)}\right)^2 \tag{4.2.9}$$

Bu yerda J – qiymat funksiyasi.

Agarda neyron tarmoqlari 2 ta sinfdan iborat bo‘lgan tasniflash muammmosiga tadbiq etilayotgan bo‘lsa, binar-logarifmik xatolik funksiyasini tadbiq etishimiz mumkin.

$$J = -\sum_{i=1}^{N}\left[y_i \cdot \log(f(x_i))\right] \tag{4.2.10}$$

Yuqoridagi formulada y_i- sinfning haqiqiy qiymati va $\log(f(x_i))$ - neyron tarmoqning oxirgi qatlamidagi chiqish qiymati.

Ortga siljish algoritmi – Neyron tarmoqlarning samaradorligini oshirish uchun qiymat funksiyasini minimallashtirish, ya’ni optimal koeffitsiyentlarini topishimiz kerak. Bunda biz ortga siljish

algoritmidan foydalanamiz. Ortga siljish algoritmining asosiy maqsadi – bu neyron tarmog‘idagi har bir koeffitsiyentning o‘zgarishi qiymat funksiyasiga qanday ta’sir qilishini topishdir. Ushbu ta’sirni topganimizdan so‘ng koeffitsiyentni qiymat funksiyasini kamaytiradigan tarzda yangilashimiz mumkin. Ushbu ta’sirni topish uchun avvaliga oldinga siljish algoritmini amalga oshirib, so‘ngra ortga siljish algoritmini hisoblab topiladi. Bunda harakat oldinga siljishga nisbatan teskari, ya’ni oxirgi qatlamdan boshlanadi. Ortga siljish algoritmi koeffitsiyentlarning qiymat funksiyasiga tasirini o‘rganadi, yani barcha qatlamlaridagi neyron koeffitsiyentlari uchun

$$\frac{\partial J}{\partial a_{i,j}^{(l)}} va \frac{\partial J}{\partial b_i^{(l)}} \tag{4.2.11}$$

(4.1.10) formula ko‘rinishdagi hisoblashlar amalga oshiriladi. Ushbu ifodalarni hisoblashdan oldin har bir qatlamdagi neyron uchun yordamchi parametr - δ_i^L ni hisoblab olishimiz kerak bo‘ladi. Ushbu parametr odatda neyron xatoligi deb ataladi.

$$\delta_i^L = \frac{\partial J}{\partial z_i^{(L)}} \tag{4.2.12}$$

4.2.12-formulada δ_i^L-qiymati sifatida chiqish qatlamdagi z_i^L o‘zgaruvchisining qiymati (J) funksiyasiga ta’siri hisoblanadi. Ushbu ta’sirni topish, albatta, oxirgi qatlamdagi chiqish funksiyasiga hamda foydalanilayotgan qiymat funksiyasiga bevosita bog‘liqdir.

Neyron tarmoqlarni faollashtirish funksiyalari, neyron og'irligi, ma'lumotlar oqimi va boshqalar asosida bir nechta turlarga bo'lish mumkin. Neyron tarmoqlarining turlari quyidagilardan iborat:

- Perceptron
- Recurrent Neural Networks
- Bidirectional Recurrent Neural Networks
- Convolutional Neural Networks

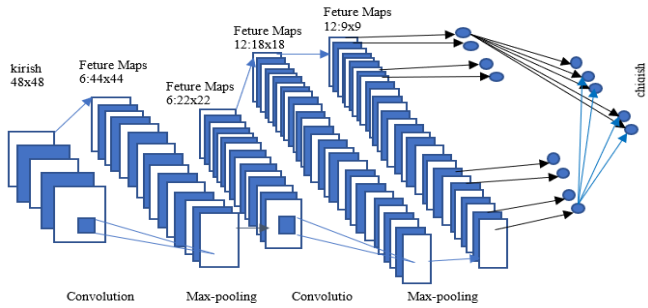
Nazorat savollari

1. Neyron tarmoq nima va uning afzaliklari va kamchiliklari nimada?
2. Neyron tarmoqlar qanday muammolarni hal qila oladi?
3. Neyron tarmoqda aktivlash funksiyalarining asosoiy vazifasi nimada?

4.2.5. CNN arxitekturasi va ishlash strukturasi

Convolutsion neyron tarmoqlari(CNN) - ko'p qatlamli neyron tarmoqlarining bir turi sanaladi. U kichik qadamlar bilan masshtabni va buzilishlarni o'zgartirish asosida tarmoqning invariantliligini ta'minlaydigan ikkita asosiy g'oyani o'zida birlashtiradi bular: lokal resyeptiv maydonlar(local receptive fields) va umumiy vazn (shared weights). CNN ikki o'lchovli ma'lumotlarni tanish uchun mo'ljallangan, shuning uchun ham neyron tarmoqning har bir qatlamida tekisliklarni xosil qiladi. Bu kiruvchi ma'lumotlarning topologiyasini saqlab qolish uchun amalga oshiradi. Kiritish qatlami bitta tekislikdan iborat bo'lib, uning o'lchami kirish ma'lumotlarining

o‘lchamiga to‘g‘ri keladi. Keyingi qatlamlar svyortkali hisoblanadi va belgilar kartasi deb nomlangan (feature maps) bir nechta neyronlar tekistliklaridan tashkil topgan. Convilation qatlamning har bir neyroni o‘zidan oldingi qatlamning lokal reseptivli maydoniga ulangan. Tarmoqning oxirgi ikkita qatlami amaliy jihatdan oddiy to‘g‘ri tarqalgan tarmoqni o‘zida aks ettiradi.



4.2.4- rasm. CNN arxitekturasi

CNN tarmoqlari ikki o‘lchamli ma’lumotlarga asoslanganligi sababli, ma’lumotlarni tasvir ko‘rinishiga (spektrogramma) keltiradi. CNN tarmoqlarining aksariyati tasvirdagi timsollarni tanishning samarali usuli sanaladi va ular spektrogrammadagi timsollarni tanish uchun ishlatilayotganligini ko‘rishimiz mumkin.

CNN ni quyidagi afzalliklari sababli tasvirlarni tanish tizimida samarali foydalanilmoqda.

1) To‘liq bog‘lanishli neyron tarmoqlari bilan solishtirganda o‘qitish tezligi yuqori va o‘qitish parametrlari soni kamligi.

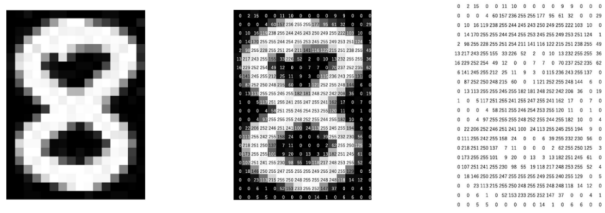
2) Grafik prosessorlarda (GPU) o'qitish algoritmini parallel amalga oshirish imkoniyatining mavjudligi.

3) Vizual tanib olish imkoniyatiga ega.

CNN arxitekturasi ko'p qatlamlardan tashkil topgan inson vizual tizimidagi neyronlarning tuzilishini taqlid qilishga harakat qiladi. Yuqoridagi 4.2.4-rasmda ko'rsatilganidek, odatiy CNN to'rtta asosiy qatlamning kombinatsiyasidan iborat:

- Konvolyutsion qatlamlar
- Rektifikatsiya qilingan chiziqli birlik (qisqacha ReLU)
- Birlashtiruvchi qatlamlar
- To'liq bog'langan qatlamlar

Konvolyutsiya qatlamining ishlash prinsipi. Tasvirlardan obyektning shakli va qirralarini aniqlashda Konvolyutsion neyron tarmoq piksel qiymatlarini solishtirish orqali amalga oshiradi. Quyidagi 4.2.5-rasmda bizga 8 raqamining tasviri va ushbu tasvirga to'liq mos keluvchi piksel qiymatlari keltirilgan. Demak, bu yerda shaklni aniqlashning oddiy usuli qo'shni piksel qiymatini solishtirishdan iborat.



4.2.5-rasm. CNN orqali rasmni tanib olish

Bu jarayonda biz qanday yo‘l tutishimiz kerak? Piksel bo‘ylab harakat qilishimiz va bu qiymatlarni solishtirishimiz kerakmi? Yo‘q! Ushbu ma’lumotni olish uchun tasvir filtr bilan (shuningdek, bu jarayonda qatnashayotgan filter tushunchasi matematik modelda “yadro” sifatida ham ataladi va shu nomda yuritiladi) konvolutsiya (svyortka) qilinadi.

Konvolyutsiya ko‘pincha matematik tarzda yulduzcha * belgisi bilan ifodalanadi. Agar bizda kiruvchi tasvir qiymati X sifatida ifodalangan bo‘lsa va f harfi bilan ifodalangan filtr (yadro) mavjud bo‘lsa, u holda ifoda quyidagicha bo‘ladi.

$$Z = X * f \tag{4.2.13}$$

Keling, biz oddiy misol yordamida konvolyutsiya jarayonini tushuntirib o‘tishga harakat qilamiz. Bizga 3 x 3 o‘lchamdagi rasm va 2 x 2 o‘lchamdagi filtr berilgan deb faraz qilaylik (4.2.6-rasm).

1	7	2
11	1	23
2	2	2

*

1	1
0	1

4.2.6-rasm. 3 x 3 o‘lchamdagi rasm va 2 x 2 o‘lchamdagi filtr

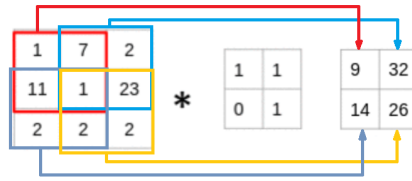
Bu jarayonning matematik amallar ketma-ketligini quyidagicha tushuntirish mumkin. Filtr tasvirlarning ma’lum katakchalarini o‘z ichiga olgan bo‘lakchalardan o‘tadi, mos katakchalardagi elementlar bo‘yicha ko‘paytirishlar amalga oshiriladi hamda qiymatlar umumlashtiriladi:

(1*1 + 7*1 + 11*0 + 1*1) = 9

(7*1+2*1 + 1*0 + 23*1) = 32

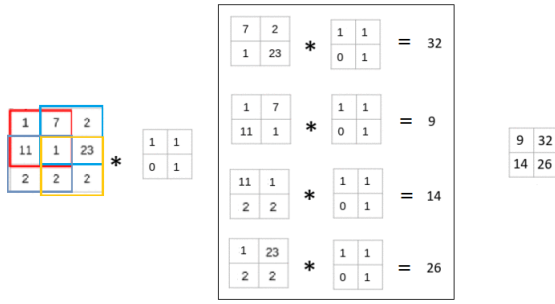
(11*1 + 1*1 + 2*0 + 2*1) = 14

(1*1 + 23*1 + 2*0 + 2*1) = 26



4.2.7-rasm. Feture Maps

Bu jarayonni yanada soddaroq tushuntirish uchun diqqat bilan ko‘rib chiqamiz. Bunda filtr bir vaqtning o‘zida tasvirning bir nechta kichik qismlarini ko‘rib chiqayotganini sezishimiz mumkin. Biz buni har biri filtr bilan birlashtirilgan kichikroq bo‘lakchalarga bo‘lingan bitta rasm sifatida ham tasavvur qilishimiz mumkin .



4.2.8-rasm. Feture Maps hisoblash jarayoni

Yuqoridagi misolda bizda kiruvchi qiymat matritsasi (3x3) va filtr matritsasi (2x2) o‘lchamli edi. Bu rasm va filtrning o‘lchamlari juda kichik bo‘lgani uchun, chiqish matritsasining o‘lchami ham (2x2) ekanligini tushunish oson. Ammo real hayotda CNN ni qo‘llash uchun bizga qo‘yiladigan masalalarda ancha murakkab kiruvchi qiymatlar, ya’ni, tasvirlarning matratsaviy o‘lchamlari juda katta bo‘ladi. Shu

o‘rinda bir savol tug‘iladi filtr o‘lchamlari va chiquvchi qiymat uchun natijaviy o‘lchamlarini qanday topishimiz mumkin. Buning uchun quyidagi oddiy formuladan foydalanishimiz mumkin:

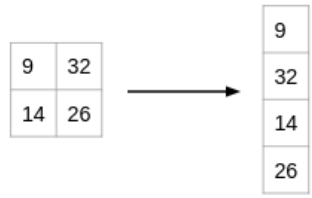
Rasm o‘lchami = (n, n)

Filtrning o‘lchami = (m, m)

Chiquvchi natija o‘lchami $((n - m + 1), (n - m + 1))$ bo‘ladi.

Konvolyutsion qatlam qanday ishlashini yaxshi tushunishimiz uchun keling, CNN arxitekturasining keyingi qismiga o‘tamiz.

