O‘ZBEKISTON RESPUBLIKASI OLIY TA’LIM, FAN VA INNOVATSIYALAR VAZIRLIGI

SHAROF RASHIDOV NOMIDAGI SAMARQAND DAVLAT UNIVERSITETI

NAZAROV FAYZULLO, YARMATOV SHERZODJON

SUN’IY INTELLEKT ASOSLARI

(Dasturiy injiniring, sun’iy intellekt yo‘nalishlari uchun darslik)

SAMARQAND - 2025

UDK: 681.14(075)
BBK 32.97

Sun’iy intellekt asoslari. **Darslik** –SamDU, 2025 –yil, 181- bet.

Mazkur darslik sun’iy intellekt asoslari fanini o‘rganishga bag‘ishlangan bo‘lib, bunda sun’iy intellekt asoslari fani uchun sun’iy intellekt haqida dastlabki ma’lumotlar, sun’iy intellekt uchun python dasturlash kutubxonalari, mashinaviy o‘qitish, nazoratli va nazoratsiz o‘qitish, regressiya va klassifikatsiya algoritmlari va ularni qo‘llashning qoidalari batafsil keltirib o‘tilgan. Mazkur darslikda sun’iy intellekt va ma’lumotlarga ishlov berish, mashinaviy o‘qitish, mashinaviy o‘qitishning regressiya algoritmlari, mashinaviy o‘qitishda klassifikatsiya algoritmlari, mashinaviy o‘qitishda klasterizatsiya algoritmlari keltirib o‘tilgan. Bu darslik nafaqat pedagogik va muhandis yo‘nalishidagi talabalar, balki mustaqil o‘rganuvchilar uchun ham asosiy qo‘llanma hisoblanadi.Mazkur darslik dasturiy injiniring, sun’iy intellekt yo‘nalishlarining o‘quv rejasi hamda o‘quv dasturi asosida yaratildi. Har bir mavzu bo‘yicha ma’lumotlar nazariy, amaliy qismlardan iborat va mavzu oxirida nazriy savollar to‘plami keltirib o‘tilgan.

Mualliflar

Nazarov Fayzullo Maxmadiyarovich – *Sharof Rashidov nomidagi Samarqand Davlat Universiteti “Sun’iy intellekt va axborot tizimlari“ kafedrasi dotsenti, PhD, dotsent.*

Yarmatov Sherzodjon Shokir o‘g‘li – *Sharof Rashidov nomidagi Samarqand Davlat Universiteti “Sun’iy intellekt va axborot tizimlari“ kafedrasi mudiri, PhD, dotsent.*

Ma’sul muharrir

Agrawal Kant Krishna – *Hindistonning Galgotias Universiteti Computing Science $ Engineering fakulteti professori.*

Taqrizchilar

Sobirov Muslimbek – *Farg‘ona Politexnika Instituti intellektual muhandislik tizimlari kafedrasi dotsenti.*

Axatqulov Sahobiddin – *Sharof Rashidov nomidagi Samarqand Davlat Universiteti “Kompyuter ilmlari va texnologiyalari fanlari“ kafedrasi dotsenti, fizika-matematika fanlari doktori.*

Rashidov Akbar – *Sharof Rashidov nomidagi Samarqand Davlat Universiteti “Sun’iy intellekt va axborot tizimlari“ kafedrasi dotsenti, PhD.*

ISBN 978-9943-

MUNDARIJA

KIRISH

Hozirgi vaqtda barcha sohalarni raqamlashtirish bilan bir qatorda sun’iy intellekt mexanizmlarini avtomatlashtirilgan tizimlarga joriy qilish dolzarb masalalardan biri bo‘lib bormoqda. Sun’iy intellektni jadallik bilan rivojlanishi, ta’lim sohasida ham axborot texnologiyalari va sun’iy intellektning o‘rni keskin ortib borayotganligini ko‘rsatib bormoqda. Hozirda raqamli iqtisodiyot va raqamli texnologiyalarning asosiy bo‘g‘ini hisoblangan ma’lumotlar tahlilini rivojlantirish bugungi kunning dolzarb masalasi hisoblanadi. Bugungi kunda ma’lumotlar oqimining ko‘pligi tufayli ularni qisqa vaqt ichida qayta ishlash muommosi ham ortib bormoqda. Sun’iy intellekt algoritmlarini rivojlantirish, ma’lumotlarni statistik tahlil qilish, oldindan intellektual qayta ishlash kabi muammolarni hal etishning yechimlaridan biri hisoblanadi. Zamonaviy jamiyatda tobora o‘sib borayotgan axborot oqimi, axborot texnologiyalarining turlitumanligi, kompyuterda yechiladigan masalalarning murakkablashuvi, ushbu texnologiyalardan foydalanuvchining oldiga bir qator vazifalarni qo‘ydi. Bu vazifalarni bosqichma bosqich raqamlashtirish va sun’iy intellekt tizimlari orqali hal etish mumkin. Sun’iy intellekt tizimlarini yaratish va joriy qilish uchun bu tizimlarni ishlab chiqishda mashinaviy va chuqur o‘qitish algoritmlarini rivojlantirish kerak bo‘ladi. Sun’iy intellekt tizimlari asosida ma’lumotlarni intellektual qayta ishlash oldindan muammoni yechimlarini topish imkoniyatini yaratadi. Mazkur darslikda sun’iy intellekt va ma’lumotlarga ishlov berish, mashinaviy o‘qitish, mashinaviy o‘qitishning regressiya algoritmlari, mashinaviy o‘qitishda klassifikatsiya algoritmlari, mashinaviy o‘qitishda klasterizatsiya algoritmlari keltirib o‘tilgan. Bu darslik nafaqat pedagogik va muhandis yo‘nalishidagi talabalar, balki mustaqil o‘rganuvchilar uchun ham asosiy qo‘llanma hisoblanadi. Sun’iy intellekt uchun dasturlashni python tili va uning kutubxonalaridan foydalanish asosida batafsil keltirib o‘tilgan. Darslikda sun’iy intellekt va ma’lumotlarga ishlov berishning nazoratli va nazoratsiz mashinaviy o‘qitish algoritmlari keltirilgan. Mazkur darslik dasturiy injiniring, sun’iy intellekt yo‘nalishlarining o‘quv rejasi hamda o‘quv dasturi asosida yaratildi. Har bir mavzu bo‘yicha ma’lumotlar nazariy, amaliy qismlardan iborat va mavzu oxirida nazariy savollar to‘plami keltirib o‘tilgan.

I-BOB. SUN’IY INTELLEKT VA MA’LUMOTLARGA ISHLOV BERISH

Mazkur bobda sun’iy intellekt haqida umumiy ma’lumotlar, python dasturlash tili kutubxonalari asosida sun’iy intellekt tizimlarini ishlab chiqish, Sun’iy intellekt tizimlarida daslabki ma’lumotlarga ishlov berish va sun’iy intellekt tizimlarini ishlab chiqishda ma’lumotlarni vizualizasiya qilish usullari keltirib o‘tilgan.

1.1. Kirish. Sun’iy intellekt haqida umumiy ma’lumotlar

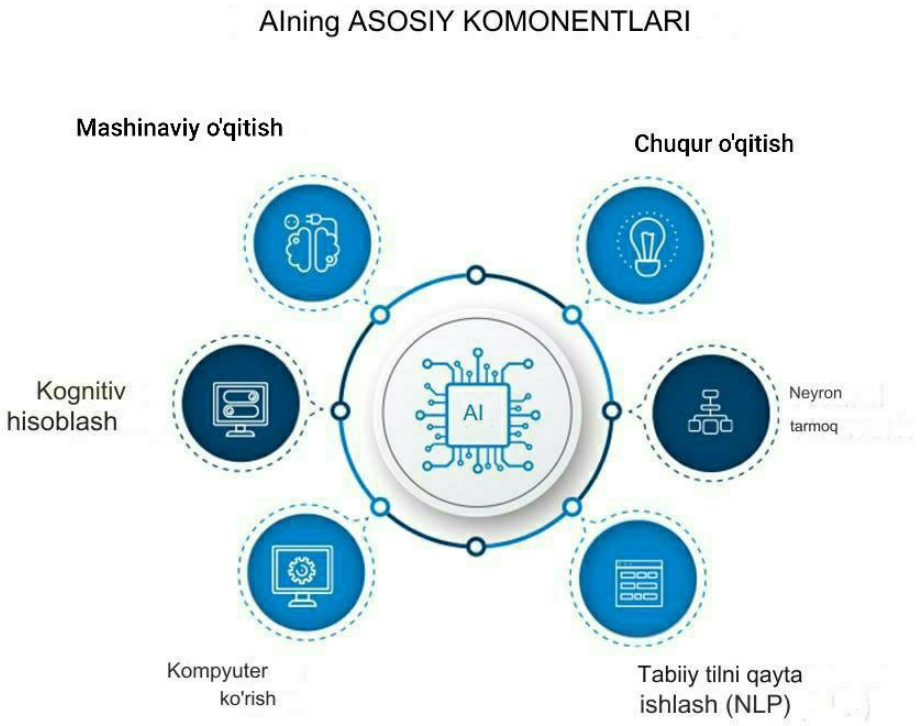
Reja:

Sun’iy intellekt yo‘nalishlari va imkoniyatlari;

Sun’iy intellektning rivojlanish tarixi;

Sun’iy intellekt inson intellektiga qarshi.

Sun’iy intellekt — informatikaning alohida sohasi bo‘lib, odatda inson ongi bilan bog‘liq imkoniyatlar: vizual idrok etish, qaror qabul qilish, nutqni anglash, o‘rgatish, muhokama qilish, masalani yechish, tarjima, bashorat qilish va shu kabi odatda inson aql-zakovatini talab qiladigan vazifalarni bajarishga qodir kompyuter tizimlarini yaratish bilan shug‘ullanadi. Zamonaviy sun’iy intellektlar mashinaviy o‘qitish algoritmlari yordamida yaratiladi. Mashinaviy o‘qitish – kompyuterlarni dasturlamagan holda o‘z-o‘zini o‘qitish imkoniyatini berishga xizmat qiladi. Sun’iy intellekt (SI) bizga nima uchun zarurligini sabablari va yo‘nalishlari quyidagi sxema orqali keltirilgan.



Sun’iy intellektning yo‘nalishlari va imkoniyatlari

1. Sun’iy intellekt xizmatlarining “beminnatligi” va vaqt meyorlarining cheklanmaganligi bilan foydali hisoblanadi.

2. Sun’iy intellekt buzilishlar va toyilishlarga moslasha oladi, Amerikalik olimlar sun’iy intellekt bilan jihozlangan robot bilan tajriba o‘tkazganda, u jiddiy zarar ko‘rgan taqdirda ham ishlashda davom etishini aniqladilar. Tajriba davomida “jarohat olgan” robot kamida oltita turli xil jarohatlarga, shu jumladan ikkita pastki oyoq-qo‘llarining to‘liq yo‘qolishiga moslasha oldi va robotning “qo‘li” kamida 14 turdagi jarohatlarga, shu jumladan uning ikkita dvigatelining ishdan chiqishiga moslasha oldi.

3. Sun’iy intellekt yaratuvchilarining e’tiqodi va stereotiplarini meros qilib olishga mo‘ljallangan, hisoblanadi. Sun’iy miya o‘z xulosalarini dastlab unga kiritilgan ma’lumotlar asosida chiqaradi. Tadqiqotlar shuni ko‘rsatdiki, yuzni tanib olish uchun ba’zi kompyuter tizimlari 35% hollarda qora tanli ayollarning jinsini chalkashtirib yuborgan va oq tanli erkaklarning atigi 0,8%ni chalkashtirgan. Buning sababi shundaki, sun’iy intellekt ishlaydigan ma’lumotlar bazalaridagi fotosuratlari 75% erkaklar tashkil etadi, ularning 80% oq tanli fotosuratlardan iborat bo‘lgan.

4. Sun’iy intellekt berilgan savollarga javob berishi mumkin, bugungi kungacha eng kuchli sun’iy intellekt bilan ishlaydigan matn ishlab chiqaruvchisi - OpenAI-dan GPT-2 butun xatboshilarini yozishi mumkin va xatolarga yo‘l qo‘ymaydi. Shu bilan birga, tizim umumiy bilimlarga tegishli bo‘lsa, savollarga to‘g‘ri javob beradi.

5. Sun’iy intellekt inson qila oladigan hamma narsani o‘rganishga qodir hisoblanadi. Tadqiqotchilarning fikriga ko‘ra sun’iy intellekt 2060 yilga kelib insonning deyarli barcha vazifalarini mustaqil ravishda bajara olishiga umid qilishmoqda. Masalan, Oksford universiteti olimlari **Googlening DeepMind** SI bo‘limi bilan hamkorlikda tizimni odamlarga qaraganda labda o‘qishni yaxshi o‘rgatishdi. **Watch, Attend and Spell** dasturi xuddi shunga o‘xshash lablar harakati bilan so‘zlar orasidagi farqni aniqlaydi va 50% gacha jim nutqni tahlil qila oladi. Lab harakatlarni o‘qigan mutaxassislar uchun xuddi shu ko‘rsatkich atigi 12%

ni tashkil qiladi. Tizim BBC yangiliklar dasturlarini tomosha qilish orqali oʻqitildi, unda videodan 118000 ta jumlani oʻrganib chiqqandan soʻng, 17,500 dan ortiq soʻzlardan iborat gaplarni tahlil qilib chiqdi.

6.Sun’iy intellektga ega robotlar allaqachon diktor boʻlib ishlaydi, kosmosga uchadi, kemalarda patrullik qiladi va futbol oʻynaydi. Xitoyning Sinxua davlat axborot agentligida robotlashtirilgan diktor yangiliklarni oʻqiydi. U Chjan Vanveyning hayotiy prototipi asosida yaratilgan. Robot nafaqat yangiliklar matnlarini oʻqiy oladi, balki ularning mimikasi va nutq uslubiga taqlid qilib, inson hamkasblaridan oʻrganishi mumkin. **CIMON 2 roboti XKS**da kosmonavtlar bilan aloqa qiladi: u oʻzining sun’iy intellekti sifatida **Watson IBM** tizimidan foydalanadi. Watson Tone Analyzer xizmati bilan yangilanish CIMON 2-ga odamlarning his-tuygʻularini tushunishga va ularga javob berishga imkon beradi. CIMON loyihasi Germaniyaning Aerokosmik markazi tomonidan Airbus va IBM bilan hamkorlikda ishlab chiqilgan. Norvegiyaning Aker BP neft kompaniyasi kemalaridan birini qoʻriqlash uchun Spot (Boston Dynamics tomonidan ishlab chiqilgan) deb nomlangan robot itidan foydalanadi. Zamonaviy robotlar hatto futbol oʻynashni ham bilishadi: bunday modellar Berlinning bepul universiteti qoshidagi sun’iy intellekt guruhida yaratilgan.

7. Sun’iy intellekt hattoki koronavirus bilan kurashishda ham yordam beradi. Dunyo boʻylab sun’iy intellektga asoslangan tizimlar virus yuqtirgan odamlarni kuzatishda, virus haqida ma’lumot toʻplashda va vaksinani qidirishda yordam beradi. Masalan, Isroilning **Vocalis Health** kompaniyasi Isroil hukumati bilan hamkorlikda ovozli spektr tahliliga asoslangan COVID-19 kasalligini aniqlash texnologiyasini ishlab chiqdi. Bundan tashqari, sun’iy intellektli robotlar jamoat joylarini patrul qilishda foydalaniladi (Singapur). Megvii ReID texnologiyasi yordamida Xitoyda inson oqimida yuqori isitma bilan kasallanganlarni aniqlaydigan tizim ishlab chiqildi.

8. Sun’iy intellekt sayyorani tejash va odamlarni oziq-ovqat bilan ta’minlashni intellectual tahlil qilish mumkin. Amerika Qoʻshma Shtatlari, Kanada va Lotin Amerikasida bioxilma-xillikni saqlashni yoritishga bagʻishlangan "NatureServe" notijorat tashkiloti SAS analitik kompaniyasi bilan global Data for Good tashabbusi doirasida hamkorlik qildi. SI oʻsimlik va hayvon turlari toʻgʻrisidagi ma’lumotlarni toʻplash,

ularning joylashuvi va populyatsiyalarning kontsentratsiyasini aniqlash uchun ishlatiladi. Birlashgan Millatlar Tashkilotining Oziq-ovqat va qishloq xo‘jaligi tashkiloti (FAO) ham sun’iy intellektning afzalliklarini tan oladi: ular "razvedka", ob-havo sharoiti, zararkunandalar, tuproq namligi va boshqa muhim ko‘rsatkichlar haqidagi ma’lumotlarni hisobga olgan holda, fermerlarga ishni yanada samarali rejalashtirishga imkon berishiga ishonadilar.