To‘liq bog‘langan qatlam. Yuqorida bajarilgan matematik amallar yordamida konvolyutsiya qatlami ma’lumotlardan ba’zi kerakli xususiyatlarni ajratib olish imkonini berdi. Keyingi qadamda biz ushbu xususiyatlarga ega bo‘lgan ma’lumotlarni yakuniy natijalarni yaratadigan to‘liq bog‘langan qatlamga yuboramiz.



4.2.9-rasm. Birlik matritsa hosil qilish

Konvolyutsiya qatlamining natijaviy qiymati 2D (ikki o‘lchamli) matritsa edi. Ideal holda, biz har bir satr bitta kiruvchi qiymat tasvirini ko‘rsatishini istashimiz tabiiy. Biroq, to‘liq bog‘langan qatlam faqat 1D (bir o‘lchamli) ma’lumotlar bilan ishlaydi. Shu o‘rinda bizga yangi

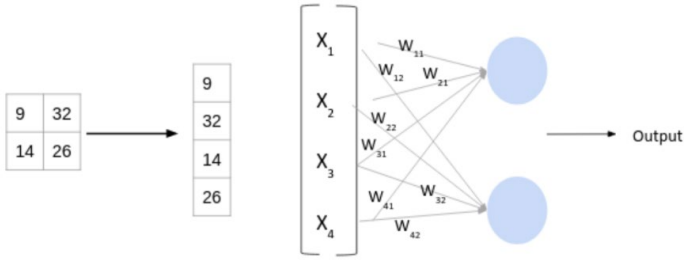
vazifa yuklanadi ya'ni, oldingi operatsiyadan olingan qiymatlarni birinchi navbatda biz 1D (bir o'lchamli) formatiga aylantirishimiz lozim bo'ladi.

Chiziqli transformatsiya. Ma'lumotlar 1D massivga aylantirilgach, u to'liq bog'langan qatlamga yuboriladi. Bu individual bog'lanishlarning barchasi tasvirni ifodalovchi alohida xususiyatlar sifatida qaraladi. To'liq bog'langan qatlam kiruvchi ma'lumotlar bo'yicha ikkita operatsiyani bajaradi. Chiziqli transformatsiya va chiziqli bo'lmagan transformatsiya.

Biz birinchi navbatda ushbu ma'lumotlarga chiziqli o'zgartirishni amalga oshiramiz. Chiziqli transformatsiya tenglamasi:

$$Z = W^T * X + b \qquad (4.2.14)$$

Bu yerda: X – kiruvchi qiymat, W - og'irlik (gradiyent), b – o'zgarmas kattalik (bias deb ataladi). Shu o'rinda e'tibor berishimiz lozim bo'lgan jihat bor, bu yerda w (tasodifiy ishga tushirilgan) sonlar matritsasi bo'ladi.

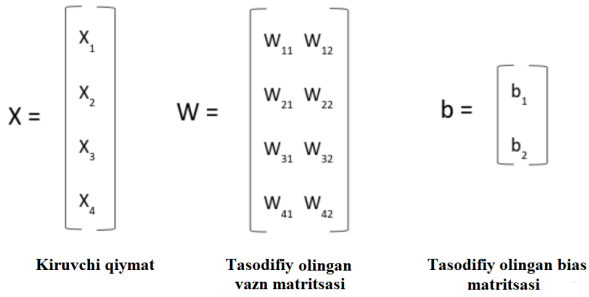


4.2.10-rasm. Neyron qatlamini hisoblash(Full connekt)

Biz buning uchun avvalo matritsaning o‘lchamini hisobga olgan holda (m, n) o‘lchamli matritsa deb qarasak bo‘ladi. Bu yerda - m ushbu qatlam uchun xususiyatlar yoki kiruvchi qiymatlar soniga teng bo‘ladi. Bu qiymatlarning sonini esa biz konvolyutsiya qatlamidan 4 ta xususiyatga ega bo‘lgan natijamiz hosil bo‘lgani uchun bu yerda $m = 4$ bo‘ladi.

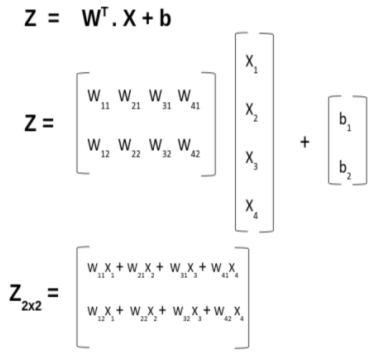
n – qiymati esa qatlamdagi neyronlar soniga bog‘liq bo‘lgan holda aniqlanadi. Masalan, agar bizda ikkita neyron bo‘lsa, unda vazn (gradiyent) matritsasi o‘lchami $(4, 2)$ bo‘ladi:

Ushbu bog‘liqlik orqali qurilgan neyron tarmoqning strukturaviy ko‘rinishi misol tariqasida 4.2.11-rasmda ko‘rsatib o‘tilgan.



4.2.11-rasm. Neyron tarmoq og‘irliklarini hisoblash

Og‘irlik va bias matritsasini aniqlagandan so‘ng, keyingi vazifamiz vazn matritsasini transportatsiya qilgan holda (2) – formulaga qo‘yamiz, natijani chiziqli o‘zgartirish tenglamasiga qo‘ygach quyidagi 9-rasmdagidek ko‘rinish hosil bo‘ladi:



4.2.12-rasm. Neyron tarmoq og‘irliklarini transformatsiya qilish

Oldinga tarqalish jarayonida bitta yakuniy qadam bor bu ham bo‘lsa chiziqli bo‘lmagan o‘zgartirishlar. Keling, chiziqli bo‘lmagan transformatsiya tushunchasini va uning oldinga siljish jarayoniga batafsil to‘xtalib o‘tamiz.

Oldinga tarqalishning matematik modeli. Bu bo‘limda biz yuqorida keltirib o‘tilgan tushuncha va ifodalardan foydalangan holda jarayonning to‘liq matematik modelini ishga tushiramiz hamda aniq ma’lumotlar asosida amaliy jihatdan qo‘llab ko‘ramiz. Buning uchun quyidagi ketma-ketlikda ishni boshlaymiz.

1-qadam: Kiruvchiq qiymat ma’lumotlarini (rasmlarini) o‘zgaruvchiga yuklab olamiz (aytaylik, X ga)

2-qadam: Filtr matritsasini aniqlaymiz (tasodifiy ishga tushiriladi random orqali). Tasvirlar filtr bilan birlashtiriladi (1).

$$Z_1 = X * f$$

3-qadam: hosil bo‘lan natijaga Sigmoid faollashtirish funksiyasini qo‘llaymiz (3).

$$A = sigmoid(Z_1)$$

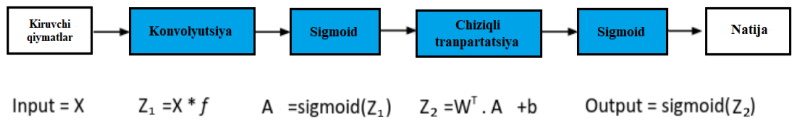
4-qadam: Og‘irlik (tasodifiy ishga tushiriladigan gradiyent) va bias matritsasini aniqlaymiz. Hosil bo‘lgan qiymatlarga chiziqli o‘zgartirishni qo‘llaymiz.

$$Z_2 = W^T * A + b$$

5-qadam: Ma’lumotlarga sigmoid funksiyasini qo‘llaymiz va yakuniy natijani hosil qilamiz.

$$B = sigmoid(Z_2)$$

CNN modelining yakuniy arxitekturaviy ko‘rinishi quyidagicha bo‘ladi



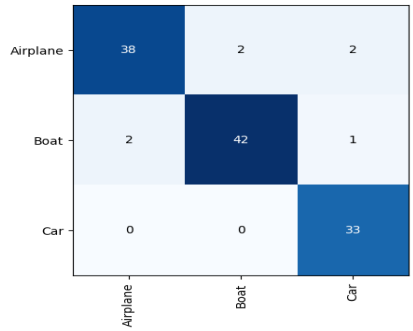
4.2.13-rasm. CNN modelining yakuniy arxitekturaviy ko‘rinishi

CNN modeli bu qiymatlarni o‘quv jarayonida tasodifiy ishga tushiriladigan va o‘rganiladigan parametrlar sifatida ko‘radi va yuqoridagi ketma ketlikda hisoblashlarni amalga oshiradi.

Misol: Bu misolda biz CNN algoritmidan foydalanib avtomobil, samolyot va kemalarni sinflashtiruvchi model yaratdik. Bu modelni yaratishda Google kompaniyasining ma’lumotlaridan foydalandik.

Har bir sinfni o‘qitish uchun 200 ta dan ortiq rasmlardan foydalanildi. Jami 600 ta rasmdan foydalanildi.

```
from fastai.vision.all import *
from ipywidgets import widgets
#path
path = Path('train')
# fls = get_image_files(path)
# failed = verify_images(path)
transports = DataBlock(
    blocks=(ImageBlock, CategoryBlock),
    get_items = get_image_files,
    splitter = RandomSplitter(valid_pct=0.2,seed=42),
    get_y = parent_label,
    item_tfms=Resize(224))
#Dataloader yaratamiz
dls = transports.dataloaders(path)
#datasetni tekshirish
# dls.train.show_batch(max_n=32,nrows=4)
#O‘qitish
learn = cnn_learner(dls,resnet34, metrics=accuracy)
learn.fine_tune(4)
interp = ClassificationInterpretation.from_learner(learn)
interp.plot_confusion_matrix()
```



4.2.13-rasm. CNN modelining aniqligini chalkashlik matritsasi orqali baholash

Bu chalkashlik matritsasidan ko‘rishimiz mumkinki, samolyot sinfiga tegishli 42 ta test qiymatimizdan 38 tasini to‘g‘ri aniqladi, 2 tasini kema va 2 tasini avtomobil deb bashorat qildi. Hamda 45 ta kema sinfiga tegishli test qiymatlarimizni 42 tasini to‘g‘ri aniqladi, 2 tasini samolyot va 1 tasini avtomobil deb bashorat qildi. Avtomobil sinfiga tegishli 33 ta test qiymatdan hammasini to‘g‘ri aniqladi.

4.2.6. RNN arxitekturasi va ishlash strukturasi

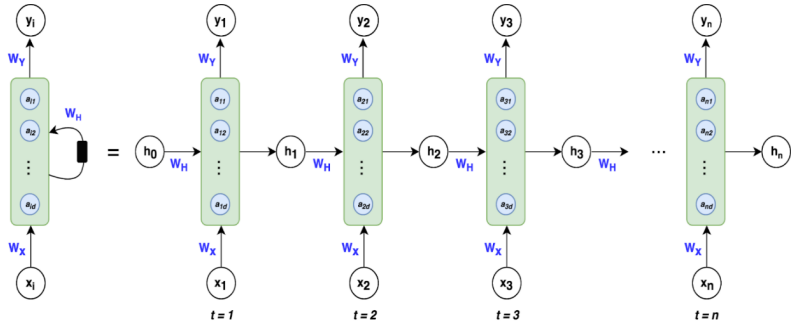
Recurrent neyron tarmoq (RNN). To‘g‘ridan to‘g‘ri bog‘langan ananaviy neyron tarmoqlari, masalan KQP lar atrofidagi konteksdan mustaqil, o‘rnatilgan o‘lchamdagi kirishlarini hisobga oladigan statik klassifikator sifatida qaraladi. Bir yoki bir nechta yashirin qatlamlar bilan o‘zaro bog‘langan RNN bashoratlash uchun yanada samarali ekanligini ko‘rish mumkin. RNN teskari aloqa vaznini sozlash orqali o‘tgan hodisalarni modellashtirishni o‘rganishi mumkin. Bu esa unga avvalgi konteksdan foydalanish imkonini beradi. Berilgan kiruvchi

ketma–ketlikdagi $x = (x_1, \dots, x_y)$ uchun standart rekurrent tarmoq t=1, ...,T gacha bo'lgan iteratsiya asosida yashirin vektor ketma-ketlik $h = h_1, \dots, h_T$ va chiquvchi vektor ketma-ketlik $y = (y_1, \dots, y_T)$larni hisoblaydi. Bunda h_1 va y_1 quyidagicha ifodalanadi:

$$h_i = \delta(w_i \cdot x_i + w_i \cdot h_{i-1}, \dots, b_i) \quad (4.2.15)$$

$$y_i = softmaxs(w_{ky} \cdot h_i, \dots, b_i) \quad (4.2.16)$$

Bunda w-vazn matritsasi, b -bias vektor.



4.2.14-rasm. Recurrent neyron tarmoq

Recurrent neyron tarmoq RNN ning bir nechta amalga oshirish variantlari mavjud va bular:

Ikki tomonga yo'naltirilgan recurrent tarmoq (Bidirectional RNN - BRNN) ;

Uzun, qisqa muddatli xotiraga ega tarmoq(Long Short Term Memory networks LSTM). LSTM ga asoslangan ikki tomonga yo'naltirilgan rekkurent tarmoq (Bidirectional LSTM based RNN - BLSTM). Ushbu RNN tarmog'i va uning modifikatsiyalangan turlari

boshqa neyron tarmoq algoritmlariga nisbatan quyidagi afzalliklarga va kamchiliklarga ega.

Yutuqlari:

- RNN model xar bir ma’lumotni vaqt davomida yodda saqlab qolish uchun modellashtiriladi, bu har qanday vaqtli qatorlarni bashoratlash uchun juda samarali. Hatto kiruvchi o‘lcham katta bo‘lganda ham model o‘lchami oshmaydi;

-RNN ixtiyoriy kiruvchi ma’lumotlarga ishlov berish uchun o‘zining ichki xotirasidan foydalanishi mumkin.

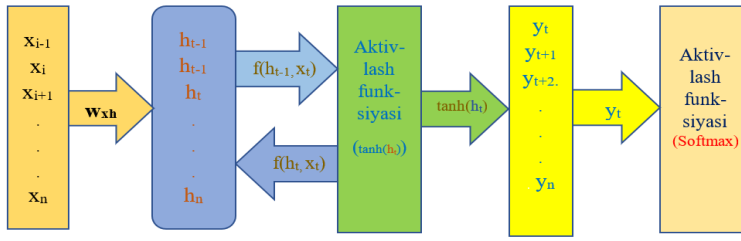
Kamchiliklari :

- Uning takrorlanuvchi tabiati hisobiga hisoblash sekin amalga oshiriladi.

- RNN modelini o‘qitish murakkab sanaladi.

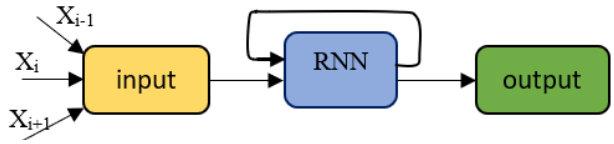
- RNN ixtiyoriy kiruvchi ma’lumotlarga ishlov berish uchun o‘zining ichki xotirasidan foydalanish mumkin.

RNN lar o‘zgaruvchan hodisalar ketma-ketligini yoki obyektlardan olingan ma’lumotlarni qayta ishlash uchun ishlatiladi. Ularning arxitekturasi quyidagi ko‘rinishda shakllanadi.



4.2.15-rasm. Recurrent neyron tarmoq arxitekturasi

Yuqoridagi arxitekturada ko‘k rangdagi RNN ning yadro bloki hisoblanadi. Uning kirish vektoriga va oldingi holatiga takrorlanish algoritmi qo‘llaniladi. Bunday holda, birinchi kiruvchi qiymatdan (x_{i-1}) oldin hech qanday qiymat mavjud emas. Shuning uchun ikkinchi kirish x_i, qiymatidan takrorlanish algoritmini qo‘llashni boshlaymiz. Bular kirishning turli vaqt bosqichlari deb nomlanadi. Shunday qilib, agar t vaqtida kirish x_i bo‘lsa, t-1 vaqtida kirish x_{i-1} ga teng bo‘ladi. Takrorlanish qoidasi x_i va x_{i-1} uchun ham qo‘llanilib yangi holatga ega bo‘ladi.



4.2.16- rasm. Recurrent neyronning yadro qatlami arxitekturasi

Yadro qatlamidagi matematik modelni hisoblash jarayoni quyidagi ko‘rinishda amalga oshiriladi.

$$h_t = f(h_{t-1}, x_t) \qquad (4.2.17)$$

h_t - hozirgi holat, h_{t-1} - oldingi holat, x_t-kirish holati. Endi bizda kirishning o‘rniga oldingi kirish holati mavjud, chunki kirish neyroni avvalgi kiritishimizda o‘zgarishlarni qabul qila oladi. Shunday qilib, har bir keyingi kirish vaqt bosqichi deb ataladi. Bunday holda, bizda tarmoqqa berilishi kerak bo‘lgan uchta kirish mavjud, takrorlanish paytida har bir vaqt bosqichida tarmoqqa bir xil funksiya va bir xil

og‘irliklar qo‘llaniladi. Keyingi holatda aktivlashtirish funksiyasining qo‘llanilishi ko‘rib chiqiladi. Bunda geperpologik tangens funksiyasidan foydalaniladi.

$$h_t = \tanh(W_{hh} \cdot h_{t-1} + W_{xh} \cdot x_t) \quad (4.2.18)$$

W_{hh}- takrorlanuvchi neyrondagi og‘irlik, W_{xh}-neyron kirish qiymatidagi og‘irlik. Har bir siklda takroriy neyron tarmoq ma’lum bir qiymatga erishadi. y_i chiqish qiymatini hisoblash uchun quyidagi formuladan foydalanamiz.

$$y_t = W_{hy} \cdot h_t \quad (4.2.19)$$

y_t - chiqish, W_{hy} – chiqish og‘irligi. Kirishning birinchi qiymati, x_t tarmoqqa beriladi. Keyin joriy kirish va oldingi holat kombinatsiyasidan foydalanib, uning joriy holati hisoblanadi, ya’ni h_t hisoblab chiqiladi. Barcha holat bosqichlari bajarilgandan so‘ng, yakuniy holat y_t chiqishini hisoblash amalga oshiriladi. Keyin chiqish natijalarini softmax aktivlash funksiyasi yordamida aniqligi tekshiriladi.

$$\delta(y)_t = \frac{e^{y_t}}{\sum_{t=0}^{N} e^{y_t}} \quad (4.2.20)$$

$\delta(y)_t$-softmax aktivlash funksiyasi. Softmax funksiyasida hisoblangan natijalarning xatoligini baholash uchun Cross entropy funksiyasidan foydalaniladi.

$$L_t = -\frac{1}{N}\left[\sum_{t=0}^{N}[\bar{x}_t \cdot \log(\delta_t) + (1-\bar{x}_t)\cdot \log(1-\delta_t)]\right] \qquad (4.2.21)$$

L_t-Cross entropy. Ushbu funksiya, ma'lumotlar tizimining ishonchliligi va aniqliligi haqida ma'lumot beradi. Cross entropy tarmoqning qatlamlaridagi hisoblash natijalarini haqiqiy chiqish bilan taqqoslaydi va buning natijasida xatolikni baholaydi. Agar xatolik yuqori bo'lsa xatolikni kamaytirish va og'irliklarni yangilash uchun qiymat yana tarmoqqa qaytariladi (Backpropagation). Orqaga tarqalishni tushunish va vizualizatsiya qilish uchun, tarmoqni har doim bosqichma-bosqich ishlash jarayonini tushunish zarur. RNN da har bir vaqt bosqichida chiqishga ega bo'lishimiz yoki bo'lmasligimiz mumkin. Yo'nalish bo'yicha tarqalish holatida kirishlar har bir vaqt bosqichida kirib boradi va oldinga siljiydi.

Yuqoridagi ketma-ketliklardan foydalanib ma'lumotlarni qayta ishlash va tiklashni amalga oshirib ko'ramiz. Buning uchun birinchi navbatda o'quv tanlanmani sozlash amalga oshiriladi.

4.1-jadval. Kiruvchi ma'lumotlarni birlik matritsa ko'rinishiga keltirib olish

x_t-signal qiymati	1	2	3	4	5	6	7	8
	0.381	0	0	0	0	0	0	0
	0	0.428	0	0	0	0	0	0
Birlik	0	0	0.341	0	0	0	0	0
matritsa	0	0	0	0.432	0	0	0	0
	0	0	0	0	0.468	0	0	0
	0	0	0	0	0	0.456	0	0
	0	0	0	0	0	0	0.443	0
	0	0	0	0	0	0		0.525

O‘quv tanlanma sifatida eksperimental signalning 8 ta qiymati olingan. Bu jadvaldan foydalanib hisoblashlarni amalga oshiramiz. Keyingi bosqichda W_{xh}- neyron kirish qiymatidagi og‘irliklarning qiymatini hosil qilamiz.

4.2-jadval. Neyron kirish qiymatidagi og‘irliklar

W_{xh}			
0.297023	0.865608	0.623425	0.426832
0.890287	0.891523	0.712345	0.188837
0.567523	0.082246	0.655678	0.511448

Yuqoridagi 4.2.17-formula yordamida hisoblash amalga oshiriladi. Keyingi bosqichda W_{hh}- takrorlanuvchi neyrondagi og‘irlikni hosil qilamiz.

4.3-jadval. Takrorlanuvchi qatlamdagi og‘irlik qiymati

$W_{hh} + b_t$	
W_{hh}	b_t
0.527023	0.575608
0.627021	0.475602
0.727022	0.675703
0.587024	0.585651

Jadvaldagi qiymatlardan foydalanib 4.2.18-formulani hisoblashni amalga oshiramiz. Buning uchun esa chiqish og‘irliklarini hosil qilamiz.

4.4-jadval. Chiqish qatlamdagi og‘irlik qiymati

W_{hy}		
0.377023	0.986561	0.823425
0.377023	0.986561	0.823425
0.390287	0.2891523	0.6712345
0.667523	0.092246	0.355678
0.912663	0.32581642	0.144631

Bu jadvaldan foydalanib 4.2.20-formula qiymatlarini hisoblab chiqamiz. Hisoblash natijalarining qiymatlarini 4.2.21-formula yordamida eng katta qiymatini aniqlab olamiz. Aniqlash xatoligini esa Cross entropy yordamida baholashni amalga oshiramiz.

4.5-jadval. Xatoliklarni baholash

x_i	$\delta(y)_t$	L_t
0.381234	0.498748	0.69
0.428452	0.194682	1.66
0.341234	0.162429	1.83
0.432123	0.223141	1.51

Jadvaldagi natijalardan ko‘rinib turibdiki kiruvchi ma’lumot qiymatini boshqalariga nisbatan aniqroq topdi. Ammo xatolik ancha yuqori, bu xatoliklarni kamaytirish uchun backpropagation amalga oshiriladi.

4.2-bo‘lim bo‘yicha amaliy qism

4.2-bo‘lim bo‘yicha amaliy qism quyidagi mavzular bo‘yicha keltirilgan. Bunda keltirilgan havola yoki QR kodni skaner qilish orqali GitHubga murojat qilib dasturiy kodidan foydalanish imkoniyatiga ega bo‘lasiz.

https://github.com/TUITAI/

4.3. TABIIY TILGA QAYTA ISHLOV BERISH

Sun’iy intellekt sohasi bo‘lgan NLP tabiiy tildan foydalangan holda kompyuterlar va odamlar o‘rtasidagi o‘zaro aloqalarni bog‘lashga yordam beradigan algoritmlar to‘plamidir. NLP inson va kompyuterning o‘zaro ta’siri bilan bog‘liq bo‘lgan sohadir. NLP usullari ko‘plab turli xil ilovalarda qo‘llaniladi, masalan, avtomatik umumlashtirish, mashina tarjimasi, savollarga javob berish, ma’lumot qidirish, matn tasnifi, matn klasteri, hissiyotlarni tahlil qilish, dialog tizimlari va boshqalar.