Sun’iy intellektning rivojlanish tarixi

Sun’iy intellekt atamasi 1956 yilda paydo bo‘lgan, ammo bugungi kunda SI texnologiyasi ma’lumotlar hajmini ko‘paytirish, algoritmlarni takomillashtirish, hisoblash quvvatini va ma’lumotlarni saqlash vositalarini optimallashtirish fonida haqiqiy mashhurlikka erishdi.

O‘tgan asrning 50-yillarida boshlangan SI sohasidagi birinchi tadqiqot-muammolarni hal qilish va ramziy hisoblash tizimlarini rivojlantirishga qaratilgan edi. 60-yillarda bu sohada AQSh Mudofaa vazirligi qiziqish uyg‘otdi: AQSh harbiylari insonning aqliy faoliyatini simulyatsiya qilish uchun kompyuterlarni o‘qitishni boshladi.

Masalan, mudofaa vazirligining ilg‘or tadqiqot loyihalari agentligi (DARPA) 1970-yillarda bir qator virtual ko‘cha xaritalari loyihalarini yakunladi va DARPA mutaxassislari Siri, Alexa va Cortana paydo bo‘lishidan ancha oldin 2003 yilda aqlli shaxsiy yordamchilarni yaratishga muvaffaq bo‘lishdi. Ushbu ishlar zamonaviy kompyuterlarda, xususan, qarorlarni qo‘llab-quvvatlash tizimlarida va inson imkoniyatlarini kengaytirish uchun ishlab chiqilgan aqlli qidiruv tizimlarida qo‘llaniladigan avtomatlashtirish va rasmiy mantiqiy tamoyillar uchun asos bo‘ldi. Garchi SI ko‘pincha ilmiy fantastika filmlari va romanlarida ilmiy qudratli robotlar sifatida tasvirlangan bo‘lsa-da, dunyo miqyosida o‘z kuchini egallagan, SI texnologiyasini rivojlantirishning hozirgi bosqichida, SIlar unchalik qo‘rqinchli va aqlli emaslar. Aksincha, sun’iy intellektni rivojlantirish ushbu texnologiyalarga iqtisodiyotning barcha sohalarida haqiqiy foyda keltiradi.

Sun’iy intellekt inson intellektiga qarshi

Insonlar kabi fikrlay oladigan mashina yasash g‘oyasi fantastika olamidan real dunyoga yetib keldi. Hozirda ilgari imkonsiz bo‘lgan vazifani robotlar bajara oladi. Sun’iy intellekt haqida g‘oyalar paydo

bo‘lgandan beri insonlar sun’iy intellekt va inson aqlini solishtirib kelmoqda. Quyida asosiy farqlar bilan tanishamiz. Oddiy farq shundaki, inson o‘z miyasidan, fikrlash qobiliyatidan, xotirasidan foydalanadi, SI mashinalari esa ularga berilgan ma’lumotlarga bog‘liq bo‘ladi. Zamonaviy kompyuterlar odatda 2 vatt energiya sarflaydi, inson miyasi esa taxminan 25 vatt energiya sarflaydi. Mashinalar odamlarga qaraganda ko‘proq ma’lumotni tezroq ishlay oladi. Hozircha odamlar kompyuter tezligini yengib bo‘lmaydi.

Inson tafakkuri va SI tizimini solishtirish quyidagicha solishtirish mumkin.

Inson tafakkur tizimi	*SI tizimi*
<u>Ustunliklari</u> 1. Ijod qiluvchi 2. Moslashuvchan 3. Hissiy idrokdan foydalanadi 4. Har tomonlama 5. Keng qamrovli bilimdan foydalanadi	<u>Kamchiliklari</u> 1. Sun’iy oldindan dasturlashtirilgan 2. Aytib turish kerak 3. Belgili idrokdan foydalanadi 4. Tor yo‘nalishli 5. Maxsus bilimdan foydalanadi

Bu tizimlarni afzalliklari va kamchiliklarini tahlil qilib, inson ekspert asosiy afzalliklari, u ko‘p sohada, masalan, ijodkorlikda, topqirlikda,ma’lumot uzatishda va umuman mazmunan SIdan ustunlikka ega bo‘ladi.

Amaliy topshiriq

1. Sun’iy intellektning har bir yo‘nalishi bo‘yicha tahlil o‘tkazish.

2. Sun’iy intellektning ijobiy va salbiy tomonlarini aniqlash bo‘yicha loyiha tayyorlash.

Nazorat uchun savollar

1.Sun’iy intellekt va uning imkoniyatlari?

2.Sun’iy intellektning asosiy yo‘nalishlari va vazifalari?

3.Inson intellektining sun’iy ongdan asosiy farqlari?

4.Sun’iy intellekt tarixi?

5.Sun’iy intellekt texnologiyalariga misollar keltiring.

6.Sun’iy intellekt bilan inson qanday yutuqlarga erishadi?

1.2. Sun’iy intellekt tizimlari va texnologiyalari

Reja:

Sun’iy intellekt qo‘llaniladigan ba’zi sektorlar;

Sun’iy intellekt tizimlari va texnologiyalari;

Teachable Machine.

Barchamizga ma’lumki, sun’iy intellekt bizning kundalik hayotimizda muhim o‘rin tutadi va bugungi jamiyatda uning ko‘plab ilovalari mavjud. Sog‘liqni saqlash, qishloq xo‘jaligi, moliya, ta’lim va boshqa ko‘plab sohalarda SI murakkab masalalarni ijodiy hal qila oladi.

Sun’iy intellekt qo‘llaniladigan ba’zi sektorlar

1. **Sog‘liqni saqlashda SI**. SI sog‘liqni saqlash sohasida juda ko‘p foyda keltirmoqda. Bunday texnologiyalar inson omilisiz tezroq va yaxshiroq tashxis qo‘yishga yordam beradi, shuningdek, inson xatolarini kamaytirishga yordam beradi. SI sog‘liqni saqlash sohasida ko‘plab usullarda qo‘llaniladi, masalan, sun’iy intellekt yordamida robotli jarrohlik, ish jarayoni va ma’muriy vazifalarda yordam beradi va yana bir qancha jarayonlarni amalga oshiradi. Tibbiyotda sun’iy intellekt inson tanasidagi ma’lumotlarga asosan kasalliklarni tashxislash va prognozlash imkoniyatini beradi.

2. **Ijtimoiy tarmoqlarda SI**. FB, Snapchat, Twitter, Telegram kabi ijtimoiy tarmoqlarda milliardlab foydalanuvchi profillari mavjud va bundagi ma’lumotlarni sifatli saqlash kerak va SI buni optimal boshqarishga xizmat qiladi. SI ijtimoiy tarmoqdagi ma’lumotlarni intellektual tahlil asosida boshqarish imkoniyatini beradi.

3. **Robototexnika sohasida SI**. Robototexnika sohasida SI katta rol o‘ynaydi. Ba’zi robotlar ularga berilgan takroriy vazifalarni bajarishi mumkin, ammo SI yordamida ular o‘ylaydigan va vazifalarni bajara oladigan robotlarni yaratishi mumkin.

4. **Iqtisodiyotda SI**. Iqtisodiyot sektorida SI iqtisodiy ma’lumotlar asosida iqtisodiy ko‘rsatkichlarni prognozlashni amalga oshirishga xizmat qiladi.

5. **Sayohatda SI**. SI sayohat sanoatida ham juda zarur. Sayohat sektorlari yaxshi javob olish uchun mijozlar bilan insoniy muloqotni yaratishi mumkin bo‘lgan sun’iy intellektga asoslangan chatbotlardan foydalaniladi.

6. **Ma’lumotlar xavfsizligi sohasida SI**. Ma’lumotlar har qanday kompaniya uchun juda muhim va bugungi kunda bu raqamli dunyoda firibgarlik holatlari juda yuqori. Shunday qilib, SI ma’lumotlaringizni xavfsizroq qilishga yordam beradi. Bunda SI ma’lumotlardan foydalanish chegaralarini belgilashga xizmat qiladi.

7. **Avtomobil sanoatida SI**. Yaxshiroq ishlash uchun ba’zi avtomobil sanoati o‘z foydalanuvchilariga haqiqiy yordamchilarni berish uchun sun’iy intellektdan foydalanadi.

8. **Qishloq xo‘jaligida SI** . SI qishloq xo‘jaligi sohasini raqamlashtirish va prognozlash imkoniyatini yaratadi.

9. **Ta’limda SI**. Sun’iy intellekt chatbotlari o‘quvchilar bilan o‘quv yordamchilari shaklida muloqot qilishlari mumkin. Kelajakda SI ta’lim sohasida yanada ko‘proq rivojlanishni namoyish etadi. O‘quv ko‘rsatkichlarini tahlil qilish asosida prognozlash masalalari yechiladi.

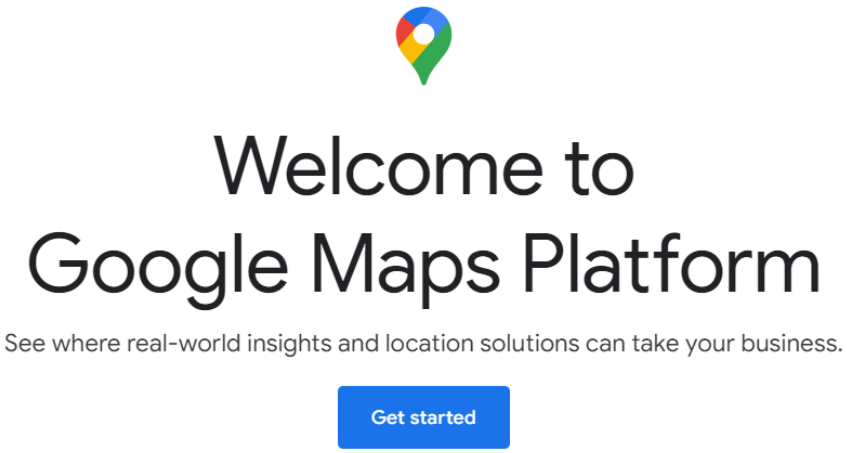
10. **Elektron tijoratda SI.** Sun’iy intellekt xaridorlarga tavsiya etilgan rang, o‘lcham yoki brend bilan tegishli tovarlarni topishda yordam beradi. SI elektron tijorat sohasida yanada talabchan bo‘lib bormoqda .

11. **O‘yinda SI**. SI deyarli barcha sohalarda mavjud bo‘lib, shulardan o‘yin texnologiyalaridagi qadamlarni tashkil qilish va yangilash imkoniyatini beradi.

Hozirgi vaqtda sun’iy intellektni rivojlanishi bilan bir qatorda bir qancha sun’iy intellekt tizimlari ishlab chiqilgan. Hozirda keng qo‘llaniladigan ba’zi sun’iy intellekt tizimlar va texnologiyalariga quyidagi misollarni keltirish mumkin.

Sun’iy intellekt tizimlari va texnologiyalari

1. Google xaritalari. Ilgari insonlar yangi joyga borishda manzillarni aniqlash muammosini hal etolmasdilar. Ammo endi insonlar yangi manzilga sayohat qilish haqida ko‘p o‘ylashi shart emas. Buning uchun Google xaritasi bo‘lgan bitta ilovani ochish kifoya bo‘ladi va u manzilga yetib borish uchun yo‘l ko‘rsatib beradi. Endi sun’iy intellekt foydalanuvchilarga o‘zlarining atrof-muhitlari haqida ko‘proq tanishish imkonini berish orqali yordam beradi. Bu bizning yurishimizni osonlashtiradi, chunki u bizga tirbandlik, qolgan masofa va marshrutning ovozli usuli haqida taklif qiladi.



2. **Aqlli avtomobillar.** Aqlli avtomobillar AI ahamiyatini oshirayotgan sohalardan biridir. Nafaqat Tesla kabi yirik kompaniyalar, balki boshqa ko‘plab kompaniyalar o‘z avtomobillarida avtomatlashtirishdan foydalanmoqda.

3. **Smartfonlar**. Smartfonlarning hayot tarzimizdagi ahamiyatini aytib o‘tishning hojati yo‘q, chunki biz hayotimizni smartfonlarsiz tasavvur qila olmasligimizni hamma biladi. Bilasizmi yoki yo‘qmi, siz SI bilan o‘zaro aloqadasiz. Biz bilamizki, Google Assistant, Siri bizning telefonimizdagi sun’iy intellektning namunalari hisoblanadi.

4. **Avtomatik tuzatish**. Bu tizimda imlo xatolar tuzatiladi, matnda so‘zlar va imlolarni, grammatikani taklif qiladi. Imlo, xatolar va grammatikani avtomatik ravishda tuzatadigan mashinalar mavjud.

5. **Chatbotlar**. Sun’iy intellekt paydo bo‘layotgan yana bir soha - bu Chatbotlar hisoblanadi. Mijoz sifatida biz har doim savollarimizga imkon qadar tezroq javob berishni xohlaymiz. SI bunda mashinalarga mijozlar bilan qanday gaplashish kerakligini tushuntirish orqali yordam beradi.

6. **Video O‘yinlar.** Video o‘yinlar sanoati, ehtimol, sun’iy intellektning dastlabki qabul qiluvchilaridan biridir. Poyga o‘yinlarini o‘ynab, sizga raqib kim ekanligi haqida hech o‘ylab ko‘rganmisiz? Bu sizga qarama-qarshi o‘ynaydigan SI botlari hisoblanadi. Shunday qilib, sun’iy intellekt video o‘yinlar sanoatidagi o‘z o‘rnini egallab turibdi.

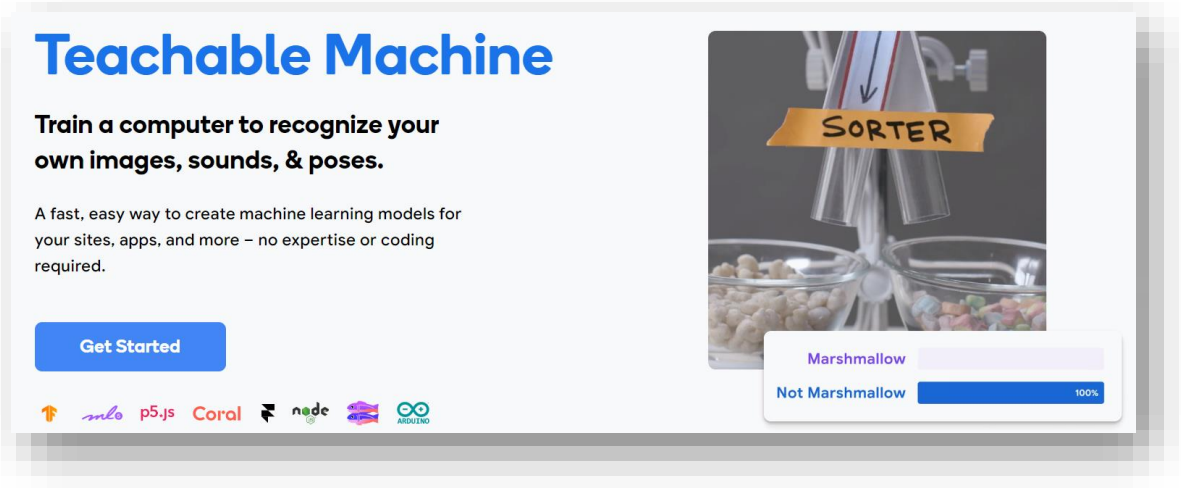
7. **Yuzni tanish**. “Yuzni tanish ” funksiyasining bu xususiyati hammaga ma’lum SI bizning smartfonlarimizni shunchaki yuzimizni ko‘rsatish orqali ochishga imkon beradi. Aqlli mashinalar bizning yuzimizga mos keladi va keyin telefonlarimizni ochadi. Ha, bu qiziq va SI bizga shunga o‘xshash bir nechta xususiyatlarni taqdim etadi.

8. **Ijtimoiy tarmoqlar tasmalari**. Agar siz ijtimoiy tarmoqlardan foydalansangiz, qarorlaringizning aksariyati SI tomonidan qabul qilinadi. Xronologiyada ko‘rgan tasmalardan tortib, ushbu ilovalardan olgan bildirishnomalargacha hammasi SI tomonidan tartibga solinadi. O‘tgan qidiruvlaringiz va o‘zaro ta’sirlaringizdan SI siz qidirayotgan narsani his qiladi va tavsiyalarni ko‘rsatadi. SIning asosiy maqsadi sizni ularning veb-saytlariga qaytib kelishingizga qaram qilishdir.