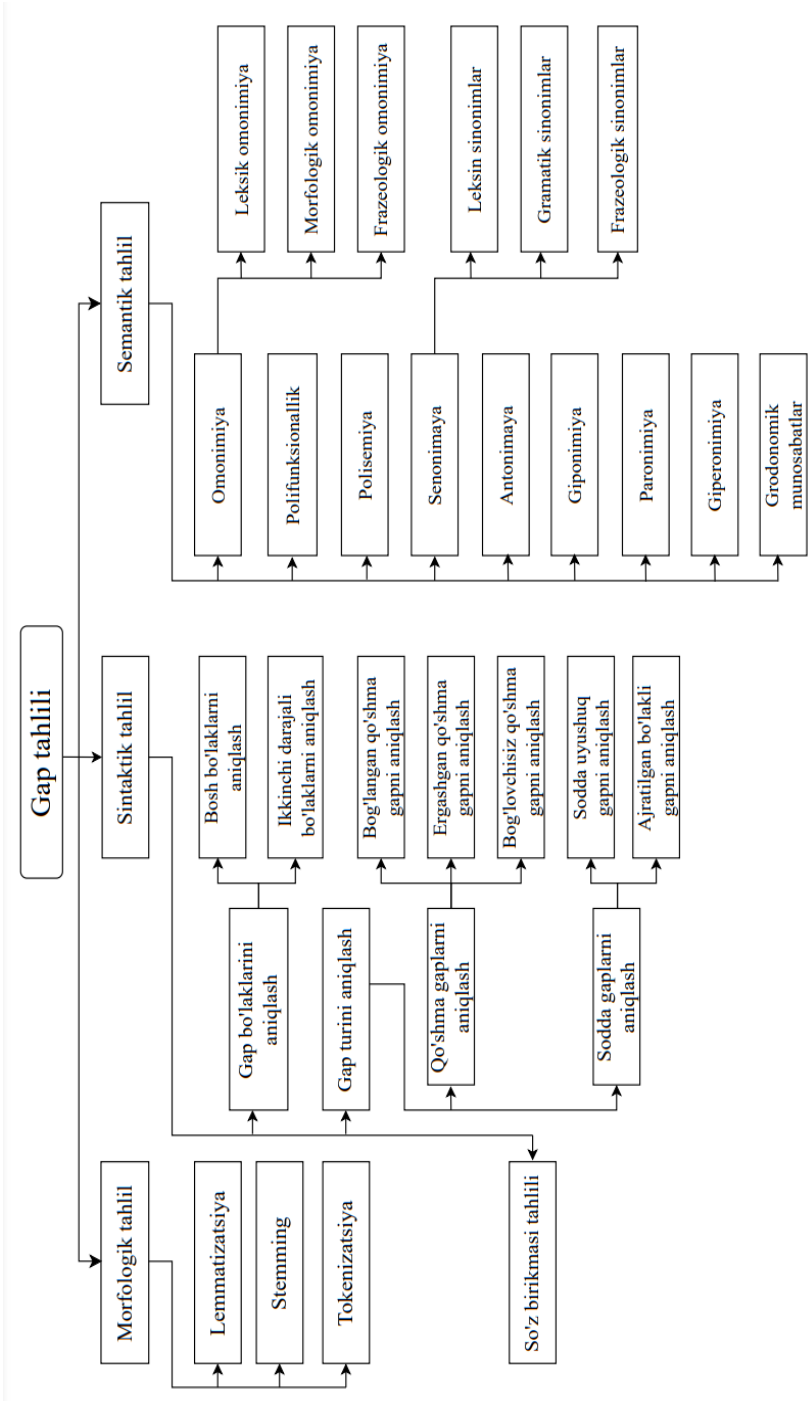
Tabiiy tilga avtomatik ishlov berish muammosi keyingi yarim asrdan ortiq davr mobaynida dolzarb bo‘lib qolmoqda. Muammoning murakkabligi va aniq g‘oyaning yo‘qligi uni hal qilish yo‘llarining qiyinligini bildiradi. Lingvistik analizatorlar esa gapga avtomatik ishlov berish vositalari sifatida alohida ahamiyat kasb etadi. Lingvistik analizatorni quyidagi turlarga bo‘lib o‘rganish mumkin:

1. Morfologik analizator.

2. Sintaktik analizator.

3. Semantik analizator.

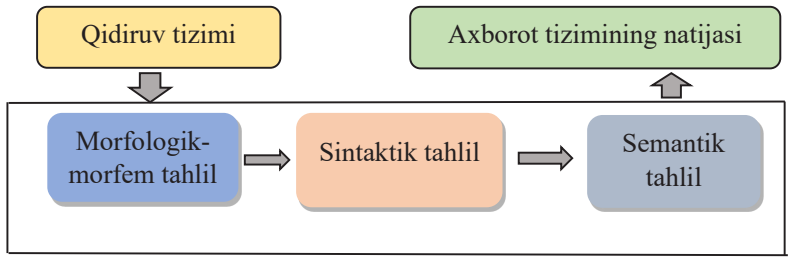
Gap tahlili *lemmatizator, morfoanalizator, stemmer, parser va orfokorrektorlar* kabi analizatorlarni o‘z ichiga oladi. Bu analizatorlarning har biri o‘zining ishlash prinsipi, maqsadi, vazifasiga ega. Umuman olganda, har bir analizator o‘z maqsadiga erishishda qator qadamlarni bosib o‘tishi talab qilinadi.



4.3.1-rasm. Gap tahlili

223

Tabiiy tillarga ishov berish jarayonida morfologik, sintaktik va semantik tahlil ketma-ketlik bilan amalga oshiriladi. Kirish tilidagi matnlarni semantik tahlil qiluvchi tizimning natijasi morfologik va sintaktik tahlilga asoslanadi. Bunday tizimlarning tahlil jarayonini 4.3.2-rasmda ko‘rishimiz mumkin:



4.3.2 -rasm. Tabiiy tillarga ishlov berish

Xulosa o‘rnida aytish mumkinki, avtomatik semantik tahlil va qidiruv kompyuter lingvistikasining dolzarb hamda murakkab vazifalaridan biri sifatida baholanadi.

4.3.1. Morfologik tahlil

NLP ning birinchi bosqichi morfologik tahlil hisoblanadi. Bu bosqich dastlabki ma’lumotlarni belgilar oqimi sifatida qabul qiladi va uni mazmunli leksemalarga aylantiradi. U butun matnni tokenlarga ajratib olish bosqichlarini amalga oshiradi. Shuningdek, NLP algoritmlari so‘zlarni so‘z turkumlariga ajratadi va leksemalarni morfemalarga ajratadi.

Tabiiy tilga ishlov berish sohasida morfologik (leksik) tahlil matndagi so‘z va iboralarning lemmasini va o‘zagini aniqlash jarayonidir. Buni turli usullar, jumladan, so‘zlar qo‘llanilayotgan kontekstni ko‘rib

chiqish, gaplar tuzilishini tekshirish va murakkab algoritmlardan foydalanish orqali amalga oshirish mumkin. Leksik tahlilda so‘zlarning tuzilishi tahlil qilinadi. Asosan so‘zning lemmasiga, ildiziga, qo‘shimchalarga va so‘z turkumiga qaraladi. Biz ushbu so‘zning kontekstda qanday ishlatilishini tahlil qilib chiqamiz. Bu so‘zning ma’nosini tushunishimizga yordam beradi. Leksik tahlil tilni birliklarga yoki leksemalarga - paragraflar, jumlalar, iboralar va so‘zlarga ajratadi. Morfemalarning 2 turi mavjud:

- **Erkin morfemalar** mustaqil ravishda so‘z vazifasini bajaradi (“mushuk” va “uy” kabi).
- **Bog‘langan morfemalar** kattaroq so‘zlarni hosil qiladi. “Tasavvur qilib bo‘lmaydigan” so‘zida “ - emas” (inkor kontekstni bildiruvchi bog‘langan morfema), “tasavvur” (butun so‘zning erkin morfema ildizi) va “mavjud” (o‘zak morfema qobiliyatini bildiruvchi bog‘langan morfema) morfemalari mavjud.

Morfologik tahlil tabiiy tilni qayta ishlashning asosiy jihati bo‘lib, so‘zlarning tuzilishi, shakllanishi va o‘zgarishini o‘rganishni o‘z ichiga oladi. U stemming, lemmatization va nutq qismlarini teglash kabi jarayonlarni o‘z ichiga oladi. Bu jarayonlar so‘zlarni o‘z ildiz shakllariga ajratish va gaplardagi grammatik rolini tushunishga yordam beradi. Pythonda NLTK, spaCy va TextBlob kabi bir nechta kutubxonalar yordamida morfologik tahlil qilish uchun foydalaniladi. Bu yerda biz amaliy kod misollari bilan ushbu kutubxonalar yordamida morfologik tahlilni qanday amalga oshirishni ko‘rib chiqamiz.

1. Stemming va lemmatizatsiya. NLTK kutubxonasi yordamida tahlil qilish. O‘zaklash - so‘zni asos yoki ildiz shakliga keltirish jarayoni. NLTK kutubxonasi Porter Stemmer kabi bir nechta stemmerlarni taqdim etadi.

Misol. Matndagi so‘zlarni o‘zagini bo‘laklarga ajratish.

```
import nltk
from nltk.stem import PorterStemmer
# Kerakli resurslarni yuklab olish
nltk.download('punkt')
# Stemmerni ishga tushirish
stemmer = PorterStemmer()
# Misol uchun matn
text = "running runners ran easily"
# Tokenize (matni belgilash)
tokens = nltk.word_tokenize(text)
# Har bir so‘zni asos qilib olish
stems = [stemmer.stem(token) for token in tokens]
print("Original Text:", text)
print("Stemmed Text:", stems)
Natija: running; runners; ran; easily;
```

2. NLTK kutibxonasi yordamida lemmatizatsiyani amalga oshirish. Lemmatizatsiya so‘zlarni kanonik shakl bo‘lgan lemmasiga qisqartiradi. Bu kontekstni ko‘rib chiqish uchun quydagi ketma-ketlik amalga oshiriladi.

```
from nltk.stem import WordNetLemmatizer
# Kerakli resurslarni yuklab olish
```

```
nltk.download('wordnet')
nltk.download('omw-1.4')
# Lemmatizatorni ishga tushirish
lemmatizer = WordNetLemmatizer()
# Har bir so‘zni lemmatizatsiya qilish
lemmas = [lemmatizer.lemmatize(token)
for token in tokens]
print("Lemmatized Text:", lemmas)
Natija: lem mas;
```

3. Nutq qismini teglash. Nutq qismi (POS) teglari har bir so‘zga ot, fe’l va sifatlar kabi nutq qismlarga ajratadi.

```
#NLTK bilan POS teglari
# Kerakli resurslarni yuklab olish
nltk.download('averaged_perceptron_tagger')
# POS teglarini amalga oshirish
pos_tags = nltk.pos_tag(tokens)
print("POS Tags:", pos_tags)
Natija: pos _tags;
```

4. Morfologik tahlil uchun spaCy dan foydalanish. spaCy-samarali morfologik tahlilni o‘z ichiga olgan ilg‘or NLP funksiyalarini taklif qiluvchi kuchli kutubxona hisoblanadi.

SpaCy bilan lemmatizatsiya va POS teglari. spaCy to‘g‘ridan to‘g‘ri stemmingni ta’minlamaydi, lekin lemmatizatsiya va POS belgilarida ustunlikni beradi.

```
import spacy

# SpaCy modelini yuklash

nlp = spacy.load('en_core_web_sm')

# Matnni qayta ishlash

doc = nlp(text)

# Lemmalar va POS teglarini ajratib olish

lemmas = [token.lemma_ for token in doc]

pos_tags = [(token.text, token.pos_)

for token in doc]

print("Lemmatized Text:", lemmas)

print("POS Tags:", pos_tags)
```

SpaCy bilan rivojlangan morfologik xususiyatlar. spaCy shuningdek, aspekt kabi batafsil morfologik xususiyatlarni taqdim etadi.

```
# Morfologik xususiyatlarni ajratib olish

morphs = [token.morph for token in doc]

for token in doc:

    print(f"Token: {token.text}, Morph: {token.morph}")
```

Oddiy morfologik tahlil uchun TextBlob dan foydalanish. TextBlob - bu NLTK-ni qamrab oluvchi va foydalanish uchun qulay NLP vositalarini taqdim etadigan kutubxona.

```
# TextBlob obyektini yaratish
from textblob import TextBlob
blob = TextBlob(text)
# So‘zlar, lemmalar va POS teglarini ajratib olish
words = blob.words
lemmas = [word.lemmatize() for word in words]
pos_tags = blob.tags
print("Words:", words)
print("Lemmatized Text:", lemmas)
print("POS Tags:", pos_tags)
Natija: words; lemmas; pos_tags;
```

Morfologik tahlilda matn ichidagi so‘zlarning tuzilishi va ma’nosini tushunish juda muhim hisoblanadi. NLTK, spaCy va TextBlob kabi kutubxonalardan foydalanib, biz turli xil morfologik tahlil vazifalarini bajarishimiz mumkin.

Stemming va Lemmatization:

So‘zlarni ildiz yoki kanonik shakllariga qisqartirish;

Gaplardagi so‘zlarning grammatik ma’nosini aniqlash;

Kengaytirilgan morfologik xususiyatlar;

So‘zlar haqida batafsil grammatik ma’lumotlarni ajratib olish.

Ushbu vositalar matnni oldindan qayta ishlash, his-tuyg‘uli gaplarni tahlil qilish, ma’lumot olish uchun zarurdir. Taqdim etilgan misollardan foydalanib, siz morfologik tahlilni boshlashingiz va har bir kutubxonaning imkoniyatlarini o‘rganishingiz mumkin.

4.3.2. Sintaktik tahlil

Tabiiy tilga ishlov berish sohasida sintaktik tahlil qismiga kirishdan oldin, sintaktik tuzilmalarning turlarini tushunishimiz kerak. Sintaktik tuzilmalar so‘z birikmalari, bo‘laklar, o‘rindoshlik munosabatlari, tobelik munosabatlari kabi bir qancha elementlardan iborat.

Sintaktik tuzilishdagi so‘z birikmalari. So‘z birikmalari jumla ichida bir komponent sifatida birgalikda faoliyat yurituvchi so‘z turkumini bildiradi. Umumiy atamalar fe’l so‘z birikmalari bilan bir qatorda ot so‘z birikmalarini ham o‘z ichiga oladi.

Sintaktik tuzilmalarda tanlov munosabatlari. Tanlov grammatikasi jumlalarni fe’l va ot iboralari kabi tarkibiy qismlarga ajratish uchun xizmat qiladi. Bu bo‘limning vazifasi - gaplarning sintaktik tuzilishiga oydinlik kiritadigan tarkibiy qismlarga ajratib olish.

Sintaktik tuzilmalarda tobelik munosabatlari. Tobelik grammatikasida gap so‘zlari o‘rtasida bog‘lanish mavjud. Bog‘lanish bog‘liqlik munosabati bo‘lib, gaplardagi so‘zlarning bir-biriga qanday bog‘liqligini ochib beradi. Sintaktik tahlilda tobelik munosabatlarining

ahamiyati juda katta. Buning sababi shundaki, ular so‘zlar orasidagi sintaktik birikmalarni ajratib olishda yordam beradi.

Sintaktik tahlilning asosiy usullari. NLPdagi sintaktik tahlil tilda grammatik tuzilmani ajratish uchun turli xil yondashuvlardan foydalanadi. Sintaktik tahlilni o‘zlashtirishni kuchaytirish uchun siz ushbu yondashuvlar bilan tanishishingiz kerak. Sintaktik tahlil usullari jumlalardagi so‘zlar, so‘z brikmalari va iboralarning bir-biri bilan qanday bog‘lanishini tushunish uchun asos bo‘lib xizmat qiladi.

Sintaksis tahlili amaliyotda. Sintaksis tahlili, shuningdek, parsing deb ham ataladi, so‘zlarning sintaktik jihatdan bir-biri bilan qanday bog‘lanishini aniqlash uchun gapning grammatik tuzilishini tahlil qilish jarayonidir. Pythonning NLTK moduli sintaksis tahlilini amalga oshirish uchun vositalar va usullarni taqdim etadi. Sintaksis tahlilni amalga oshirishdan avval quyida ba’zi nazariy ma’lumotlar va teglar bilan tanishib olamiz:

Tabiiy tilning ingliz tili uchun ayrim sintaktik kategoriyalari quyidagilardan iborat:

- Sentence(S) – gap;
- Noun Phrase(NP) – ot iborasi;
- Determiner(Det) - aniqlovchi;
- Verb Phrase(VP) – fe’l iborasi;
- Prepositional Phrase(PP) – old qo‘shimchalar;
- Verb(V) – fe’l;
- Noun(N) - ot.

NLTK yordamida sintaktik tahlilni qanday amalga oshirish jarayonini ko‘rib chiqamiz:

```
# kerakli modullar
import nltk
from nltk import pos_tag # tokenlarni teglash uchun
from nltk.corpus import treebank # sintaksis daraxtini tuzish uchun
from nltk.tokenize import word_tokenize # matnlarni so‘z tokenlariga ajratish uchun
from nltk.chunk import RegexpParser # parsing uchun class
# matnni e'lon qilamiz:
sentence = "The quick brown fox jumps over the lazy dog."
# matnni tokenizatsiya qilamiz:
tokens = word_tokenize(sentence)
tokens
# teglasgni amalga oshiramiz
pos_tags = pos_tag(tokens)
print(pos_tags)
# sintaksis analiz uchun gramatik qoidalarni yaratish
grammar = r"""
  NP: {<DT|JJ|NN.*>+}
```

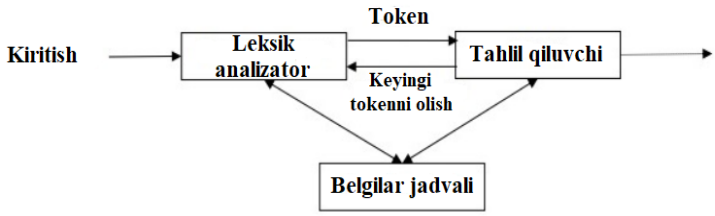
```
    PP: {<IN><NP>}
    VP: {<VB.*><NP|PP|CLAUSE>+$}
    CLAUSE: {<NP><VP>}          """
# yuqoridagi gramatika bo'yicha parser olamiz
chunk_parser = RegexpParser(grammar)
chunk_parser
# teglangan matn uchun parserni ishlatamiz va sintaksis daraxtini olamiz
parsed_tree = chunk_parser.parse(pos_tags)
# sintaksis daraxtini vizualizatsiya qilamiz
#parsed_tree.draw()
```

Neyron tarmoqqa asoslangan yondashuvlar. Sintaktik tahlilda umumiy neyron tarmoqqa asoslangan yondashuvlar takroriy neyron tarmoqlari (RNN), konvolyutsion neyron tarmoqlari (CNN) va transformator modellarini o'z ichiga oladi. Ushbu usullarning har biri turli xil xususiyatlarga ega, ularni qo'llashni bilishdan oldin ular bilan tanishishingiz kerak. Takroriy neyron tarmoqlar ketma-ketlik yoki so'z turkumlarini qayta ishlash imkoniyatiga ega. Bundan tashqari, ular kontekstli ma'lumotni olish uchun mas'ul bo'lgan yashirin holatni saqlaydilar. RNN yuqori ahamiyatga ega bo'lgan o'ziga xos sintaktik tahlil vazifasi nutqning bir qismini belgilashdir. Biroq, RNN ning asosiy cheklovi uning imkoniyatlarini cheklaydigan ketma-ket ishlov berish

atributidir. Bundan tashqari, uzoq muddatli bog‘liqliklar haqida gap ketganda, takroriy neyron tarmoqlari ijobiy ta’sir ko‘rsatadi. Konvolyutsion neyron tarmoqlari kirishda mavjud bo‘lishi mumkin bo‘lgan ma’lumotlar va belgilarni saqlab qolish imkoniyati mavjud. Tabiiy tilni qayta ishlashda hozirda zamonaviy modellari GPT va BERT hisoblanadi. Ushbu modellar mahalliy va global sintaktik ma’lumotlarni to‘plash imkoniyatiga ega. Ularning yuqori imkoniyatlari tufayli ular hozirgi davrning eng zamonaviy vositalari sifatida qaraladi. Ushbu modellar uchun ideal bo‘lgan sintaktik tahlil vazifalaridan ba’zilari tobelikni tahlil qilishni o‘z ichiga oladi.

Tabiiy tilni qayta ishlash - sintaktik tahlili

Sintaktik tahlilning maqsadi gapni to‘g‘ri tuzilishga olib kelishdan iborat bo‘lib, uni rasmiy grammatika qoidalari bilan solishtirganda mazmunliligini tekshiradi. Masalan, “issiq muzqaymoq” kabi jumla semantik analizator tomonidan rad etiladi chunki sintaktik tahlil to‘g‘ri emas. Sintaktik tahlil qilish tabiiy tilda grammatika qoidalariga mos keladigan belgilar qatorlarini tahlil qilish jarayoni sifatida qaraladi. **Parser tushunchasi** – bu tahlil qilish vazifasini amalga oshirish uchun ishlatiladi. Bu kirish ma’lumotlarini (matnni) olish va rasmiy grammatika bo‘yicha to‘g‘ri sintaksisni tekshirgandan so‘ng kirishning tizimli ko‘rinishini berish uchun mo‘ljallangan dasturiy ta’minot komponenti sifatida belgilanishi mumkin. Shuningdek, u odatda tahlil daraxti, mavhum sintaksis daraxti yoki boshqa iyerarxik tuzilma shaklida ma’lumotlar strukturasini quradi.



4.3.3 – rasm. Tabiiy tilga ishlov berish (NLP) da sintaktik tahlil axitekturasi

Grammatika tushunchasi: Grammatika yaxshi tuzilgan dasturlarning sintaktik tuzilishini tavsiflash uchun juda zarur va muhim ahamiyatga ega. Adabiy ma'noda ular tabiiy tillarda suhbatning sintaktik qoidalarini bildiradi. Tilshunoslik ingliz, hind va boshqalar kabi tabiiy tillar paydo bo'lganidan beri grammatikalarni aniqlashga harakat qilinib kelinmoqda. Rasmiy tillar nazariyasi kompyuter fanlari sohalarida, asosan dasturlash tillari va ma'lumotlar strukturasida qo'llaniladi. Misol uchun, "C" tilida, aniq grammatik qoidalar ro'yxatlar va matnlarda qanday qilib funksiyalar yaratilganligini bildiradi.

Grammatikaning matematik modeli 1956-yilda **Noam Xomskiy** tomonidan berilgan bo'lib, u kompyuter tillarini yozish uchun samaralidir. Matematik jihatdan G grammatikasi rasmiy ravishda 4-tube (N, T, S, P) sifatida yozilgan, bu yerda :

- **N** yoki $\mathbf{V_N}$ = terminal bo'lmagan belgilar to'plami, ya'ni o'zgaruvchilar.
- **T** yoki $\sum$ = terminal belgilar to'plami.
- **S** = Boshlanish belgisi bu erda $S \in N$

- **P** terminallar va terminallar uchun ishlab chiqarish qoidalarini bildiradi. U a → b ko‘rinishga ega, bu erda a va b $V_N \cup \sum$ satrlari va a ning kamida bitta belgisi V_N ga tegishli.

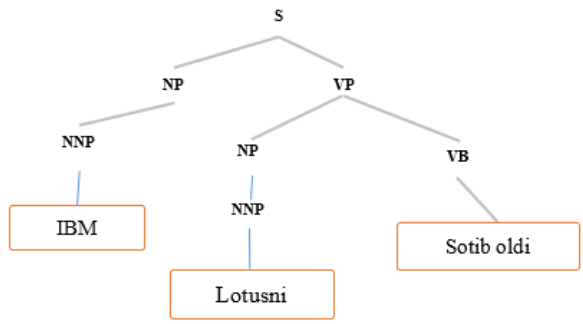
Gap tuzilishi yoki grammatikasi:

Xomskiy grammatikalarining iyerarxiyasida tillar 4 guruhga ajratilgan. Ularni ichida eng qiziqarlisi kontekstsiz grammatika va tillar. Kontekstsiz grammatika tushunchasi xorijiy manbalarda “Ehtimoliy kontekstsiz grammatika” deyiladi. Sintaktik analizator gaplarni nodeterministik tahlil qilish uchun 2 xil algoritmni amalga oshiradi:

yuqoridan pastga (top-down);

pastdan yuqoriga (bottom-up);

Yuqoridan pastga tomon yo‘naltirilgan algoritm S (jumla)ni keng ko‘lamining tuzilishini ehtimoliy tuzilishidan boshlanadi, so‘ngra bu ehtimolni rekursiv so‘zlar darajasida aniqlaydi.



4.3.4-rasm. Sintaktik analizator gaplarni nodeterministik tahlil qilish yuqoridan pastga algoritmi

Pastdan yuqoriga tomon yo‘nalgan algoritm gap daraxtining barglaridan ildizigacha sintaktik tahlilini yaratadi. Ya’ni so‘zlar bilan

tahlil qilishni boshlaydi va birinchi so‘z juftlarini bog‘laydi. Keyin yangi so‘zlarni yoki boshqa tegishli bo‘lgan juftlarni bu juftlarga bog‘laydi. Bog‘lanish jarayoni asta-sekin boshlang‘ich noterminal S ga keladi, ya’ni gaplardagi barcha so‘zlar bir tuzilishga bog‘lanadi.

Jahonda kompyuter lingvistika sohasida qator sintaktik analizatorlar yaratilgan bo‘lib, ular shu ikkita algoritm asosida sintaktik tahlilni amalga oshiradi. Hozirda PetitParser, Etap-3, Bison kabi ko‘plab sintaktik analizatorlardan foydalanilmoqda. Bu sohada qator dunyo olimlari N.V. Arefev, Zo Zon Su, V.A. Bushtedt, A. Sokirko, S. Toldova tadqiqot olib borganlar. O‘zbek tilshunosligida G‘.Abdurahmonov, A.Sulaymonov, X.Xoliyorov, J.Omonturdiyev, A.G‘ulomov, M.Asqarov, N.Maxmudov, A.Nurmonov kabi qator olimlar o‘zbek tili grammatikasida matnlarni sintaktik tahlili borasida izlanishlar olib borganlar. Ular asosan gaplarni sintaktik tahlil qilishda, avvalo, morfologik tahlil natijalariga tayangan.