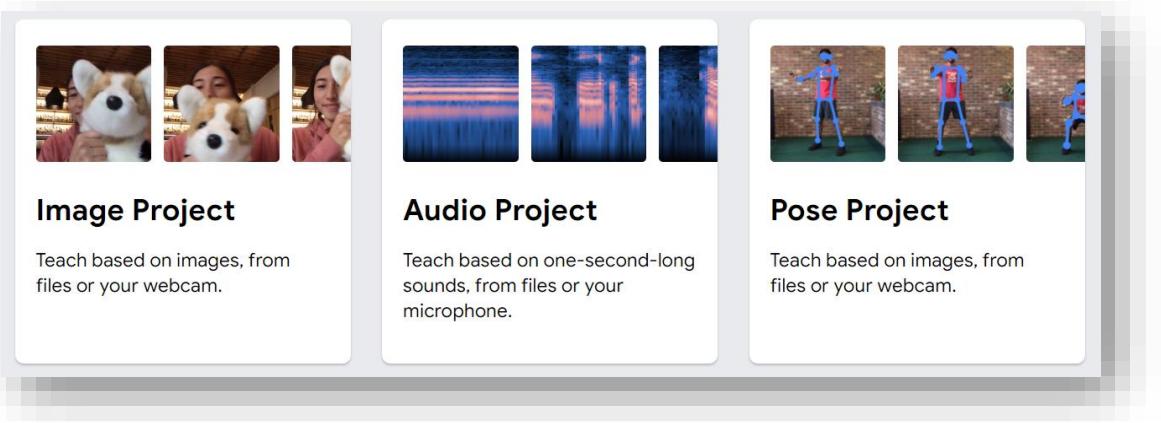
9. **Smart elektron pochta ilovalari**. Zamonaviy elektron pochta ilovalari avvalgilaridan juda farq qiladi. Spark kabi ilovalar spam-xabarlardan xalos bo‘lish uchun sun’iy intellektdan maksimal darajada foydalanadi. Bundan tashqari, u elektron pochta xabarlarini toifalarga ajratadi, shunda siz darhol muhimroqlariga kirishingiz mumkin.

Teachable Machine

Teachable Machine - bu vebga asoslangan vosita bo‘lib, u mashinaviy o‘qitish modellarini tez, oson va hamma uchun ochiq qiladi.

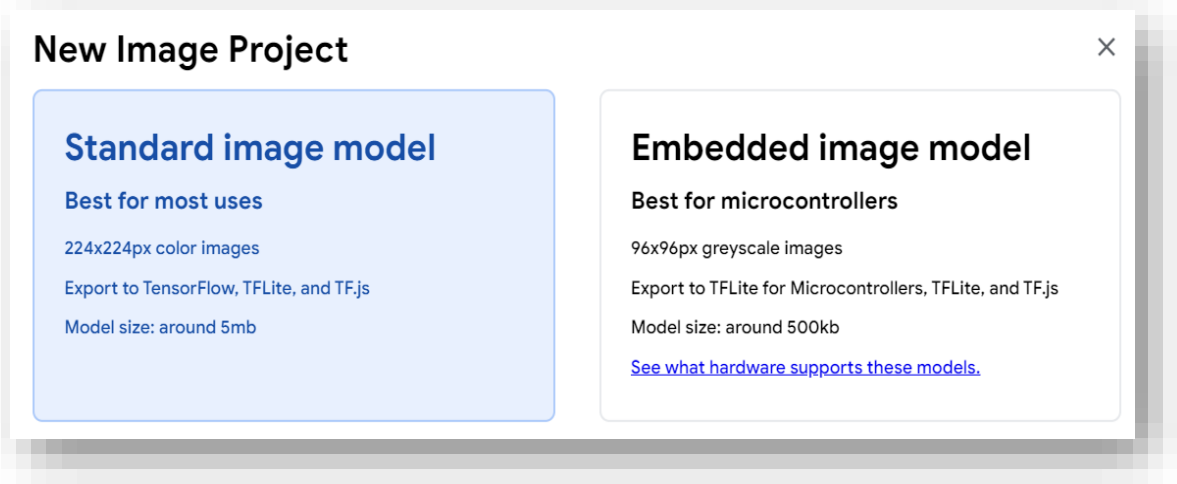


O‘zingizning rasmlaringiz, tovushlaringiz va pozalaringizni tanib olish uchun kompyuterni o‘qitish mumkin. Saytlar, ilovalar va boshqalar uchun mashinaviy o‘qitish modellarini yaratish tez va oson usuli – hech qanday tajriba yoki kodlash talab etilmaydi. Teachable Machine yordamida 3 turdagi sun’iy intellektlarni yaratish mumkin.

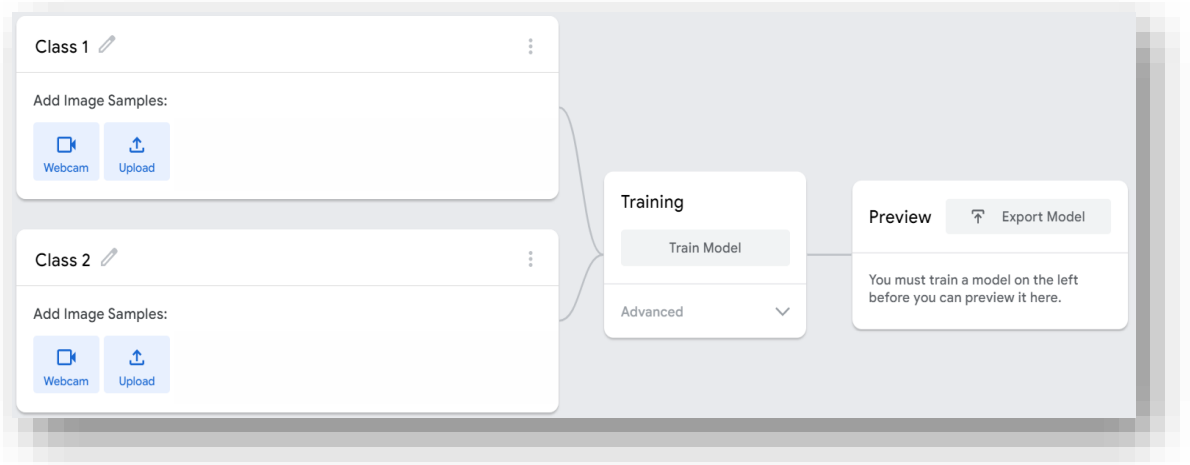


- *Image Project (Rasmlar bilan ishlaydigan sun’iy intellekt)*
- *Audio Project (Ovozlar bilan ishlaydigan sun’iy intellekt)*
- *Pose Project (Harakatlar bilan ishlaydigan sun’iy intellekt)*

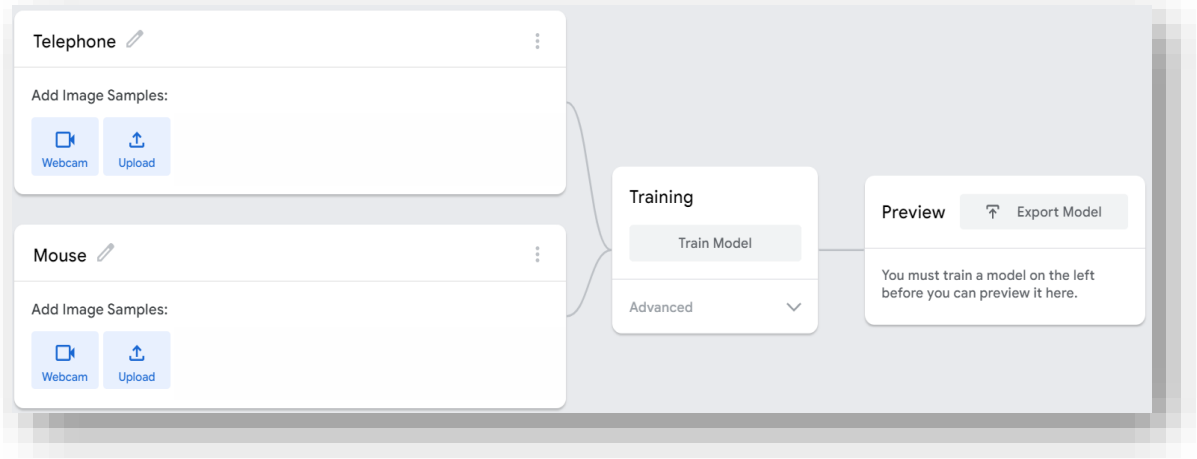
I. Image Project. Bu bo‘limda rasmlarni klassifikatsiya qilish imkoniyati mavjud. 1. Dastlab **New Image Project** bo‘limi tanlanadi. Telefon va sichqonchani klassifikatsiya qiladigan Image project yaratishni qarab o‘tamiz.



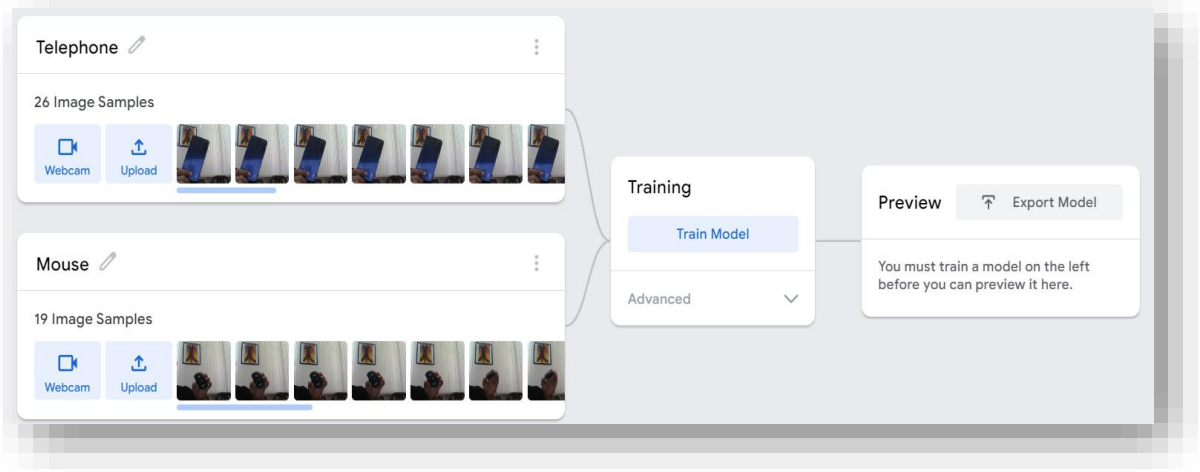
2. Bizda nechta obyekt bo‘lsa shuncha klass yaratiladi.



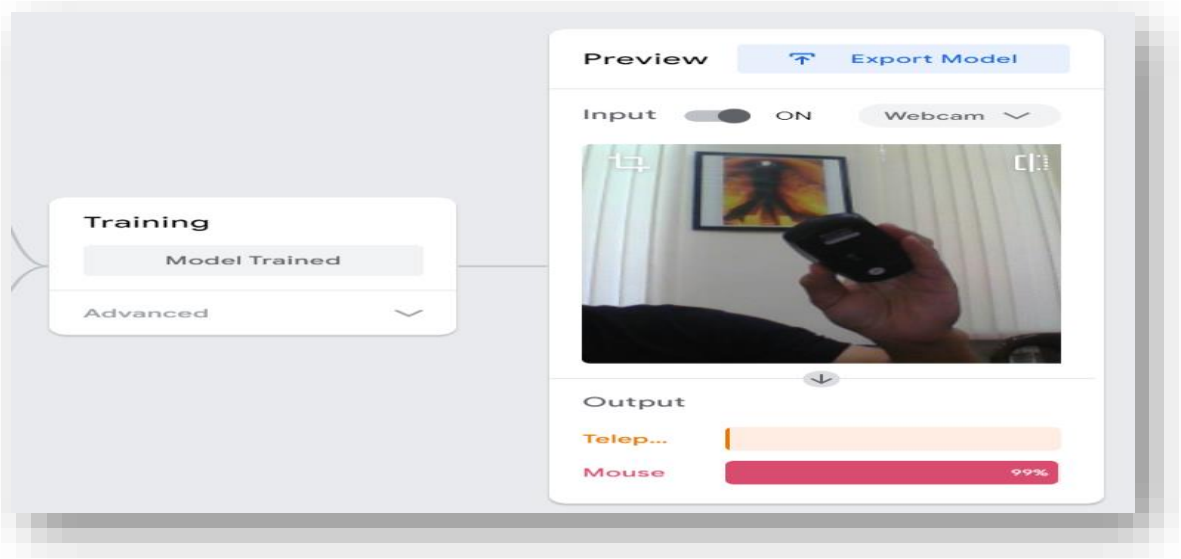
3. Telephone va mouse klasslarini yaratamiz.



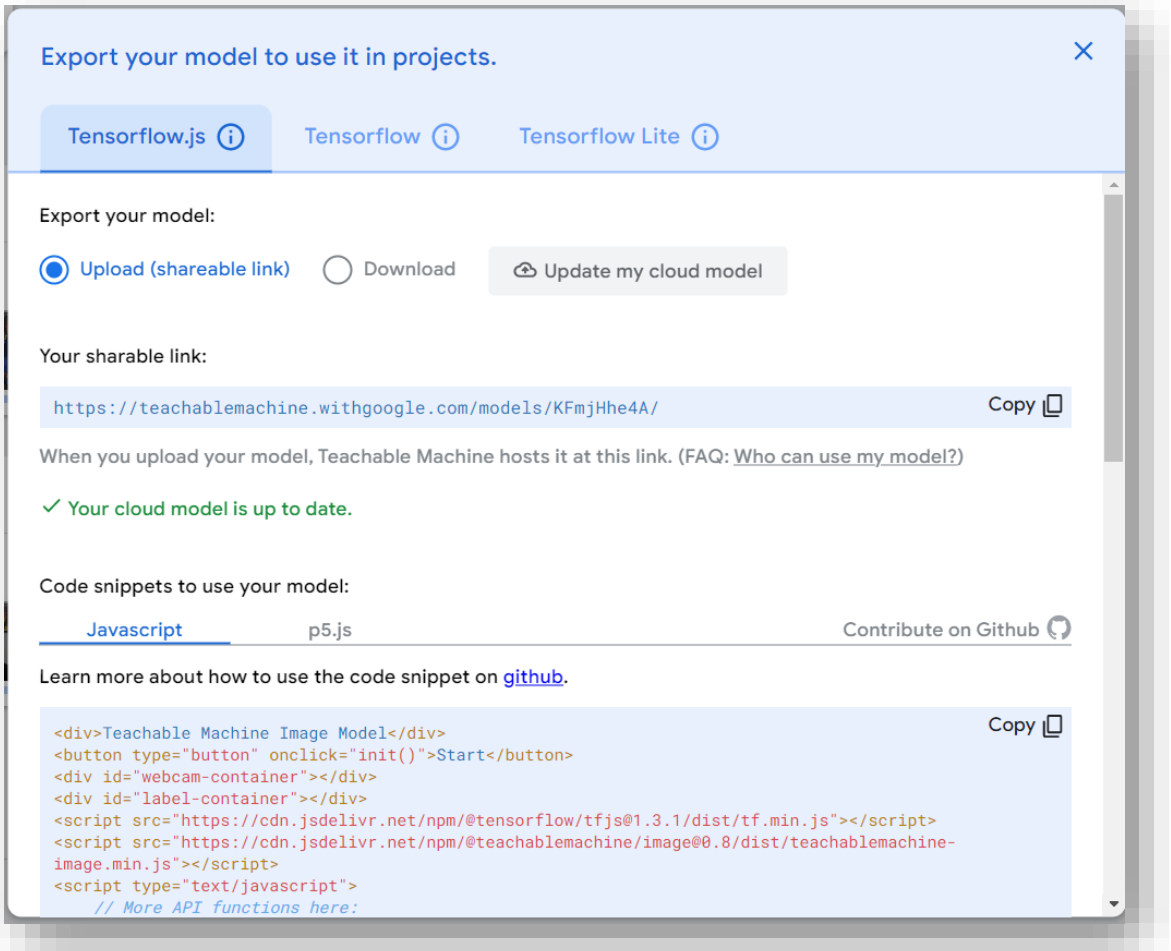
4. Obyekt haqida kerakli rasmlarni yuklaymiz , bunda ikki xil yo‘l bor “webcam” yoki “upload”.



5. Rasmlarni yuklab bo‘lgandan so‘ng “**Train Model**” faollashtiriladi. Natijani bilish uchun kompyuterimiz web camerasiga kerakli obyektni ko‘rsatishimiz mumkin.