Bog‘liqlik grammatikasi: U Lucien Tesniere tomonidan taqdim etilgan. Tobelik grammatikasi (TG) okrug grammatikasiga qarama-qarshidir, chunki unda frazema tugunlari mavjud emas.

Misol: Bog‘lik grammatikasiga misol keltirishdan oldin, biz bog‘lik grammatikasi va bog‘lik munosabati haqidagi asosiy fikrlarni bilishimiz kerak.

- TGda lisoniy birliklar, ya’ni so‘zlar bir-biri bilan yo‘naltirilgan bog‘lanishlar orqali bog‘lanadi.
- Fe’l gap tarkibining markaziga aylanadi.

• Har bir boshqa sintaktik birliklar fe'lga yo'naltirilgan bog'lanish nuqtai nazaridan bog'lanadi. Bu sintaktik birliklar ***bog'liqliklar*** deyiladi. Gap tahlili quyidagicha amalga oshiriladi dastlab morfologik tahlil, undan keyin sintaktik tahlil va semantik tahlil.

4.3.3. Semantik tahlili

Semantik tahlil kompyuter matnning ma'nosini faqat alohida so'zlarga qarab emas, butun matnni tahlil qilish orqali amalga oshiriladi. Semantik tahlil haqida gap ketganda, so'z qanday kontekstda ishlatilishi juda muhimdir. Semantik tahlil NLP uchun juda muhim, chunki uning jarayonlari so'zlarning turli ma'nolarini aniqlashga imkon yaratadi. Bundan tashqari, bu jarayonlar mashinaga butun jumlalar va matnlarning ma'nosini tushunishga yordam beradi. NLP semantikasining ikkita tipik jarayoni mavjud bo'lib, ular ham o'z navbatida bajariladigan vazifasiga ko'ra qator guruhlarga bo'linadi:

So'z ma'nosini ajratib ko'rsatish.

O'zaro munosabatlarni ajratib olish.

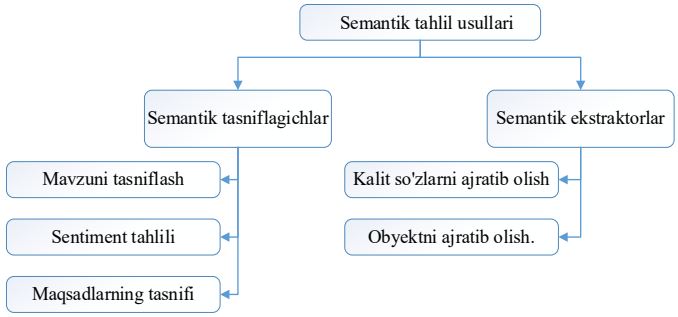
So'z ma'nosini ajratib ko'rsatish. Ushbu tushuncha har qanday so'zning kontekstini aniqlashning avtomatik jarayonini tavsiflaydi. Tabiiy tilda bir so'z ko'pincha bir nechta ma'noga ega. Masalan, "kul" so'zi yong'in natijasida hosil bo'lgan *kukun* ma'nosini anglatishi mumkin, ammo u *kulmoq* fe'lining buyruq shaklini ifodalashi ham mumkin. Kompyuterning vazifasi so'zni ma'lum bir kontekstda tushunish va eng yaxshi ma'noni tanlashdir.

O'zaro munosabatlarni ajratib olish. O'zaro munosabatlarni

ajratib olish tushunchasi ushbu obyektlar orasidagi semantik munosabatlarni chiqarish jarayonini tavsiflaydi. Aytaylik, kompyuter “Men qahva ichyapman” jumlasini tahlil qilishi kerak. Jumla “men” va “qahva” kabi ikkita so‘zdan iborat bo‘lib, bu ikki so‘z o‘rtasidagi munosabat “ichimlik” so‘zidir.

Semantik tahlil turlarini 4.3.5-rasmda ko‘rish mumkin.

Keltirilgan ma’lumotlar shuni ko‘rsatadiki, bizning tadqiqot ishimiz so‘z ma’nosini aniqlash vazifaning bir qismini bajarishga xizmat qiladi. Semantik tahlilni avtomatik amalga oshirish O‘zbek tili milliy korpusida axborot-qidiruv tizimining faoliyatini mukammallashtiradi. Izlanayotgan leksik birliklarning semantikasini aniqlash NLP sohasidagi aosiy vazifalardan biri.



4.3.5 - rasm. Semantik tahlil turlari

Semantik tahlil orqali semantik qidiruv amalga oshiriladi. Semantik tahlil qanchalik mukammal ishlab chiqilgan bo‘lsa, qidiruv shunchalik samarali natija beradi. Semantik tahlilni amalga oshirish bevosita lingvistik resurslarga bog‘liq. Leksik resurslarga lug‘at, tezaurus va ontologiyalar kiradi. Lug‘at leksik axborotning eng an’anaviy shakli

hisoblanadi: ular birliklar (odatda, so‘z yoki so‘z birikmasi), tuzilishi va leksikaning qamrovi (turli sohalarning atamalar lug‘ati, umumiy leksika lug‘ati, sinonim va paronimlar lug‘ati va h.) bilan farqlanadi.

Semantik tahlilning amaliyotda qo‘llanilishi. Semantik tahlil - matndan ma’no chiqarish jarayoni. Bu kompyuterlarga jumlalarni, paragraflarni yoki butun hujjatlarni ularning grammatik tuzilishini tahlil qilish va ma’lum bir kontekstdagi alohida so‘zlar o‘rtasidagi munosabatlarni aniqlash orqali tushunish va izohlash imkonini beradi.

NLTK (Natural Language Toolkit) semantik tahlil uchun bir qancha vositalar va usullarni o‘z ichiga oladi. Biz bunga python dasturlash tilida quyidagi misollarni keltiramiz:

1. *WordNet* - bu so‘zlar orasidagi semantik munosabatlarni ta'minlovchi leksik ma’lumotlar bazasi. NLTK WordNet interfeysini o‘z ichiga oladi, bu sizga sinsetlarga (sinonimlar to‘plami) kirish va gipernimlar (ko‘proq umumiy atamalar) va giponimlar (aniqroq atamalar) kabi semantik munosabatlarni o‘rganish imkonini beradi.

```
from nltk.corpus import wordnet
# "happy" so‘zining sinonimlarini wordnet bazisidan topish
synonyms = []
for syn in wordnet.synsets("happy"):
    for lemma in syn.lemmas():
        synonyms.append(lemma.name())
print(set(synonyms))
```

Natiaja: {'well-chosen', 'happy', 'glad', 'felicitous'}

2. *Word Embeddings - So'zlarni joylashtirish:* So'zlarni joylashtirishlar semantik jihatdan o'xshash so'zlar bir-biriga yaqinroq bo'lgan uzluksiz vektor fazodagi so'zlarni ifodalaydi. NLTK o'rnatilgan so'zlarni joylashtirish modellarini ta'minlamaydi, lekin siz Gensim yoki TensorFlow kabi boshqa kutubxonalardan oldindan o'rgatilgan so'zlarni o'rnatishni yuklash uchun foydalanishingiz mumkin.

3. *Distributional Semantics - Tarqatish semantikasi:* Tarqatish semantikasi taqsimot gipotezasiga asoslanadi, ya'ni o'xshash ma'noga ega bo'lgan so'zlar o'xshash kontekstlarda uchraydi. Biz NLTK-dan atama chastotasi-teskari hujjat chastotasi (TF-IDF) ballari yoki birgalikdagi matritsalarni hisoblash kabi vazifalar uchun foydalanishimiz mumkin.

4. Semantic Similarity - Semantik o'xshashlik: NLTK so'zlar yoki iboralar o'rtasidagi o'xshashlikni ularning semantik xususiyatlariga qarab o'lchash usullarini taqdim etadi.

```
# modullarni import qilamiz
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
from collections import defaultdict
import numpy as np
# matn e'lon qilamiz va uni tokenlarga ajratamiz
```

```
text = "This is a sample sentence for demonstrating distributional semantics. Distributional semantics is based on the distributional hypothesis."

tokens = word_tokenize(text)

print(tokens)

from nltk.corpus import wordnet

# Misol: "dog" va "cat" so‘zlarining o‘xshashligini hisoblaymiz

dog = wordnet.synset("dog.n.01")

cat = wordnet.synset("cat.n.01")

similarity = dog.wup_similarity(cat)

print(similarity)
```

Natijalar: ['This', 'is', 'a', 'sample', 'sentence', 'for', 'demonstrating‘, 'distributional', 'semantics', '.', 'Distributional', 'semantics', 'is', 'based', 'on', 'the', 'distributional', 'hypothesis', '.']

0.8571428571428571

5. *Textual Entailment - Matnni kiritish*: Matnga taalluqlilik deganda birining ma’nosi boshqasidan mantiqiy kelib chiqadigan ikkita matn bo‘lagi o‘rtasidagi munosabat tushuniladi. NLTK matnli ma’lumotlar to‘plami bilan ishlash va ushbu vazifa uchun modellarni amalga oshirish uchun resurslarni o‘z ichiga oladi.

Leksik resurslarning murakkabroq turlari tezaurus va ontologiyalardir. Tezaurus – semantik lug‘at, ya’ni so‘zlarning semantik aloqalari – sinonimlar, assotsiatsiyalar taqdim etilgan lug‘at. Misol sifatida rus tili uchun ijtimoiy-siyosiy leksikani qamrab oluvchi axborot-

qidiruv RuTez tezaurusini keltirish mumkin. Ontologiya tushunchasi tezaurus tushunchasi bilan chambarchas bog‘liq. Ontologiya – turli vazifalar uchun ko‘p foydalanishga qaratilgan muayyan bilim sohasi tushunchalari, obyektlar majmuyi. Ontologiyalar tildagi mavjud so‘z birikmalari asosida yaratilishi mumkin. Bu holda ular lingvistik ontologiya deb ataladi. Xuddi shunday lingvistik ontologiya WordNet tizimi hisoblanib, u inglizcha so‘zlarni o‘z ichiga olgan katta leksik manba. Unda ot, sifat, fe’l, ravish hamda ularning bir necha turdagi semantik bog‘lanishlari keltirilgan.

Semantik komponent ontologiya asosida amalga oshirilishi mumkin. Ushbu yondoshuvni amalga oshirish uchun matn resurslariga mos tushunchalarga ta’rif berish va har bir sinf uchun matn resurslari tarkibining formal strukturasini aniqlash zarur. Semantik tahlil formal semantik yondoshuv asosida ham amalga oshirilishi mumkin. Bu holda gapning qiymati matematik formula yordamida ifodalanadi. Semantik komponent matnning semantik va sintaktik tahlil natijalariga asoslanib, har bir gapning sintaktik tuzilishini tasvirlovchi belgiga ega bo‘ladi

Qidiruv tizimlari SQ (Semantik Qidiruv)ni amalga oshirishda juda ko‘p muammolarga duch keladi. Jumladan, qidiruv so‘rovini kiritishda foydalanuvchi aynan nimani nazarda tutganini, ya’ni turli kontekstlarda so‘z yoki iboraning bir qancha ma’nolari mavjudligini aniqlash. Semantik qidiruv kompyuter yo‘li bilan hal qilib bo‘lmaydigan muammolarni yechishda yordam berolmaydi. Semantik qidiruv nafaqat kontekstni, balki boshqa omillar tahlilini ham inobatga oladi. Data Science Blogathon nashri o‘zining “Step by Step Guide to Master NLP – Semantic Analysis”

ilmiy maqolasida semantik tahlil qilishning qadamlarini sanab o‘tgan. Unga ko‘ra:

1. Semantik tahlilning birinchi tarkibiy qismi so‘z ma’nosini ajratish bo‘lib, unda biz alohida so‘zlarning ma’nosini o‘rganamiz. Bu komponentni leksik-semantika deb nomlagan.

2. Alohida so‘zlarning kombinatsiyalarini, brikuvchilarini o‘rganish.

1. Semantik takomillashtirish algoritmlarini matn ma’lumotlari namunalari bilan ta’minlash orqali Machine learning algoritmlari yordamida avtomatik ravishda semantik tahlilni amalga oshirish mumkin. Qidiruv tizimi va uning oldingi natijalari asosida aniq bashorat qilishga o‘rgatishimiz mumkin.

2. Semantik tahlil elementlarini alohida o‘rganish zarur va ular quyidagi bo‘laklardan tashkil topgan.

– giponimiya;
– omonimiya;
– polisemiya;
– polifunksionallik;
– sinonimiya;
– antonimiya;
– paronimiya;
– giperonimiya;
– grodonomik munosabatlar.