6. “**Export Model**” bo‘limi orqali sihlab chiqilgan modelni kodi bilan tanishishimiz mumkin. .



7. “Update my cloud model” bo‘limi orqali modelimizni linkini yuklab olishimiz mumkin.
https://teachablemachine.withgoogle.com/models/KFmjHhe4A/

Nazorat uchun savollar

1. Sun’iy intellekt qo‘llaniladigan ba’zi sektorlar va ularning vazifasi?
2. Sun’iy intellekt tizimlari va texnologiyalari?
3. Teachable Machine dasturi va uning vazifasini tushuntiring?
4. Sun’iy intellektning yangi texnologiyalariga misollar keltiring?
5.“Google Map”da ishlash afzalliklarini ayting?
6 .Zamonaviy smartfonlarda qanaqa intellektual tizimlar qo‘llanilmoqda?
7. Ijtimoiy tarmoqlarda qanaqa sun’iy intellekt na’munalari qo‘llanilmoqda?

1.3. Python dasturlash tili dasturlari asosida sun’iy intellekt tizimlarini ishlab chiqish

Reja:

Anaconda;
Jupyter Notebook;
Google Colab.

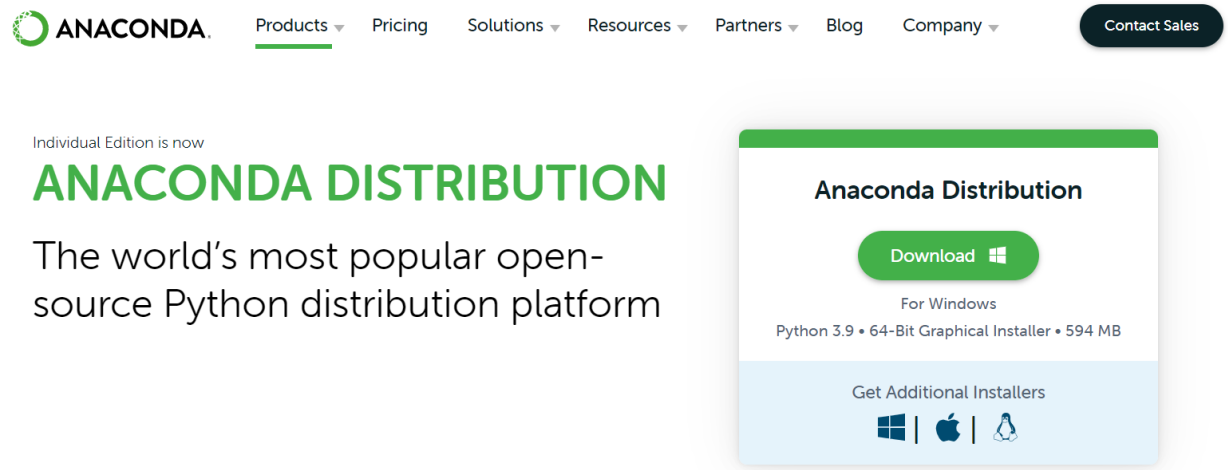
Mazkur mavzuda sun’iy intellekt uchun python dasturlash tili va uning imkoniyatlari hamda kutubxonalaridan foydalanish texnologiyalari keltirib o‘tilgan. Sun’iy intellekt dasturlarini python dasturlash tilida ishlab chiqish uchun avval pythonning kutubxonalaridan foydalanishni o‘rganish kerak bo‘ladi. Sun’iy intellektli python dasturlash tili kodlarini yozish uchun bir nechta kompilyatorlarni ishlatish mumkin, quyida ba’zi kompilyatorlar va ularning imkoniyatlari keltirib o‘tilgan.

Anaconda: Jupyter Notebook

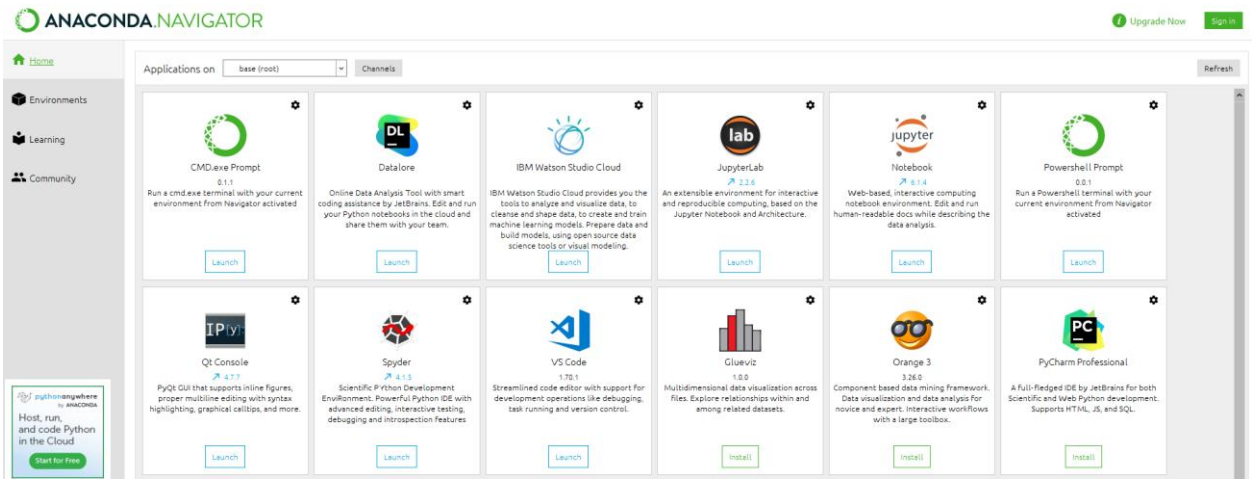
Anaconda komplyatori – bu **Python** va **R** dasturlash tillarining tarqatilishi, jumladan ma’lumotlar tahlili va mashinaviy o‘qiish muammolari bilan birlashtirilgan mashhur bepul kutubxonalar to‘plami hisoblanadi. Bunda quyidagi ishlarni amalga oshirish mumkin.

- Ma’lumotlarni vizuallashtirishdan tortib robototexnikagacha bo‘lgan har qanday sohadagi loyihalar uchun zarur bo‘lgan ochiq kodli dasturiy ta’minot.
- Bu intuitiv platforma yordamida siz osongina paketlarni qidirishingiz va o‘rnatishingiz, shuningdek muhitlarni yaratishingiz, yuklashingiz va almashtirishingiz mumkin.
- Xavfsiz joylashtirilgan paketlar va artefaktlar uslubiy sinovdan o‘tkaziladi va muntazam yangilanadi.

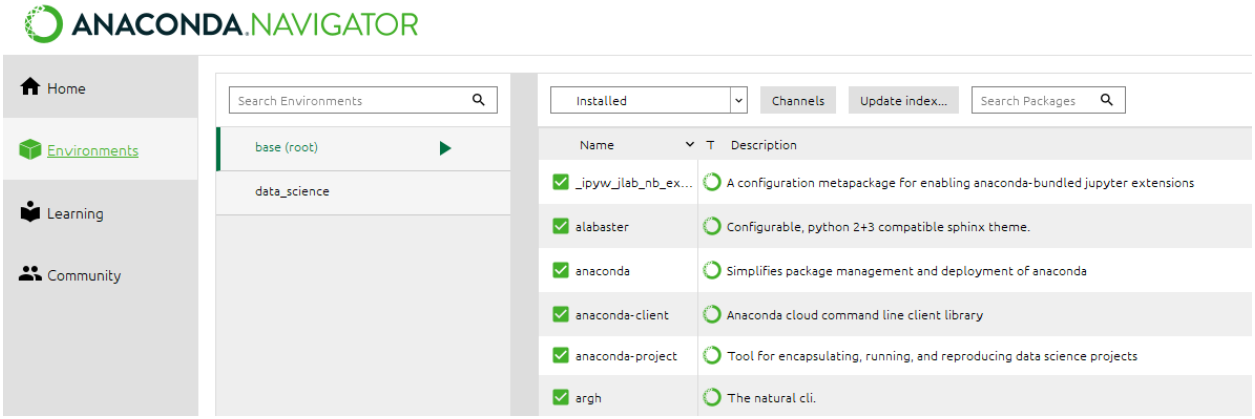
Anaconda dasturidan foydalanish uchun **Anaconda.com** sayti orqali anaconda dasturini yuklab olib, kompyuterga o‘rnatib ishlatish mumkin.



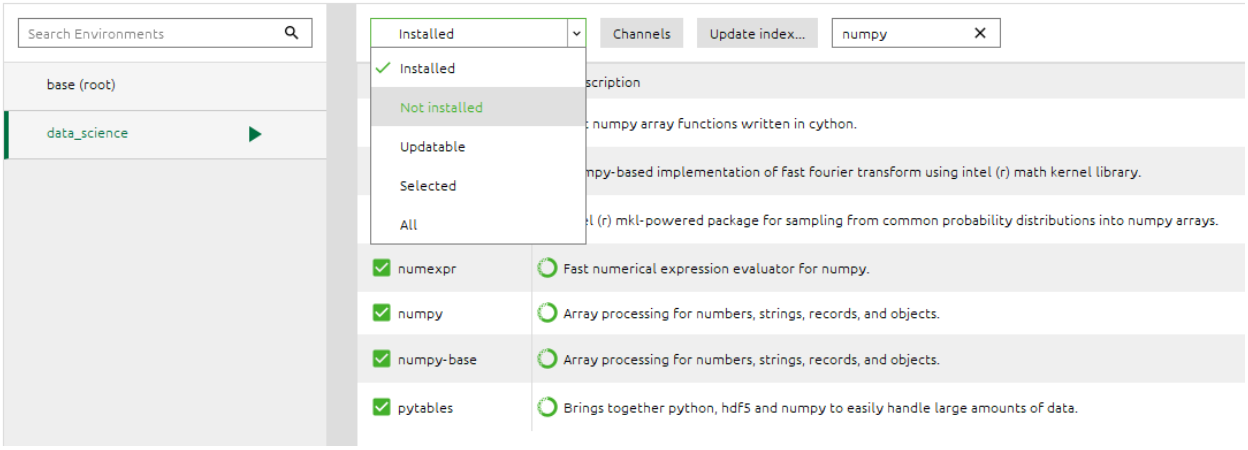
Dasturni o‘rnatgandan so‘ng, quyidagi ko‘rinishda ochiladi.



Bu o‘rnatilgan dastur orqali virtual muhit yaratish va unda ishlashni amalga oshirish mumkin. **Environments** bo‘limi yordamida “**data_science**” nomli virtual muhit yaratib olinadi.



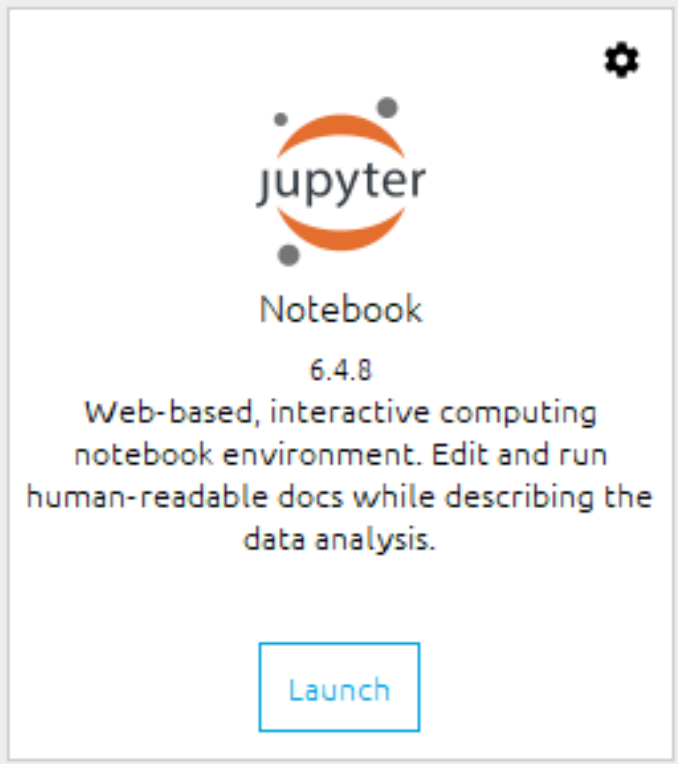
Yaratilgan muhitda bizga kerakli kutubxonalarni o‘rnatib foydalanish mumkin, masalan, Numpy, Pandas kabi kutubxonalar.



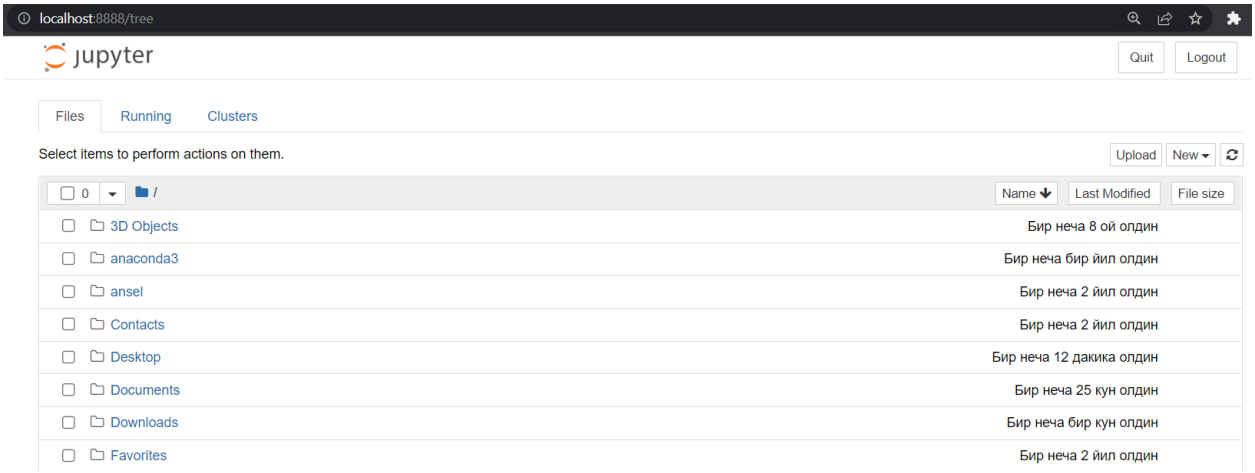
Anaconda dasturi orqali **Jupyter Notebook** xizmati o‘rnatiladi.

Jupyter Notebook

Jupyter Notebook - barcha dasturlash tillarida interaktiv hisoblash uchun bepul dasturiy ta’minot, ochiq standartlar va veb-xizmatlarni mujassamlashtirgan hisoblanadi.

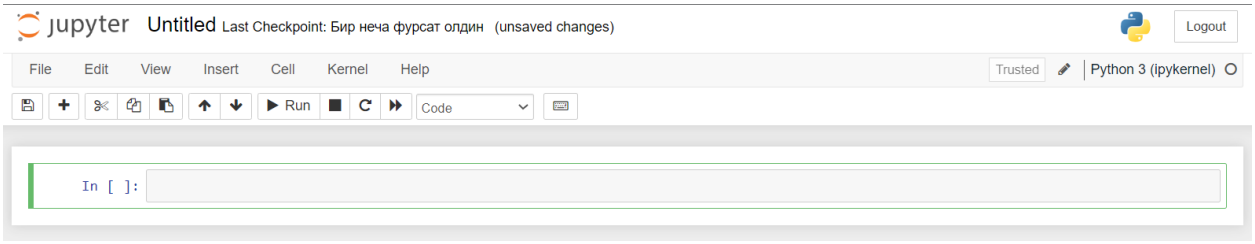


Agar sizning web-brauzeringizda quyidagi ko‘rinishdagi oyna ochilgan bo‘lsa, Jupyter notebook muvaffaqiyatli o‘rnatilgan bo‘ladi.

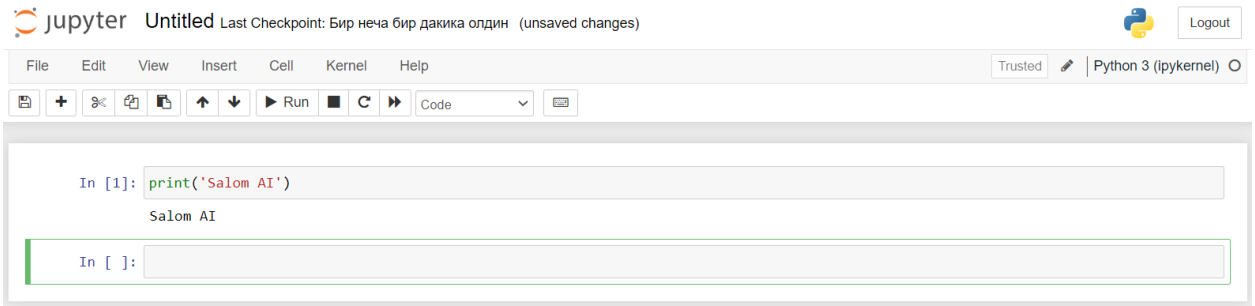


Jupyter Notebookni ishlashini quyidagicha test qilib ko‘rishimiz mumkin. New bo‘limidan “Python 3”ni tanlaymiz.

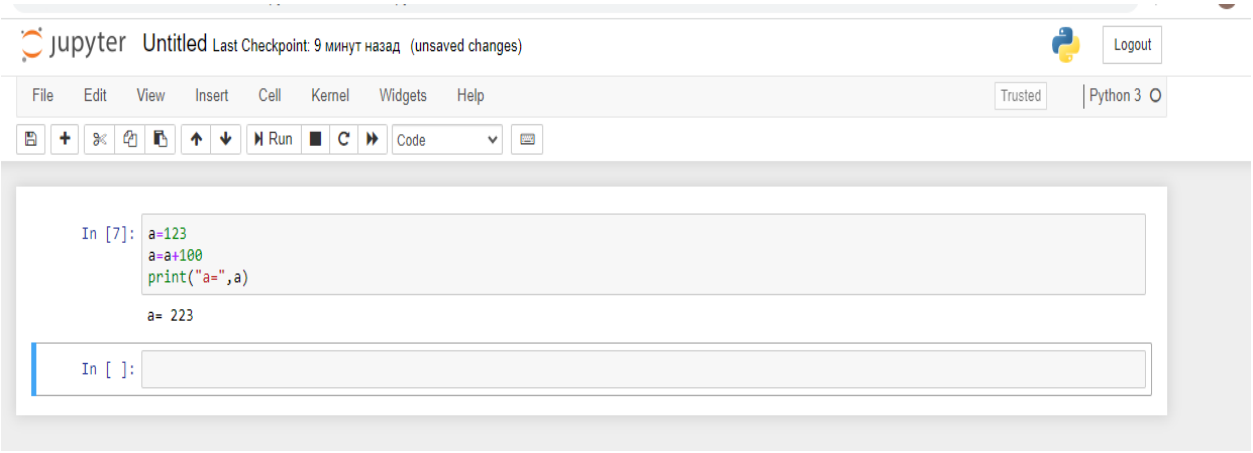
Quyidagi ko‘rinishda oyna hosil bo‘ladi.



Daslabki kodimizni kiritishimiz mumkin. “Salom AI” so‘zini chop qilamiz.

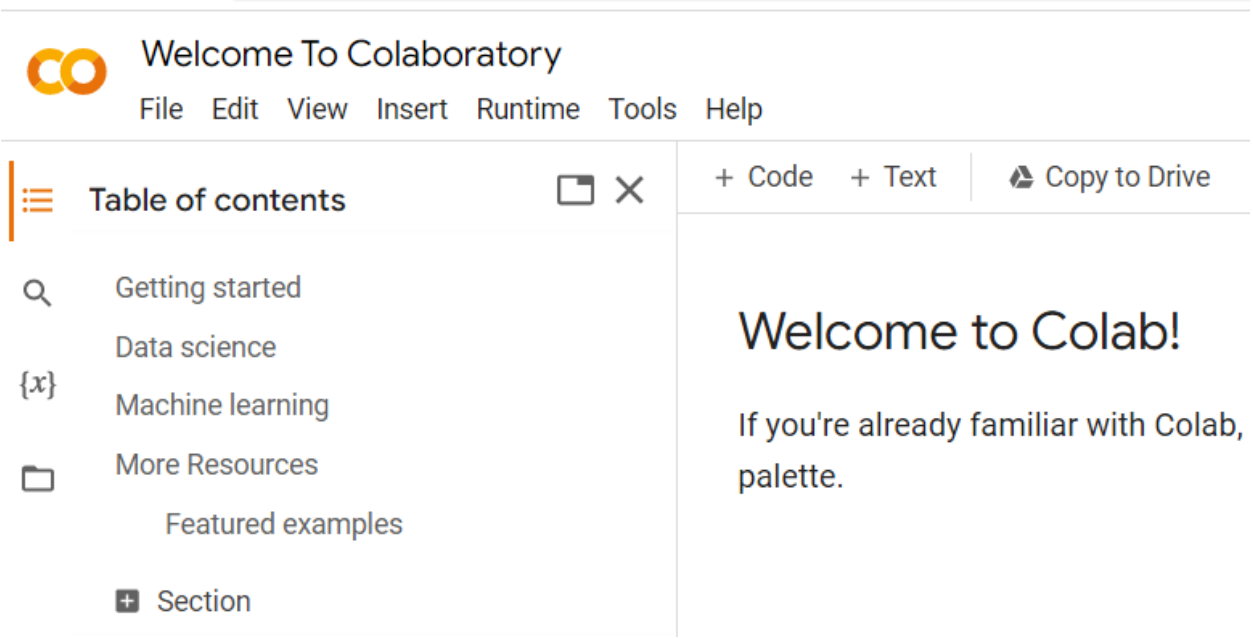


Bu ilovada boshqa amallar va jarayonlarni ham dasturlashni amalga oshirish mumkin. Quyida a o‘zgaruvchiga qiymat berish va bu qiymatni o‘zgartirib ekranga chiqarish jarayoni keltirilgan.

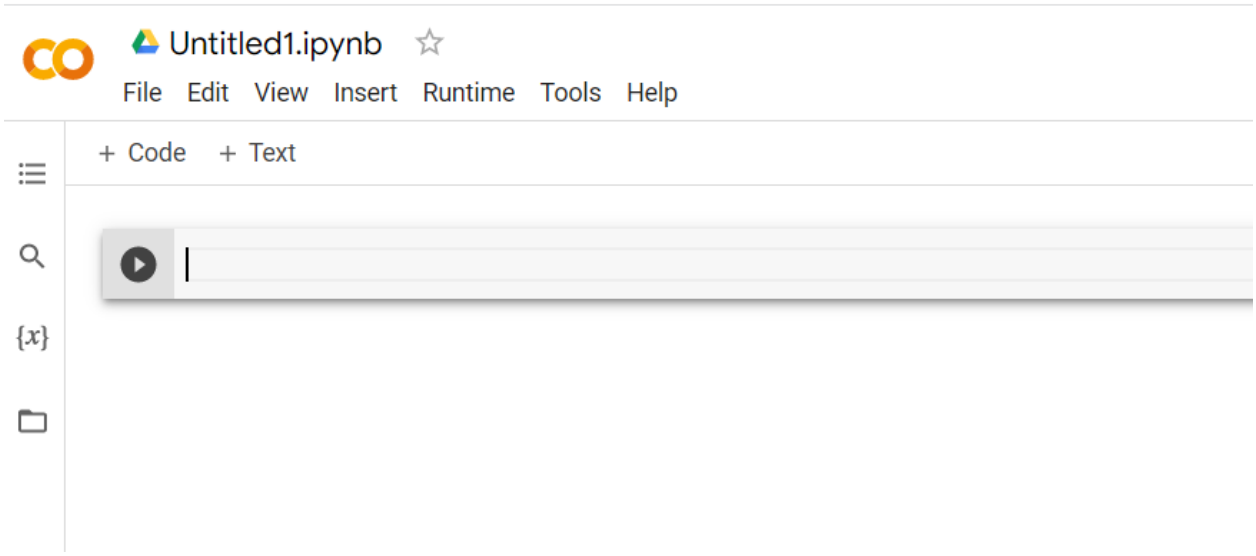


Google Colab

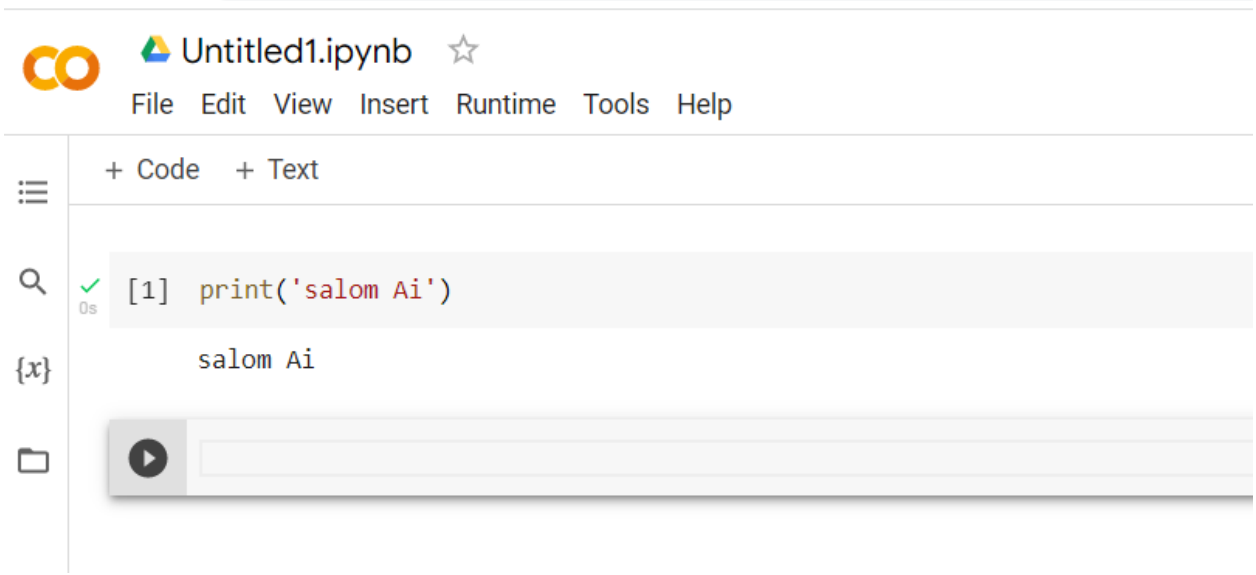
Colab ilovasi Python kodlarini brauzerda yozish va bajarish imkonini beradi. Colabning qulayligi kutubxonalarni alohida o‘rnatishni ta’lab qilmaydi, bu ilova onlayn ishlashni amalga oshiradi. Ma’lumotlar tahlili yoki sun’iy intellekt bo‘yicha mutaxassis bo‘ladiganlar uchun Colab qulay dasturiy vosita hisoblanadi. Colabdan foydalanish uchun www.colab.research.google.com sayti orqali murojaat qilinadi.



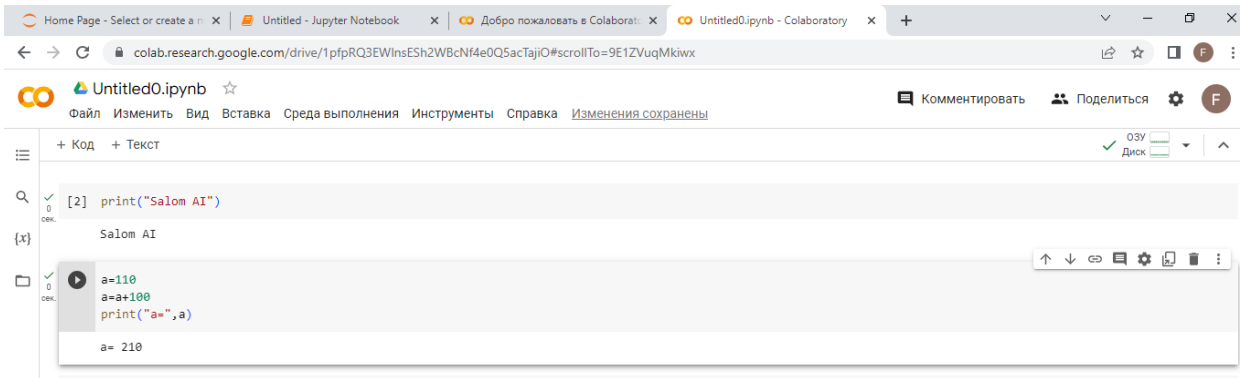
Colabdan foydalanish uchun uni googledagi foydalanuvchining accountiga bog‘lash kerak bo‘ladi. Colabda yangi fayl hosil qilish uchun **file** bo‘limidan **new notebook** bo‘limi tanlanadi va quyidagi ko‘rinishda oyna ochiladi.



Bu ilovada daslabki kodni ishga tushirish uchun, “Salom AI” so‘zini chop qilamiz.



Bu ilovada boshqa amallar va jarayonlarni ham dasturlashni amalga oshirish mumkin. Quyida a o‘zgaruvchiga qiymat berish va bu qiymatni o‘zgartirib ekranga chiqarish jarayoni keltirilgan.



Sun’iy intellekt uchun dasturlar yaratishda yuqorida keltirib o‘tilgan dasturiy ilovalardan foydalanish tavsiya etiladi. Bu dasturlar yordamida kerakli kutubxonalarni oson o‘rnatish va undan foydalanish mumkin.

Nazorat uchun savollar

1. Anaconda dasturi, dasturning imkoniyatlari va o‘rnatish ketma-ketligini tushuntiring?

2. Jupyter Notebook dasturi, dasturning imkoniyatlari va boshqa dasturlardan ustunliklarini ayting?

3. Jupyter Notebook dasturida yangi fayl yaratish jarayonini tushuntiring?

4. Google Colab dasturi, dasturning imkoniyatlari va undan foydalanishni tushuntiring?

1.4. Python dasturlash tili kutubxonalari asosida sun’iy intellekt tizimlarini ishlab chiqish

Reja:

NumPy kutubxonasi;

N-o‘lchamli massiv (array) yaratish va ularga ishlov berish;

N-o‘lchamli massiv (array)larga ishlov berish;

Universal(Unary va Binary) funksiyalar;

Matematik va statistik usullarni qo‘llash;

Tartiblash (Sorting);

Takrorlanmas (unique) va boshqa amallar;

Fayllar bilan ishlash.

NumPy kutubxonasi

NumPy kutubxonasi python dasturlash tili uchun *kutubxona bo‘lib,* katta, ko‘p o‘lchovli massivlar va matritsalar ustida bajariladigan amallarni qo‘llab-quvvatlaydi. Bunda massivlarda ishlash uchun yuqori darajadagi matematik funksiyalarning katta to‘plamini birlashtirish imkonini beradi. NumPy **2005 yilda Travis Oliphant** tomonidan yaratilgan. Bu ochiq kodli loyiha bo‘lib, undan erkin foydalanishingiz mumkin. NumPy raqamli Python degan ma’noni anglatadi, NumPy python kutubxonasi bo‘lib, qisman Python tilida yozilgan.

Pythonda massivlar maqsadiga xizmat qiluvchi ro‘yxatlar bor, lekin ularni qayta ishlash sekin bajariladi. NumPy an’anaviy Python ro‘yxatlariga qaraganda 50 karra tezroq massiv ob’ektini taqdim etishga qaratilgan. NumPydagi massiv ob’ekti **ndarray** deb ataladi , u massiv bilan ishlashni osonlashtiradigan ko‘plab yordamchi funksiyalarni ta'minlaydi. Massivlar tezlik va resurslar juda muhim bo‘lgan ma'lumotlar tahlili fanida tez-tez ishlatiladi.

Numpy ishlatishga oson, ya’ni tajribasidan qat’iy nazar o‘quvchi osongina qo‘llay olishi mumkin. Numpy ochiq manba bo‘lib, ya’ni BSD litsenziyasi asosida yaratilgan bo‘lib, doimo bepul bo‘lishi ta’minlanadi.

Numpy kutubxonasi asosida quyidagi ko‘nikmalarga ega bo‘lish mumkin:

1. Massivlarni yaratish va boshqarish;
2. Indekslash va kesish (Indexing and Slicing);
3. Matematik operatsiyalar;

4. Broadcasting;
5. Ma'lumot turlarini boshqarish;
6. Agregatsiya va statistik funksiyalar;
7. Tasodifiy sonlar generatsiyasi;
8. Matritsalar va chiziqli algebra;
9. Samarali algoritmlar yozish.