1. Semantik tizimni loyihalashtirish. Tizimning asosi bo‘lgan ma’lumotlarni tizimlashtirish, ya’ni:

– obyektlarni yaratish;

– izohlash;

– bog‘lanishlar;

– strukturalar, masalan, ot, fe’l birikmalarining strukturalari.

2. Semantik tahlil ma’noni ifodalash uchun quyidagi yondoshuvlardan foydalanadi:

– birinchi tartibli predikatlar mantiqi;

– semantik tarmoqlar;

– formalar;

– qoidalarga asoslangan arxitektura;

– case grammatikasi;

– konseptual grafika.

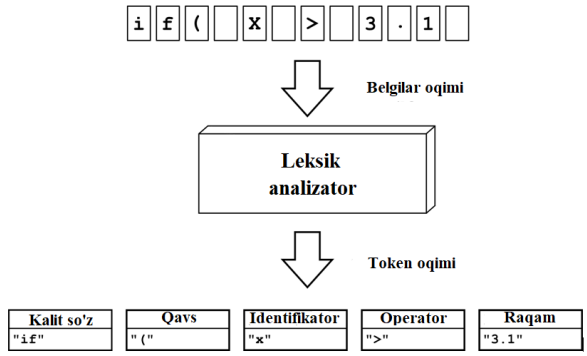
Semantik tahlil yaxshi natijalar beradi, lekin u ham ko‘proq o‘qitish va hisoblashni talab qiladi. Semantik tahlil algoritmi yaxshiroq ishlashi uchun kattaroq ma’lumotlar korpusi bilan o‘qitilishi kerak. Yillar davomida NLP sintaktik tahlildan semantik tahlilga o‘tdi.

Ma’noni ifodalash semantik tahlilning asosiy vazifasi hisoblanadi. Bu birinchi navbatda, so‘zlar, iboralar va jumlalarning so‘zma-so‘z ma’nosi bilan bog‘liq. Semantik tahlilning maqsadi matndan aniq ma’no yoki lug‘at ma’nosini ajratib olishdir.

Biz ma’lumki, leksik tahlil so‘z ma’nolari bilan shug‘ullanadi. Leksik tahlil kichikroq tokenlarga asoslanadi, semantik tahlil esa kattaroq tokenlarga asoslanadi.

Leksik tahlil. Leksik tahlil o‘tkazilganda, tahlil dastlab gapni ajratib, “tilni tashkil etuvchi tushunarli lug‘at” kabi leksema so‘z

birikmalariga ajratadi. Ushbu tahlil iborani “lug‘at” va “til” kabi so‘z leksemalariga ajratadi va tasniflaydi. NLP da leksik tahlilning ko‘plab afzalliklari mavjud. Bu natijalarni aniqligini, shuningdek, jarayonning samaradorligini oshirishga yordam beradi. Bundan tashqari, u bizga ko‘proq turli xil ma’lumotlar bilan ishlash imkonini beradi 4.4.7-rasmda leksik tahlil arxitekturasini ko‘rishimiz mumkin.



4.3.6 – rasm. Tabiiy tilga qayta ishlov berish (NLP) da leksik tahlil axitekturasi

Xulosa qilib aytadigan bo‘lsak, leksik tahlil matnni tahlil qilish uchun alohida tokenlarga bo‘lish jarayonidir. Bu jarayon axborotni qidirish, ularni qayta ishlash kabi vazifalar uchun zarurdir, chunki u mashinalarga inson tilini tushunishdagi dastlabki qayta ishlash imkonini beradi. **Leksik tahlil** matn oqimini “tokenlar” deb ataladigan eng kichik ma’noli birliklarga bo‘lish jarayonidir. Bu jumlani alohida so‘zlarga ajratish va keyin bu so‘zlarni otlar, fe’llar va sifatlarni tasniflash kabi jarayonlarni o‘z ichiga oladi.

Leksik tahlil bir necha jarayon bosqichlarga bo‘linadi:

- Tokenization – tokenizatsiya;

- Normalization – normallashtirish;
- Stopword Removal – matnda ma'no anglatmaydigan so'zlarni olib tashlash;
- Stemming va lemmatization – so'zni o'zak shakliga keltirish;
- Part-of-Speech (POS) Tagging – har bir so'z (token) ni teglash. Masalan: gapdagi so'zlarni "ot", "sifat", "son", "fe'l" larga ajratish;
- Named Entity Recognition (NER) – Atoqli otlar (ismlar, joy nomalari) ni tanib olish.

Quyida har bir jarayonni python dasturlash tilida misollar bilan ko'rsatamiz. Buning uchun bizga pythonning "nltk" – natural language toolkit modulidan foydalanamiz.

1. Tokenizatsiya:

```
# kerakli modullarni import qilamiz:

import nltk

from nltk.tokenize import word_tokenize

# biror matnni o'zgaruvchiga o'zlashtiramiz:

text = " Tokenizatsiya - bu matnni so'zlarga va tinish belgilariga bo'lish jarayoni."

# matnni tokenizatsiya qilamiz va chop etamiz:

tokens = word_tokenize(text)

tokens

#Natiaja:
```

```
[' Tokenizatsiya',
 ' bu',
 ' matnni',
 “ so‘zlarga”,
 ' va',
 ' tinish',
 ' belgilariga ',
 ' bo‘lish  ‘,
 ' jarayoni ', '.']
```

2. Normalization. Yuqoridagi kodlarni davom ettiramiz:

```
normalized_tokens = [token.lower()
for token in tokens if token.isalnum()]
print(normalized_tokens)

#Natija: [' tokenizatsiya',' bu',' matnni',“ so‘zlarga”,
 ' va',' tinish',' belgilariga',' bo‘lish‘,'jarayoni',
 '.']
```

Ko‘rib turganingizdek tokenlarni barchasi bir xil kichik harflarga o‘tkazildi.

3. Stopword removal.

```
# stopwordlarni ajratib oluvchi obyektni import qilamiz
from nltk.corpus import stopwords
# ingliz tilidagi stopwordlarni ro‘yhatini olamiz:
stop_words = set(stopwords.words('uzbek'))
```

```
filtered_tokens = [token for token in normalized_tokens if token
not in stop_words]

# stopwordlarsiz tokenlarni chop etamiz:
filtered_tokens
```

Natija: ["tokenizatsiya", "jarayon", "bo‘lish", "matn", "so‘zlar", "t inish belgilari"]

E'tibor berganingizdek tokenlar orasida "bo‘ladi", "yoki", "va" kabi ma'no anglatmaydigan so‘zlar yani stopwordlar yo‘q.

4. Stemming va Lemmatization:

```
# kerakli classlarni import qilamiz

from nltk.stem import PorterStemmer

from nltk.stem import WordNetLemmatizer

# Stemming

stemmer = PorterStemmer()

stemmed_tokens = [stemmer.stem(token) for token in
filtered_tokens]

print(stemmed_tokens)
```

Natija: ["token", "jarayon", "bo‘linish", "matn", "so‘z", "tinish bel gilari"]

```
#Lemmatization

lemmatizer = WordNetLemmatizer()
```

```
lemmatized_tokens = [lemmatizer.lemmatize(token) for token in
filtered_tokens]

print(lemmatized_tokens)
```

Natiaja: ["tokenizatsiya", "jarayon", "bo‘linish", "matn", "so‘z", " tinish belgilari"]

```
pos_tags = nltk.pos_tag(filtered_tokens)

pos_tags

[(' tokenizatsiya ', 'NN'),
 (' jarayon', 'NN'),
 (' bo‘linish', 'VBG‘),
 (' matn', 'NN'),
 (“so‘z”, 'NNS'),
 (' tinish belgilari', 'NN')]
```

5. Named Entity Recognition (NER):

Keling ushbu bosqichga misolni oldingi barcha bosqichlarni takrorlagan xolda keltiraylik:

```
# kerakli modullarni import qilamiz:
import nltk
from nltk import word_tokenize, pos_tag, ne_chunk

# matn e'lon qilish
text = "Albert Einstein was born in Germany."

# tokenlash
tokens = word_tokenize(text)
```

```
# teglash
pos_tags = pos_tag(tokens)

# NER ni amalga oshirish
ner_tags = ne_chunk(pos_tags)

# natijalarni chop etish
print(ner_tags)

Natija:
(S
  (PERSON Albert/NNP)
  (PERSON Einstein/NNP)
  was/VBD
  born/VBN
  in/IN
  (GPE Germany/NNP)./.)
```

Leksik semantika - semantik tahlilning birinchi bosqichi bo‘lib, u aniq so‘zlarning ma’nosini tekshirishni o‘z ichiga oladi. Unga yakka so‘zlar, qo‘shma so‘zlar, affikslar (kichik birliklar) va iboralar ham kiradi. Leksik bo‘laklar barcha so‘zlarni jamlaydi. Boshqacha qilib aytganda, leksik semantika leksik birikmalar, gap ma’nosi va gap sintaksisi o‘rtasidagi munosabatni o‘rganadi.

Matndan nutqqa o‘tish tahlili. Google Text-to-Speech (TTS) kabi tayyor API lardan foydalanmasdan aniqroq va ishonchli matndan nutqqa (TTS) tizimini yaratish uchun biz fonema sintezi uchun mavjud

texnologiyalardan foydalanadigan tizimni joriy qilamiz. Nutq sintezining murakkabligini hisobga olgan holda, noldan yuqori sifatli TTS yaratish odatda tadqiqot va sanoat ilovalarida mavjud bo‘lgan zamonaviy usullarni talab qiladi.

Biz quyidagi Python kutubxonalaridan foydalanamiz.

pydubaudio manipulyatsiyasi uchun.

simpleaudioaudio tinglash uchun.

Ushbu kutubxonalarni pip yordamida o‘rnatamiz:

pip install pydub simpleaudio

Tayyorgarlik.

1. Fonema yozish: fonema tovushlarini yozib olish va saqlash. Har bir fayl fonema yoki harfga mos kelishi kerak. Ushbu fayllarni katalogda saqlash, masalan, phonemes/.
2. Fonema xaritasi: Belgilarni mos keladigan fonema fayl yo‘llari bilan taqqoslaydigan lug‘at yaratish.

Bosqichma-bosqich amalga oshirish

1-qadam: Matnni fonemaga aylantirish

Matnni fonema fayl yo‘llari ro‘yxatiga joylashtirish funksiyasini yaratish.

```
phoneme_map = {
    'a': 'phonemes/a.wav',
    'b': 'phonemes/b.wav',
    'c': 'phonemes/c.wav',
    'd': 'phonemes/d.wav',
    'e': 'phonemes/e.wav',
```

```
    # Barcha kerakli harflar va fonemalar uchun xaritalarni qo‘shish
}
def text_to_phonemes(text):
    text = text.lower()
    phonemes = []
    for char in text:
        if char in phoneme_map:
            phonemes.append(phoneme_map[char])
        else:
            print(f" Ogohlantirish: uchun fonema xaritasi yo‘q
'{char}'")
    return phonemes
```

2-qadam: Fonemalarni birlashtirish. Fonemalarga mos keladigan audio fayllarni birlashtirish.

```
from pydub import AudioSegment
def concatenate_phonemes(phoneme_files):
    combined = AudioSegment.empty()
    for phoneme_file in phoneme_files:
        phoneme_audio = AudioSegment.from_wav(phoneme_file)
        combined += phoneme_audio
    return combined
```

3-qadam: Ovozni tinglash. Kutubxonadan foydalanib, birlashtirilgan audioni ijro etish “simpleaudio”.

```
import simpleaudio as sa
def play_audio(audio_segment):
    playback =  sa.play_buffer(audio_segment.raw_data,
    num_channels=audio_segment.channels,
    bytes_per_sample=audio_segment.sample_width,
    sample_rate=audio_segment.frame_rate)
    playback.wait_done()
```

To‘liq amalga oshirish. Barcha bosqichlarni to‘liq TTS tizimiga birlashtirish:

```
import os
from pydub import AudioSegment
import simpleaudio as sa
phoneme_map = {
  'a': 'phonemes/a.wav',
  'b': 'phonemes/b.wav',
  'c': 'phonemes/c.wav',
  'd': 'phonemes/d.wav',
  'e': 'phonemes/e.wav',
}
def text_to_phonemes(text):
  text = text.lower()
  phonemes = []
  for char in text:
    if char in phoneme_map:
```

```
            phonemes.append(phoneme_map[char])
        else:
            print(f" Ogohlantirish: uchun fonema xaritasi yo‘q '{char}'")
    return phonemes

# Step 2: Phoneme Concatenation
def concatenate_phonemes(phoneme_files):
    combined = AudioSegment.empty()
    for phoneme_file in phoneme_files:
        phoneme_audio = AudioSegment.from_wav(phoneme_file)
        combined += phoneme_audio
    return combined

# Step 3: Audio Output
def play_audio(audio_segment):
    playback = sa.play_buffer(audio_segment.raw_data,
    num_channels=audio_segment.channels,
    bytes_per_sample=audio_segment.sample_width,
    sample_rate=audio_segment.frame_rate)
    playback.wait_done()

# Example Usage
text = "abcde"
phoneme_files = text_to_phonemes(text)
audio_segment = concatenate_phonemes(phoneme_files)
play_audio(audio_segment)
```

Ushbu kod dasturi oldindan yozib olingan fonemalardan foydalangan holda matndan nutqqa asosiy dasturni taqdim etadi. Dastur quyidagi bosqichlarni bajaradi:

Matnga oldindan ishlov berish: Matnni fonema fayl yo‘llari ro‘yxatiga o‘zgartiradi.