NumPyni o‘rnatish: **pip install numpy** ushbu buyruq yordamida u o‘rnatiladi. NumPy o‘rnatilgandan so‘ng, kalit so‘zni qo‘shish orqali u ilovalarga import qilinadi. NumPy odatda **np taxallus ostida** import qilinadi.

```
import numpy as np
```

Pythonda list bilan va NumPy kutubxonasidagi massivlar (arraylar) hisoblashlari orasidagi farqni quyidagicha ko‘rish mumkin.

```
In [ ]: List = list(range(1000))
        Massiv = np.array(range(1000))

In [ ]: %time for _ in range(10):[x*2 for x in List] # Normal
CPU times: user 656 µs, sys: 0 ns, total: 656 µs
Wall time: 660 µs

In [ ]: %time for _ in range(10): Massiv*2 # Vektorlashgan
CPU times: user 571 µs, sys: 0 ns, total: 571 µs
Wall time: 459 µs

In [ ]: 745/98.7

Out[ ]: 7.548125633232016
```

Natija deyarli 8 martani tashkil qilganini ko‘rishimiz mumkin.

N-o‘lchamli massiv (array) yaratish va ularga ishlov berish

Numpyda massiv yaratishning eng oson yo‘llaridan biri bu array funksiyasi yordamida amalga oshiriladi. Ushbu funksiya har qanday turdagi ketma-ket ma'lumotlarni qabul qilib, uni yangi Numpy arrayiga o‘girib beradi.

“List” nomli lug‘at yaratib olib so‘ng uni Numpyga o‘girish quyidagicha bo‘ladi:

```
In [ ]: List = [2,3,5,1.7,9,6.6,41]
        arr = np.array(List)

In [ ]: arr

Out[ ]: array([ 2. ,  3. ,  5. ,  1.7,  9. ,  6.6, 41. ])
```

1) Massiv o‘lchamini **.ndim** metodi yordamida aniqlaymiz.

```
In [ ]: arr.ndim

Out[ ]: 1
```

Ikki va undan katta o‘lchamli massivlar yaratish uchun ham **list**lardan foydalanamiz.

```
In [ ]: List = [[1,3,5,7,9], [2,4,6,8,10]]
        arr = np.array(List)
        arr

Out[ ]: array([[ 1,  3,  5,  7,  9],
               [ 2,  4,  6,  8, 10]])

In [ ]: arr.ndim

Out[ ]: 2
```

2) Shape va size metodlari, shape metodi massivning qator va ustunlar sonini ko‘rsatsa, size esa massivlardagi elementlar sonini nomoyon etadi.

```
In [ ]: arr.shape

Out[ ]: (2, 5)

In [ ]: arr.size

Out[ ]: 10
```

1) N-o‘lchamli massiv yaratish arrange funksiyasi yordamida

```
In [ ]: arr = np.arange(2,29,4)
        arr
```

```
Out[ ]: array([ 2,  6, 10, 14, 18, 22, 26])
```

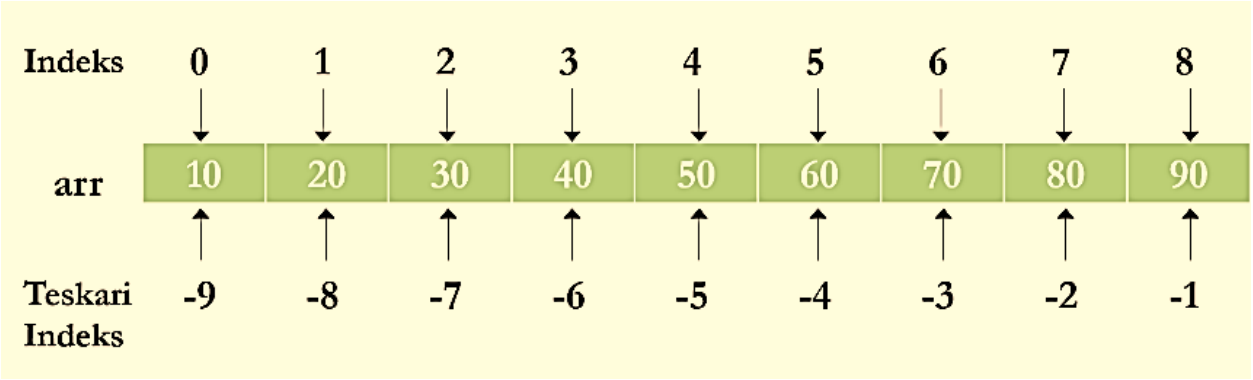
2) Random funksiyasi yordamida yaratish

```
In [ ]: arr = np.random.randn(2,5)
        arr
```

```
Out[ ]: array([[ 0.82163211, -1.10991228,  0.46544394, -0.40587405, -1.05649915],
               [-0.48576041,  1.374917  ,  0.93501954,  1.74232562,  1.31297896]])
```

N-o‘lchamli massiv (array)larga ishlov berish

N-o‘lchamli massivlarda indeks 0 dan boshlanadi yoki teskari tartibda -1 dan boshlanadi.



Bir o‘lchamli massiv yaratamiz va uni indekslariga murojaat qilamiz:

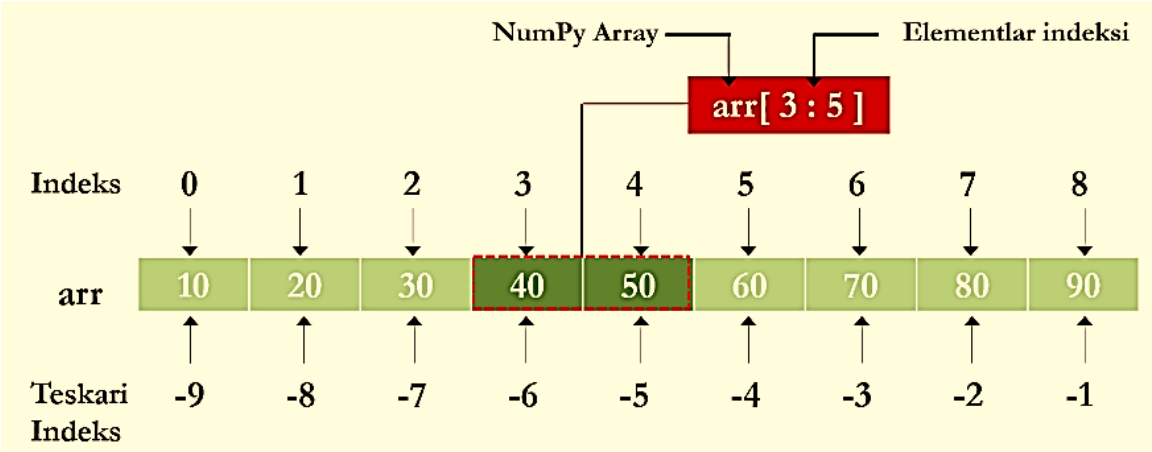
```
import numpy as np
arr=np.arange(10,100,10)
arr
```

```
array([10, 20, 30, 40, 50, 60, 70, 80, 90])
```

```
arr[3:5]
```

```
array([40, 50])
```

“arr” nomli massivning 3:5 indekslaridagi qiymatlariga murojaat qilgandagi natijani ko‘rishingiz mumkin.



Numpy kutubxonasida massivning kesib olingan qismga o‘zgartirish kirtilsa, asosiy manba massivga ta'sir qiladi.

```
arr1[:]=0
arr1
```

```
array([0, 0])
```

```
arr
```

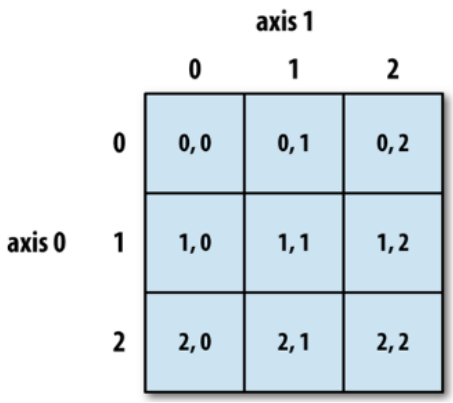
```
array([10, 20, 30,  0,  0, 60, 70, 80, 90])
```

2 o‘lchamli (2d array) massivlarda indeks va kesib olish uchun dastlab 2 o‘lchamli massiv yaratiladi va uni chop qilish quyidagicha bo‘ladi.

```
In [ ]: arr = np.array([[1,2,3], [4,5,6], [7,8,9]])
        arr
```

```
Out[ ]: array([[1, 2, 3],
               [4, 5, 6],
               [7, 8, 9]])
```

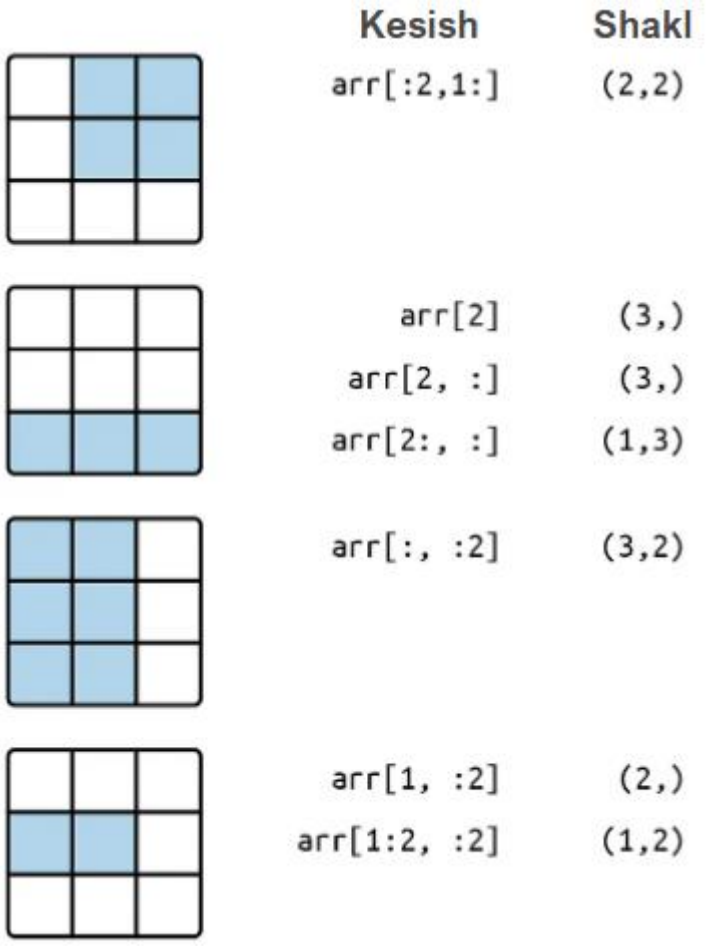
2 o'lchamli massiv



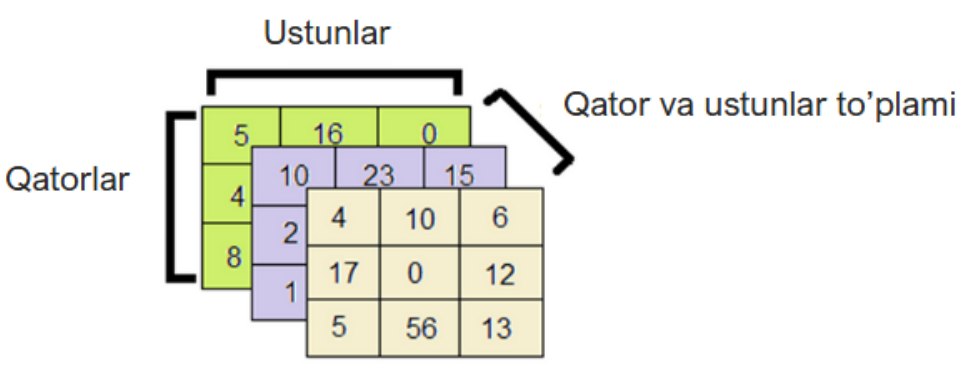
“arr” nomli massivning 3-ustun 2 va 3-elelmentlarini kesib olish tartibi quyidagicha

```
In [ ]: arr_n = arr[2,1:3]
        arr_n

Out[ ]: array([8, 9])
```

2 o‘lchamli massiv elementlarini kesib olishning bir-nechta usullarini ko‘rib chiqamiz.



3-o‘lchamli massivlarda indeks va kesib olish asoslari



3 o‘lchamli (**3d array**) massivlarda indeks va kesib olish uchun dastlab 3 o‘lchamli massiv yaratiladi va uni chop qilish quyidagicha bo‘ladi.

```
In [3]: import numpy as np

        arr = np.array([[[1, 2, 3],[4, 5, 6],[7, 8, 9]],[[10, 11, 12],[13, 14, 15],[16, 17, 18]],
                        [[19, 20, 21],[22, 23, 24],[25, 26, 27]]])
        arr
```

```
Out[3]: array([[[ 1,  2,  3],
                [ 4,  5,  6],
                [ 7,  8,  9]],

               [[10, 11, 12],
                [13, 14, 15],
                [16, 17, 18]],

               [[19, 20, 21],
                [22, 23, 24],
                [25, 26, 27]]])
```

3-o‘lchamli massivning har bir elementiga murojaat qilish tartibi quyidagicha

```
In [5]: arr[0][1,2]
```

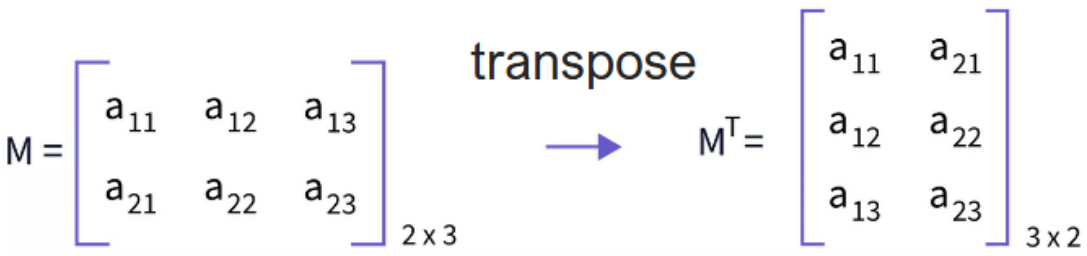
```
Out[5]: 6
```

3-o‘lchamli massiv elementlarini kesib olish tartibi ham 1 va 2 o‘lchovli massivlarnikiga o‘xshash bo‘ladi.

```
In [7]: arr[0][2,0:]
```

```
Out[7]: array([7, 8, 9])
```

Massivlar pozitsiyasini ko‘chirish Transpose va o‘qlar o‘rnini almashtirish Swapping axe orqali bajariladi.



Bir o‘lchamli massiv hosil qilish va **.reshape** metodi yordamida shaklini o‘zgartirish quyidagicha bo‘ladi.

```
In [13]: import numpy as np

         A=np.arange(6)
         A
```

```
Out[13]: array([0, 1, 2, 3, 4, 5])
```

```
In [16]: A=A.reshape(3,2)
         A
```

```
Out[16]: array([[0, 1],
                [2, 3],
                [4, 5]])
```

Trasnpose metodi va uning qo‘llanilishi quyidagicha bo‘ladi.

```
In [17]: A.T
```

```
Out[17]: array([[0, 2, 4],
                [1, 3, 5]])
```

Swapaxes metodi va uning qo‘llanilishi quyidagicha bo‘ladi.

```
In [22]: B = A.swapaxes(1,1)
         B
```

```
Out[22]: array([[0, 1],
                [2, 3],
                [4, 5]])
```

Universal (Unary va Binary) funksiyalar

Numpy kutubxonasiga tegishli (Unary va Binary) turdagi funksiyalar bilan tanishamiz. Unary funksiyalar (bitta argument qabul qiluvchi funksiya) sqrt, square, exp, log, modf, sign, isnan funksiyalarini ko‘rib chiqamiz.

Dastlab “arr” nomli bir o‘chovli massiv yaratib funksiyalarni bajarish quyidagicha bajariladi.

```
In [24]: arr = np.arange(10)
         arr
```

```
Out[24]: array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
```

sqrt massivning har bir elementining kvadrat ildizini qaytaradi

```
In [25]: np.sqrt(arr)

Out[25]: array([0.        , 1.        , 1.41421356, 1.73205081, 2.        ,
                2.23606798, 2.44948974, 2.64575131, 2.82842712, 3.        ])
```

square massivning har bir elementini kvadratga oshiradi.

```
In [26]: np.square(arr)

Out[26]: array([ 0,  1,  4,  9, 16, 25, 36, 49, 64, 81])
```

Qolgan bir nechta funksiyalarni qiymatlarini quyidagi rasmda ko‘rishimiz mumkin.