Fonema birikmasi: Uzluksiz nutq signali shaklini hosil qilish uchun fonema audio fayllarini birlashtiradi.

Ovozni ijro etish: yordamida yaratilgan audioni ko‘rsatadi simpleaudio.

Ushbu yondashuv funksional bo‘lsa-da, talaffuzning aniqligi va tabiiyligida cheklovlarga ega. Murakkab Text-to-Speech (TTS) tizimlari uchun odatda chuqur o‘rganishga asoslangan modellar (masalan, Tacotron, WaveNet) kabi zamonaviy texnologiya texnologiyalar qo‘llaniladi, ular katta hisoblash resurslari va mashinani o‘rganish bo‘yicha keng imkoniyatlarni taqdim etadi.

4.3-bo‘lim bo‘yicha amaliy qism

4.3-bo‘lim bo‘yicha amaliy qism quyidagi mavzular bo‘yicha keltirilgan. Bunda keltirilgan havola yoki QR kodni skaner qilish orqali GitHubga murojat qilib dasturiy kodidan foydalanish imkoniyatiga ega bo‘lasiz.

https://github.com/TUITAI/

XULOSA

Sun’iy intellekt matematika, falsafa, iqtisod, nevrologiya, psixologiya, kompyuter injiniringi, boshqaruv nazariyasi, shuningdek, tilshunoslik kabi turli fanlardan foydalangan holda ishlab chiqilgan. Matematika, statistika, mantiq va dasturlash asoslari sun’iy intellekt yechimlarini ishlab chiqishda muhim ro‘l o‘ynaydi. Neyron tarmoq boshqaruv nazariyasidan boshlanadi va endi chuqurlashtirilgan o‘rganish (Deep Learning) ommalashgandan so‘ng, ko‘pchilik sun’iy intellekt yechimlari shu yondashuvga asoslanadi. Shuning uchun biz ta’limga nisbatan ko‘proq fanlararo munosabatni o‘zgartirishimiz kerak. Sun’iy intellekt ta’limdagi asosiy faoliyatni, masalan, baholashni avtomatlashtirishi mumkin. O‘qituvchilar endi deyarli barcha turdagi ko‘p tanlov bo‘yicha baholashni, bo‘sh joyni to‘ldirish testini va talabalar yozma ishini avtomatik baholashni avtomatlashtirishi mumkin.

Bugungi kunda sun’iy neyron tarmoqlari an’anaviy dasturlarga qaraganda ko‘proq funksionallikka ega. Hozirgi vaqtda turli xil dasturiy ta’minot va yaratish uchun mo‘ljallangan ko‘plab ochiq kutubxonalar mavjud. Neyron tarmoqlari o‘z mohiyatiga ko‘ra juda oddiy matematik apparatga ega. Neyron tarmoqlar hisoblashda unchalik murakkab emas. Kompyuter fanlari va dasturlash tili bo‘yicha boshlang‘ich bilimga ega bo‘lgan deyarli har bir kishi o‘z neyron tarmog‘ini yaratishi mumkin. Neyron tarmoqlar va katta ma’lumotlar(Big data) fani faol rivojlanmoqda va kelajakda juda dolzarb hisoblanadi. Shunga asosan biz darsligimizda sun’iy intellekt asoslarining eng boshlang‘ich tushunchalarini keng yoritdik. Darslikdagi kamchiliklar keyingi nashrlarda to‘ldirib boriladi.

QISQARTMA SO‘ZLAR

SI – Sun’iy intellekt.

Data Mining (DM)- ma’lumot qazib olish yoki ma’lumotlar tahlili.

BAT- bilimlarga asoslangan tizimlar.

ML – mashinali o‘qitish(mashinani o‘qitish).

SVM- Support vektor mashinasi.

BPE - Back Propagation Error.

MNIST - Modified National Institute of Standards and Technology.

Batch _size - har bir bo‘lak soni.

CNN- Convolutsion neyron tarmoqlari.

RNN - Recurrent neyron tarmoq.

NLP - Natural Language Processing.

FOYDALANILGAN ADABIYOTLAR

1. “Raqamli O‘zbekiston 2030” strategiyasini tasdiqlash va uni samarali amalga oshirish chora-tadbirlari to‘g‘risida. O‘zbekiston Respublikasi Prezidentining Farmoni, 05.10.2020 yildagi PF-6079- son.
2. “Sun’iy intellekt texnologiyalarini jadal joriy etish uchun shart-sharoitlar yaratish chora-tadbirlari to‘g‘risida”. O‘zbekiston Respublikasi Prezidentining qarori, 17.02.2021 yildagi PQ-4996-son.
3. “Raqamli texnologiyalar va sun’iy intellektni rivojlantirish ilmiy-tadqiqot instituti faoliyatini tashkil etish to‘g‘risida”. O‘zbekiston Respublikasi Vazirlar Mahkamasining qarori, 31.07.2021 yildagi 475- son.
4. “Sun’iy intellekt texnologiyalarini qo‘llash bo‘yicha maxsus rejimni joriy qilish chora-tadbirlari to‘g‘risida”. O‘zbekiston Respublikasi Prezidentining qarori, 26.08.2021 yildagi PQ-5234-son.
5. “2022-2026-yillarga mo‘ljallangan yangi o‘zbekistonning taraqqiyot strategiyasi to‘g‘risida”. O‘zbekiston Respublikasi Prezidentining Farmoni, 28.01.2022 yildagi PF-60-son.
6. Birlashgan millatlar tashkilotining “elektron hukumat tadqiqoti” hisobotida O‘zbekiston respublikasining reytingini yaxshilashga doir qo‘shimcha chora-tadbirlar to‘g‘risida. O‘zbekiston Respublikasi.
7. Prezidentning qarori, 14.09.2023 yildagi PQ-308-son.
8. Zaynidinov H.N., Nurmurodov J.N. Qobilov S.Sh. Digital Signal and Image Processing Using Neural Networks/2023 Seminar on Information Systems Theory and Practice (ISTP)/ IEEE DOI: 10.1109/ISTP60767.2023.10426883.
9. Х.Н.Зайнидинов, О.У.Маллаев. Сплайнлар асосида кўп ядроли процессорлари учун параллел усуллар ва алгоритмлар. Монография. – Ташкент. 2022 й. ISBN: 978-9943-7473-3-3
10. Зайнидинов Х.Н., Усманов Б.Ш. Архитектура компьютеров и компьютерных систем, //Ташкентский университет информационных технологий им. Мухаммада Ал-Хоразмий. -Т., 2020-640 с.

11. Х.Н.Зайнидинов, Б.Р.Азимов. Сигналларга рақамли ишлов беришнинг сплайн-усуллари ва алгоритмлари. Монография. – Ташкент. 2022 й. «Adast poligraf»-192 б. ISBN: 978-9943-9460-8-8.

12. Hakimjon Zaynidinov, Javoxir Nurmurodov Qobilov Sirojiddin. Application of Machine Learning Methods for Signal Processing in Piecewise-Polynomial Bases2023 IX International Conference on “Information Technology and Nanotechnology (ITNT)”.Samara-2023. - 4p.DOI: 10.1109/ITNT57377.2023.10139002.

13. Introduction to Deep Learning: From Logical Calculus to Artifical Intelligence : монография / S. Skani. - Cham : Springer, 2018. - 191 p. - 1 экз. - ISBN 978-3-319-73003-5

14. Искусственные нейронные сети [Текст] : учебник / В. С. Ростовцев. - 2-е изд., стереотип. - СПб. ; М. ; Краснодар : Лань, 2021. - 216 с. : ил. - (Высш. образование). - Библиогр.: с. 210. - 1 (адади 50) экз. - ISBN 978-5-8114-7462-2.

15. Sun’iy intellekt asoslari. Mashinali o‘qitish [Text] : монография / S. Komolov, Sh. Raxmatov. - T. : Ijod nashr, 2022. - 104 b. - Adabiyot.: b. 117. - ISBN 978-9943-6561-8-5.

16. Нейронные сети: полный курс [Текст] : монография / С. Хайкин ; пер. с англ. - 2-е изд., испр. - М. ; СПб. : Диалектика, 2020. - 1104 с. : ил. - Библиогр.: с. 996. - Предм. указ.: с. 1070. - 1 - ISBN 978-5-907144-22-4 .

17. Искусственный интеллект: Кн. 1. Системы общения и экспертные системы. Справочник. / Под ред. Э.В. Попова.-М.: Радио и связь, 1990. – 464 с

18. Горбань А.Н., Дунин-Барковский В.Л., Кирдин А.Н., и др. Нейроинформатика. – Новосибирск: Наука. Сибирское отделение РАН, 1998. – 296 с

19. Джаратано Дж., Райли Г. Экспертные системы: принципы разработки и программирование. – М.: ООО “И.Д. Вильямс”, 2017. – 1152 с

20. Stephen Marsland. Machine Learning and Algorithmic Perspective. International Standard Book ISBN 978-1-4665-8333-7 (eBook - PDF), 2015.

21. Aurelian Geron, Hands on Machine Learning with Scikit-Learn Keras&Tensorflow // Second edition Concepts, Tools, and Techniques to Build Intelligent Systems, 2019, 510 pages

22. Monica Bianchini, Milan Simic, Ankush Ghosh, Rabindra Nath Shaw Machine Learning for Robotics Applications, Springer, 2020, 175 pages.

23. Oliver Theobald, “Machine Learning for Absolute Beginners”, second edition, 2017, 128 pages

24. Жуков Л.А., Решетникова Н.В. Приложения нейронных сетей: Учебное пособие для студентов, учащихся лицея и ЗПШНИ / Л. А. Жуков, Н. В. Решетникова. Красноярск: ИПЦ КГТУ, 2007. 154 с.

25. Галушкин А. И. Нейронные сети: основы теории. – М.: Горячая линия. Телеком, 2012. – 496 с.: ил. ISBN 978-5-9912-0082-0

26. R. M. Zulunov. // “Sun’iy intellekt texnologiyalari” // 2024-175b. ISBN: 978-9910-783-11-1

27. Python Parallel Programming Cookbook: Master efficient parallel programming to build powerful applications using Python by Giancarlo Zaccone. Published in 2021.

28. Machine Learning for Parallelism and Heterogeneous Computing by Dhabaleswar K. Panda, Xiaoyi Lu, and Wei Ding. Published in 2020.

Elektron manbalar:

https://en.wikipedia.org/wiki/Artificial_intelligence.

https://icenamor.github.io/files/books/Hands-on-Machine-Learning-with-Scikit-2E.pdf

http://cs231n.stanford.edu/

https://www.sciencedaily.com/computers_math/artificial_intelligence

https://en.wikipedia.org/wiki/Artificial_neural_network

http://neuralnetworksanddeeplearning.com/;

www.library.tuit.uz

www.msdn.microsoft.com

www.w3school.com

www.ziyonet.uz

https://www.javatpoint.com/

https://www.geeksforgeeks.org/

SUN’IY INTELLEKT ASOSLARI

Darslik

«JAVOHIR ILM NASHR» nashriyoti,
100129. Toshkent shahri, Navoiy ko‘chasi, 30-uy.

Texnik muharrir:
Javohir Meliyev

Texnik muharrir:
Ma’suma Muhammadjonova

Sahifalovchi-dizayner:
Muhammadiqbol Ismoilzoda

Terishga 16.10.2024-yilda berildi.
Bosishga 27.11.2024-yilda ruxsat etildi.
Bichimi 70x100 1/16
Ofset bosma usuli. 16,5 shart b.t.
Adadi ____ nusxa.

“ELNUR-PRINT” MCHJ bosmaxonasida chop etildi.
Toshkent shahri, Navoiy ko‘chasi, 30-uy.

ISBN: 978-9943-5805-5-8