Har bir element (element-wise) uchun amallar bajaruvchi unary funksiyalar

FUNKSIYA	BAJARILISH VAZIFASI
abs, fabs	Absolyut qiymatni qaytaradi
Sqrt	Kvadrat ildizni qaytaradi
Square	Kvadratga oshirishni bajaradi
Exp	Eksponensial funksiyalashni bajaradi (e^x)
log, log10, log2, log1p	(e) asosga ko‘ra natural logarifm, (10) asosga ko‘ra logarifm, (2) asosga ko‘ra logarifm, log(1+x)
Sign	Ishoralarni hisoblashni bajaradi: **1 (musbat), 0 (nol), -1 (manfiy)**
Ceil	O‘ziga teng yoki o‘zidan katta bo‘lgan eng yaqin butun (int) sonni qaytaradi
Floor	O‘ziga teng yoki o‘zidan kichik bo‘lgan eng yaqin butun (int) sonni qaytaradi
Rint	Butun son (int) ko‘rinishida yaxlitlashni bajaradi
Modf	Sonning qoldiq va butun qismlarini ikkita alohida arraylarda qaytaradi
Isnan	Arrayda, **NaN** (Not a Number) turdagi ma’lumotning mavjud yoki mavjud emasligiga qarab **Boolean** ma’lumotlardan tashkil topgan array qaytaradi

cos, cosh, sin, sinh, tan, tanh	Trigonometrik funksiyalar
arccos, arccosh, arcsin, arcsinh, arctan, arctanh	Teskari trigonometrik funksiyalar
Eslatma: *unary* – faqat bitta argument qabul qiluvchi funksiya.	

Binary funksiyalar (ikkita argument qabul qiluvchi funksiya)

Binary **add, multiply, maximum** funksiyalarini quyidagi vazifalarni bajaradi. **add** ikkita massivning mos elementlarini qo‘shadi. “arr1” va “arr2” nomli massiv yaratib ularni chop qilamiz.

```
In [28]: array1 = np.random.randn(7)
         array2 = np.random.randn(7)

In [29]: array1

Out[29]: array([ 1.18850805, -0.37085848,  0.67874465, -1.46876819,  1.25021799,
                -0.17012449, -2.08463909])

In [30]: array2

Out[30]: array([ 0.15250516, -1.91833638,  0.81990065, -0.03829575,  0.06773307,
                -1.34759282, -2.40953773])
```

Add funksiyasini “arr1” va “arr2” uchun qo‘llash quyidagicha bo‘ladi.

```
In [31]: np.add(array1,array2)

Out[31]: array([ 1.3410132 , -2.28919486,  1.49864531, -1.50706395,  1.31795106,
                -1.5177173 , -4.49417682])
```

multiply ikkita massivning mos elementlarini ko‘paytiradi.

```
In [32]: np.multiply(array1,array2)

Out[32]: array([0.18125361, 0.71143131, 0.55650318, 0.05624758, 0.0846811 ,
                0.22925854, 5.02301655])
```

maximum ikkita massivning elementlarin taqqoslab ulardan kattasining qiymatini qaytaradi.

```
In [33]: np.maximum(array1,array2)

Out[33]: array([ 1.18850805, -0.37085848,  0.81990065, -0.03829575,  1.25021799,
                -0.17012449, -2.08463909])
```

Har bir element (element-wise) uchun amallar bajaruvchi binary funksiyalar

FUNKSIYA	BAJARILISH VAZIFASI
Add	Arraylardagi o‘zaro mos elementlarni qo‘shadi
Subtract	Birinchi arrayning elementlarini ikkinchi arraydagi mos elementlardan ayiradi
Multiply	Array elementlarini ko‘paytiradi
Power	Birinchi arraydagi elementlarni ikkinchi elementlarda ko‘rsatilgan darajalarga oshiradi
max, fmax	Ikkala arraydagi mos qiymatlarni solishtirib maksimumini qaytaradi; fmax esa NaN larni hisobga olmaydi
min, fmin	Ikkala arraydagi mos qiymatlarni solishtirib minimumini qaytaradi; fmin esa NaN larni hisobga olmaydi
Mod	Array elementlarini bo‘linmadan qoldig‘ini (faqat butun qismini) qaytaradi
Copysign	Ikkinchi argument ishorasini birinchi argumentga nusxalash vazifasini bajaradi
greater, greater_equal, less, less_equal, equal, not_equal	Mos elementlar aro taqqoslash amalini bajarib Boolean arrayini qaytaradi (ekvivalent qo‘llanilishi: >, >=, <, <=, ==, !=)
logical_and, logical_or, logical_xor	Mos elementlar aro mantiqiy amallarni bajaradi (ekvivalent qo‘llanilishi: &, **
Eslatma: *binary – ikkita argumentni qabul qiluvchi funksiya.*	

Matematik va statistik usullarni qo‘llash

Numpy kutubxonasida massivlar ustida oddiy arifmetik amallar va maxsus matematik usullarni qo‘llash mumkin.

Asosiy statistik metodlar

Metod	Tavsifi
Sum	Arraydagi barcha elementlarin yig‘indisi

Mean	O‘rtacha arifmetik qiymat
std, var	Standart og‘ish va variatsiya
min, max	Minimum va maksimum
Cumsum	Yig‘ilib boradigan (cumulative) yig‘indi
Cumprod	Yig‘ilib boradigan ko‘paytma

“arr” nomli massiv yaratib olamiz va uni chop qilamiz

```
In [45]: arr = np.random.randn(5,4)
         arr
```

```
Out[45]: array([[-0.61362632, -0.43954461,  1.72973746, -0.18117842],
                [-1.26924814, -1.98457308,  0.28566728, -1.34244383],
                [-0.31378329,  0.67082612, -0.639406  , -0.63029541],
                [-1.29575924,  0.97801542,  0.23943462,  0.62522472],
                [-0.64108548, -1.46738515,  2.65173595,  1.90959678]])
```

1. **Sum metodi.** Massivning barcha elementlari yig‘indisini hisoblab beradi.

```
In [46]: arr.sum()
```

```
Out[46]: -1.7280906336829165
```

Summ(axis=1) qator uchun, summ(axis=0) ustun uchun yig‘indini hisoblaydi.

2.**Mean metodi.** O‘rtacha arifmetik qiymat, “data” nomli massiv yaratib olamiz va uni chop qilamiz.

```
In [49]: data = np.arange(10)
         data
```

```
Out[49]: array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
```

“data” nomli massiv uchun mean metodini qo‘llash.

```
In [50]: mean = np.sum(data)/len(data)
         mean
```

```
Out[50]: 4.5
```

```
In [51]: np.mean(data)
```

```
Out[51]: 4.5
```

np.mean(arr,axis=1) qator bo‘yicha np.mean(arr,axis=0) ustun bo‘yicha o‘rtacha qiymat qaytaradi.

3. **Cumsum** metodi yig‘lib boradigan yig‘indi.

```
In [52]: data.cumsum()

Out[52]: array([ 0,  1,  3,  6, 10, 15, 21, 28, 36, 45])
```

Tartiblash (Sorting)

To‘rtta ta taxminiy elementlardan tashkil topgan massiv yaratib olib uni chop qilish quyidagicha amalga oshiriladi.

```
In [54]: arr = np.random.randn(9)
         arr

Out[54]: array([ 1.2260589 , -0.30303175,  0.77770969,  0.78501687, -1.20870728,
                -0.68145395,  0.93387794,  0.95149068,  0.38839838])
```

Sort funksiyasi yordamida “arr” massivning elementlarini tartiblash quyidagicha bo‘ladi.

1-usul:

```
In [56]: Sarr = np.sort(arr)
         Sarr

Out[56]: array([-1.20870728, -0.68145395, -0.30303175,  0.38839838,  0.77770969,
                0.78501687,  0.93387794,  0.95149068,  1.2260589 ])
```

2-usul:

```
In [59]: arr.sort()
         arr

Out[59]: array([-1.20870728, -0.68145395, -0.30303175,  0.38839838,  0.77770969,
                0.78501687,  0.93387794,  0.95149068,  1.2260589 ])
```

3-usul: Teskari tartiblash.

```
In [60]: Tarr = -np.sort(-arr)
         Tarr

Out[60]: array([ 1.2260589 ,  0.95149068,  0.93387794,  0.78501687,  0.77770969,
                0.38839838, -0.30303175, -0.68145395, -1.20870728])
```

2 o‘lchovli massiv uchun tartiblashni ko‘ramiz, axis=0 – ustun uchun, axis=1 – qator uchun.

```
In [63]: Sarr1 = np.sort(arr, axis=0)
         Sarr1

Out[63]: array([-1.20870728, -0.68145395, -0.30303175,  0.38839838,  0.77770969,
                 0.78501687,  0.93387794,  0.95149068,  1.2260589 ])
```

Takrorlanmas (unique) va boshqa amallar

Faqat so‘zlardan va faqat sonlardan iborat 2 ta massiv yaratiladi.

```
In [67]: word = np.array(['Python', 'PHP', 'Java', 'C++', 'Ruby', 'PHP', 'Python'])
         num = np.array([3,5,11,42,21,11,34,5,4,3])
```

1.Unique “word” dagi takrorlanmas elementlarni qaytaradi (takrorlansa bittasini qabul qiladi)

```
In [68]: np.unique(word)

Out[68]: array(['C++', 'Java', 'PHP', 'Python', 'Ruby'], dtype='<U6')

In [69]: np.unique(num)

Out[69]: array([ 3,  4,  5, 11, 21, 34, 42])
```

2.**In1d** usuli “arr2” dagi elementlarni “arr1” mavjudligini tekshiradi.

```
In [71]: arr1 = np.array([1,2,3,4,5])
         arr2 = np.array([3,4,5,6,7])

In [72]: np.in1d(arr1,arr2)

Out[72]: array([False, False,  True,  True,  True])
```

3.**Setdiff1d** usuli “arr1” dagi elementlarni “arr2” massivdagi takorlanmas qismini qaytaradi.

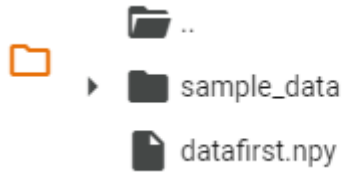
```
In [73]: np.setdiff1d(arr1,arr2)

Out[73]: array([1, 2])
```

Fayllar bilan ishlash

Yaratgan massivlarni fayllarga saqlash va ularga ishlov berish quyidagicha amalga oshiriladi:

1.Dastlab, “arr” nomli massiv yaratiladi va uni datafirst nomli faylga saqlanadi.

```
In [74]: arr = np.arange(12)
         np.save('datafirst', arr)
```



Ushbu faylni qayta yuklash uchun .load metodidan foydalaniladi.

```
In [75]: arr1 = np.load('datafirst.npy')
         arr1

Out[75]: array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11])
```

2. Bir nechta massivlarni bitta faylga saqlash uchun .savez metodidan foydalaniladi.

```
In [76]: np.savez('array', a=arr1, b=arr2)

In [77]: arr3 = np.load('array.npz')

In [78]: arr3['a']

Out[78]: array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11])

In [79]: arr3['b']

Out[79]: array([3, 4, 5, 6, 7])
```

3 .savez_compressed metodi - .saves metodiga o‘xshash lekin hajmida o‘zgarish bo‘ladi.

```
In [80]: np.savez_compressed('array2', a=arr1, b=arr2)

In [82]: arr4 = np.load('array2.npz')
         arr4['a']

Out[82]: array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11])
```

Nazorat uchun savollar

1. NumPy kutubxonasi, uni imoprt qilish va vazifalari?
2. Massivlar ustida normal va vektorlashgan hisoblashlar?
3. NumPy kutubxonasini import qilish?
4. N-o‘lchamli massiv (array) yaratish va ularga ishlov berish?
5. N-o‘lchamli massiv (array)larga ishlov berish?

6. Universal Unary va Binary funksiyalar va ularni vazifalari?
7. Mantiqiy shart operatori va uning vazifasi?
8. Matematik va statistik usullarni qo‘llash?
9. Tartiblash (Sorting) buyruqlari va ularni vazifalari?
10. Takrorlanmas (unique) va boshqa amallar?
11. Fayllar bilan ishlash va ularni vazifalari?

1.5. Sun’iy intellekt tizimlarini ishlab chiqishda Python dasturlash tili kutubxonalaridan foydalanish. Pandas kutubxonasi

Reja:

Pandas kutubxonasi;
DataFrame ma’lumotlar tuzilmasi;
DataFrame: Qayta indekslash;
DataFrame va Seriesda elementlarni tanlash;
Pandas kutubxonasida arifmetik amallar;
Pandas obyekti elementlariga funksiyalarni qo‘llash;
Tartiblash va Reytinglash;
Dataset statistikasi;
Korrelyatsiya;
Ma’lumotlarni filterlash.

Pandas kutubxonasi

Pandas - bu ochiq manbali kutubxona bo‘lib, u asosan bir-biri bilan bog‘langan yoki etiketli ma'lumotlar bilan oson va intuitiv tarzda ishlash uchun yaratilgan. U turli xil ma’lumotlar tuzilmalari va raqamli ma’lumotlar hamda vaqt seriyalarini manipulyatsiya qilish uchun operatsiyalarni taqdim etadi. Bu kutubxona **NumPy** kutubxonasi ustiga qurilgan. Pandas tez va u foydalanuvchilar uchun yuqori unumdorlikka ega hisoblanadi.

Pandas kutubxonasi afzalliklari:

- Ma’lumotlarni manipulyatsiya qilish va tahlil qilish uchun tez va samarali;
- Turli fayl ob’ektlaridan ma’lumotlarni yuklash mumkin;
- Suzuvchan nuqtada yetishmayotgan ma'lumotlarni (NaN sifatida ko‘rsatilgan) va suzuvchan nuqta ma’lumotlarini oson boshqarish;
- Hajmi o‘zgaruvchanligi: ustunlarni DataFrame va undan yuqori o‘lchamli ob'ektlardan kiritish va o‘chirish mumkin;
- Ma'lumotlar to‘plamini birlashtirish va qo‘shish;
- Ma'lumotlar to‘plamlarini moslashuvchan qayta shakllantirish va aylantirish;
- Vaqt seriyali funksiyalarini ta'minlaydi;
- Ma'lumotlar to‘plamlarida bo‘lish va birlashtirish operatsiyalarini bajarish uchun funksional imkoniyatlar mavjudligi.

Pandasda ishlashning birinchi qadami uning Python papkasida o‘rnatilgan yoki o‘rnatilmaganligiga ishonch hosil qilishdir. Agar yo‘q bo‘lsa, biz uni tizimga **pip** buyrug‘i yordamida o‘rnatishimiz kerak: **pip install pandas**.

Pandas tizimga o‘rnatilgandan so‘ng, kutubxonani import qilishingiz kerak:

Import pandas as pd

Pandas odatda ma'lumotlarni manipulyatsiya qilish uchun ikkita ma'lumotlar tuzilmasini taqdim etadilar, ular:

- Seriya;
- DataFrame.

Series ma’lumotlar tuzilmasi

Series: Pandas Series - bu har qanday turdagi ma'lumotlarni (int, string, float, python ob'ektlari va boshqalar) saqlashga qodir bo‘lgan bir o‘lchovli etiketli massiv hisoblanadi.

Pandas kutubxonasidagi Series ma’lumotlar tuzilmasini Pythondagi oddiy kutubxonalardan farqlaridan biri ro‘yhatdagi obyektlarga o‘zimiz index berishimiz mumkin.

Dastlab “object” nomli Series turidagi massiv yaratib, indexlarini o‘zimiz beramiz. So‘ng pandas kutubxonasi bilan pandasdagi Series funksiyalarini imort qilib olish kerak bo‘ladi.

```
import pandas as pd
from pandas import Series
```

```
In [ ]: object = Series([3, 7, 12, 15], index = ['a', 'b', 'c', 'd'])
        object.index

Out[ ]: Index(['a', 'b', 'c', 'd'], dtype='object')
```

Obyekt ichidagi qiymatlarga indeks orqali murojaat qilamiz:

```
In [ ]: object['c']

Out[ ]: 12
```

Series obyektiga oid ba'zi parametrlar va metodlar quyidagicha bo‘ladi:

1) **values** – obyekt qiymatlarini ekranga chiqaradi

```
In [ ]: object.values

Out[ ]: array([ 3,  7, 12, 15])
```

2) **hasnans** – NaN qiymatlar bor yoki yo‘qligini tekshiradi.

```
In [ ]: object.hasnans

Out[ ]: False
```

3) **dtype** – qiymatlarning ma’lumot turini tekshiradi.

```
In [ ]: object.dtype

Out[ ]: dtype('int64')
```

4) **is_unique** – qiymatlar takrorlanmas ekanligini tekshiradi.

```
In [ ]: object.is_unique

Out[ ]: True
```

5) **shape** – Series hajmini aniqlaydi

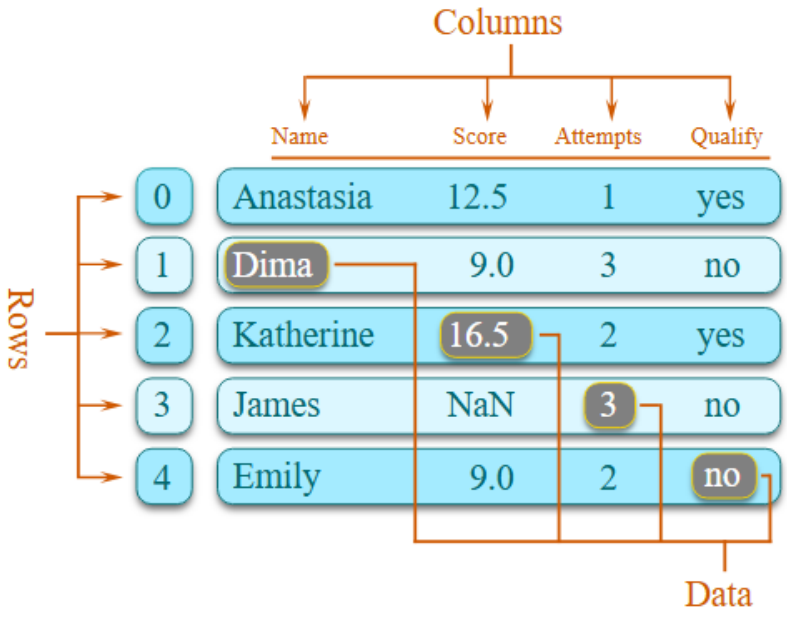
```
In [ ]: object.shape

Out[ ]: (4,)
```

6) **max va min** – qiymatlarning maksimum va minimumini aniqlaydi.

```
In [ ]: object.min()

Out[ ]: 3
```

DataFrame ma’lumotlar tuzilmasi

Pandas DataFrame 2 o‘lchovli ma’lumotlar strukturasi bo‘lib, 2 o‘lchovli massiv yoki satr va ustunli jadval kabi ko‘rinishda bo‘ladi. Pandas DataFrame uchta asosiy **ma’lumot** , **qator** va **ustun** komponentlaridan iborat.



Pandas DataFrame

Pandas DataFrame mavjud xotiradan ma'lumotlar to‘plamini yuklash orqali yaratiladi, fayllar SQL ma'lumotlar bazasi, CSV fayli va Excel fayli bo‘lishi mumkin. Pandas DataFrame ro‘yxatlar, lug‘atlar va hokazolardan yaratilishi mumkin.

DataFrame yaratishning bir nechta usuli bor va bulardan eng osoni qiymatlari bir xil uzunlikdagi ro‘yxatdan iborat lug‘at orqali yaratish mumkin.

Dastlab “**data**” nomli lug‘at yaratib olinadi.

```
data = {
    'model': ['Spark', 'Nexia', 'Cobalt', 'Gentra', 'Lacetti', 'Malibu',
              'Tracker', 'Equinox', 'Tahoe', 'Captiva'],
    'yili': [2023, 2022, 2021, 2020, 2019, 2023, 2022, 2021, 2020, 2019],
    'motor': [1.5, 1.6, 1.5, 1.5, 1.6, 2.0, 1.4, 1.5, 5.3, 2.4],
    'narx': [15000, 10000, 12000, 11000, 9000, 25000, 2000, 2200, 4000, 18000],

}
```

pd.DataFrame() buyrug‘i orqali lug‘atni DataFramega o‘tkazamiz.

```
df = pd.DataFrame(data)
df
```

Out[ ]:

	model	yili	motor	narx
0	Spark	2023	1.5	15000
1	Nexia	2022	1.6	10000
2	Cobalt	2021	1.5	12000
3	Gentra	2020	1.5	11000
4	Lacetti	2019	1.6	9000
5	Malibu	2023	2.0	25000
6	Tracker	2022	1.4	20000
7	Equinox	2021	1.5	22000
8	Tahoe	2020	5.3	40000
9	Captiva	2019	2.4	18000

DataFrame yaratishning yana bir usuli lug‘atlar lug‘atidan foydalanish ham mumkin. Bunda tashqi lug‘at kalitlari ustun, ichki lug‘at kalitlari esa indeks bo‘ladi.

In [ ]:
```
data = {
    'Xiaomi': {2020: 250, 2021: 270, 2022: 260},
    'Samsung': {2020: 200, 2021: 210, 2022: 190},
    'Apple': {2020: 180, 2021: 170, 2022: 160}
}

df = pd.DataFrame(data)
df
```

Out[ ]:

	Xiaomi	Samsung	Apple
2020	250	200	180
2021	270	210	170
2022	260	190	160

Indekslar va ustunlar o‘rnini almashtirish uchun .T (transpose) metodidan foydalanish mumkin.

In [ ]:
```
df.T
```

Out[ ]:

	2020	2021	2022
Xiaomi	250	270	260
Samsung	200	210	190
Apple	180	170	160

.values parametri yordamida DataFrame ichidagi qiymatlarni 2-o‘lchamli massiv ko‘rinishida olishimiz mumkin.

In [ ]:
```
df.values
```

Out[ ]:
```
array([[250, 200, 180],
       [270, 210, 170],
       [260, 190, 160]])
```

DataFrame: Qayta indekslash

DataFrameda qayta indekslash uchun **.index** funksiyasidan foydalaniladi.

1) **.index** funksiyasi

```
data = {
    'model': ['Spark', 'Nexia', 'Cobalt', 'Gentra', 'Lacetti', 'Malibu',
              'Tracker', 'Equinox', 'Tahoe', 'Captiva'],
    'year': [2023, 2022, 2021, 2020, 2019, 2023, 2022, 2021, 2020, 2019],
    'engine_capacity': [1.5, 1.6, 1.5, 1.5, 1.6, 2.0, 1.4, 1.5, 5.3, 2.4],
    'price': [15000, 10000, 12000, 11000, 900, 2500, 2000, 2200, 40000, 18000],
    'color': ['qora', 'kulrang', 'oq', 'sariq', 'qizil', 'kok', 'yashil',
              'pushti', 'qora', 'kulrang']
}

df = pd.DataFrame(data)
df.head()
```

Out[ ]:

	model	year	engine_capacity	price	color
0	Spark	2023	1.5	15000	qora
1	Nexia	2022	1.6	10000	kulrang
2	Cobalt	2021	1.5	12000	oq
3	Gentra	2020	1.5	11000	sariq
4	Lacetti	2019	1.6	9000	qizil

df nomli DataFrameni indekslarini o‘zgartiramiz:

In [ ]:
```
df.index = [13,14,15,16,17,18,19,20,21,22]
df
```

Out[ ]:

	model	year	engine_capacity	price	color
13	Spark	2023	1.5	15000	qora
14	Nexia	2022	1.6	10000	kulrang
15	Cobalt	2021	1.5	12000	oq
16	Gentra	2020	1.5	11000	sariq
17	Lacetti	2019	1.6	9000	qizil

2) **.reindex**

.reindex funksiyasi yordamida indekslar o‘rnini almashtirish mumkin.

In [ ]:
```
df = pd.DataFrame(data)
df.index = [13, 14, 15, 16, 17, 18, 19, 20, 21, 22]

new_index = [17, 18, 19, 20, 21, 22, 23, 24, 25, 26]
df = df.reindex(new_index)
df.head(6)
```

Out[ ]:

	model	year	engine_capacity	price	color
17	Lacetti	2019.0	1.6	9000.0	qizil
18	Malibu	2023.0	2.0	25000.0	kok
19	Tracker	2022.0	1.4	20000.0	yashil
20	Equinox	2021.0	1.5	22000.0	pushti
21	Tahoe	2020.0	5.3	40000.0	qora
22	Captiva	2019.0	2.4	18000.0	kulrang

.reindex yordamida yangi indekslar qo‘shish va ularni turli usullar (method) yordamida to‘ldirish ham mumkin.

In [ ]:
```
df.reindex(range(15,25), method='ffill')
```

Out[ ]:

	model	year	engine_capacity	price	color
15	NaN	NaN	NaN	NaN	NaN
16	NaN	NaN	NaN	NaN	NaN
17	Lacetti	2019.0	1.6	9000.0	qizil
18	Malibu	2023.0	2.0	25000.0	kok
19	Tracker	2022.0	1.4	20000.0	yashil
20	Equinox	2021.0	1.5	22000.0	pushti
21	Tahoe	2020.0	5.3	40000.0	qora
22	Captiva	2019.0	2.4	18000.0	kulrang

3) **.drop()**

DataFramedan elementlarni o‘chirish. .drop()

DataFramedan biror qator yoki ustunni o‘chirish uchun qator (ustun) indeksini bilish kifoya:

```
In [ ]: df1 = df.drop(20)
        df1
```

Out[ ]:

	model	year	engine_capacity	price	color
17	Lacetti	2019.0	1.6	9000.0	qizil
18	Malibu	2023.0	2.0	25000.0	kok
19	Tracker	2022.0	1.4	20000.0	yashil
21	Tahoe	2020.0	5.3	40000.0	qora
22	Captiva	2019.0	2.4	18000.0	kulrang

`.drop()` yordamida nafaqat qatorlarni, balki ustunlarni ham o‘chirish mumkin. Buning `.drop()` metodining `axis` parametri ishlatiladi, axis=1 qator, axis=0 ustun.

```
In [ ]: df1 = df.drop('model', axis = 1)
        df1
```

Out[ ]:

	year	engine_capacity	price	color
17	2019.0	1.6	9000.0	qizil
18	2023.0	2.0	25000.0	kok
19	2022.0	1.4	20000.0	yashil
20	2021.0	1.5	22000.0	pushti
21	2020.0	5.3	40000.0	qora
22	2019.0	2.4	18000.0	kulrang

.drop va .reindex kabi funksiyalarni qo‘llaganimizda asl DataFrame o‘zgarmaydi, lekin buni maxsus “inplace” parametri yordamida

o‘zgarishlarni saqlashimiz mumkin. Bu parametr standart False qiymatga ega bo‘ladi.

```
In [ ]: df.drop('year', axis = 1, inplace=True)
        df
```

Out[ ]:

	model	engine_capacity	price	color
17	Lacetti	1.6	9000.0	qizil
18	Malibu	2.0	25000.0	kok
19	Tracker	1.4	20000.0	yashil
20	Equinox	1.5	22000.0	pushti
21	Tahoe	5.3	40000.0	qora
22	Captiva	2.4	18000.0	kulrang

DataFrame va Seriesda elementlarni tanlash

1) Seriesda ma’lumotlar tuzilmasida elementlarni tanlash.

DataFrame aslida Serieslardan tashkil topadi. Yuqoridagi “df” nomli DataFramedan Series obyektini yaratib olamiz.

```
In [ ]: data = df['price']
        data
```

Out[ ]:

	price
17	9000.0
18	25000.0
19	20000.0
20	22000.0
21	40000.0

Type funksiyasi orqali “data”ning turini tekshirsak Series ma’lumotlar tuzilmasiga oid ekanligini ko‘rishimiz mumkin.

```
In [ ]: ty = data.dtypes
        ty
```

```
Out[ ]: dtype('float64')
```

“data” nomli Series ma’lumotlar tuzilmasining elementlarini indeks orqali tanlashimiz mumkin.

```
In [ ]: data[[18]]
```

Out[ ]:

	price
18	25000.0

dtype: float64

```
In [ ]: data[[17,20,18]]
```

Out[ ]:

	price
17	9000.0
20	22000.0
18	25000.0

“data” nomli Series ma’lumotlar tuzilmasining qandaydir oraliqdagi elementlari kerak bo‘lsa data[a:b] shaklda murojaat qilamiz.

```
In [ ]: data[0:5]
```

Out[ ]:

	price
17	9000.0
18	25000.0
19	20000.0
20	22000.0
21	40000.0

Series ma’lumotlar tuzilmasining elementlarini filtrlashning quyidagi usullari mavjud.

```
In [ ]: data[data < 25000]
```

Out[ ]:

	price
17	9000.0
19	20000.0
20	22000.0

dtype: float64

```
In [ ]: data[data == 25000]
```

Out[ ]:

	price
18	25000.0

DataFrameda ma’lumotlar tuzilmasida elementlarni tanlash. “df” nomli yangi DataFrame ma’lumotlar tuzilmasini yaratib olib bajariladi.

```
data = {
'city_names' : ['Tashkent', 'Samarkand', 'Bukhara', 'Khiva', 'Nukus', 'Andijan', 'Surkhandarya'],
'temperatures' : [25, 28, 30, 32, 35, 32, 37],
'humidity' : [50, 55, 60, 65, 70, 45, 52],
'wind_speed' : [10, 12, 15, 18, 20, 13, 11],
'precipitation' : [0, 0, 1, 2, 3, 4, 1],
'cloud_cover' : [30, 40, 50, 60, 70, 35, 45],
'pressure' : [1010, 1012, 1015, 1018, 1020, 1323, 1434]
}

df = pd.DataFrame(data)
df
```

Out[ ]:

	city_names	temperatures	humidity	wind_speed	precipitation	cloud_cover	pressure
0	Tashkent	25	50	10	0	30	1010
1	Samarkand	28	55	12	0	40	1012
2	Bukhara	30	60	15	1	50	1015
3	Khiva	32	65	18	2	60	1018
4	Nukus	35	70	20	3	70	1020
5	Andijan	32	45	13	4	35	1323
6	Surkhandarya	37	52	11	1	45	1434

“df” nomli DataFrame ma’lumotlar tuzilmasidan istalgan ustun ajratib olish mumkin.

```
In [ ]: df[['wind_speed']]
```

Out[ ]:

	wind_speed
0	10
1	12
2	15
3	18
4	20
5	13
6	11

“df” nomli DataFrame ma’lumotlar tuzilmasidan istalgan oraliqdagi indekslarni ajratib olish mumkin.

```
In [ ]: df[1:4]
```

Out[ ]:

	city_names	temperatures	humidity	wind_speed	precipitation	cloud_cover	pressure
1	Samarkand	28	55	12	0	40	1012
2	Bukhara	30	60	15	1	50	1015
3	Khiva	32	65	18	2	60	1018

“df” nomli yangi DataFrame ma’lumotlar tuzilmasida mantiqiy shartlardan ham foydalanish mumkin.

```
In [ ]: df[['temperatures']][df['temperatures']> 30]
```

Out[ ]:

	temperatures
3	32
4	35
5	32
6	37

Pandas kutubxonasida arifmetik amallar

Ikki `Series` yoki `DataFrame` obyektlari o‘rtasida arifmetik amallar bajarishda indekslar juda muhim rol o‘ynaydi. Amallar mos tushuvchi indekslar o‘rtasida bajariladi. Agar obyektlarning birida mavjud indeks ikkinchisda yo‘q bo‘lsa natija `NaN` bo‘ladi. Arifmetik amallar bajarish uchun 2 ta “s1” va “s2” Series obyektlarini yaratib olamiz.

```
In [ ]: s1 = pd.Series([23, 41, 56, 53, 29], index = ['a', 'b', 'c', 'd', 'e'])
        s2 = pd.Series([12.5, 15.9, 5.2, 13, 87], index = ['a', 'b', 'h', 'e', 's'])
```

s1+s2 amalini natijasini ko‘rishimiz mumkin.

```
In [ ]: s1 + s2
```

Out[ ]:

	0
a	35.5
b	56.9
c	NaN
d	NaN
e	42.0
h	NaN
s	NaN

s1 da mavjud c indeksi s2da mavjud bo‘lmaganligi sabab s1+s2 da c indeksdagi qiymat NaN bo‘ldi. Yuqoridagi holat DataFramelarga ham xos. Serieslardan farqli ravishda DataFrameda qator va ustun indekslari hisobga olinadi.

DataFrameda Arifmetik amallar bajarish uchun 2 ta “df1” va “df2” DataFrame obyektlarini yaratib olamiz,

```
[ ]  df1 = pd.DataFrame(np.arange(9).reshape(3, 3), columns = list('abc'),
                        index = ['sabzi', 'pomidor', 'piyoz'])
     df2 = pd.DataFrame(np.arange(12).reshape(4, 3), columns = list('bcd'),
                        index = ['sabzi', 'pomidor', 'piyoz', 'kartoshka'])
```

“df1” va “df2” obyektarini chop qilamiz.

```
In [ ]: df1
```

Out[ ]:

	a	b	c
sabzi	0	1	2
pomidor	3	4	5
piyoz	6	7	8

```
In [ ]: df2
```

Out[ ]:

	b	c	d
sabzi	0	1	2
pomidor	3	4	5
piyoz	6	7	8
kartoshka	9	10	11

“df1” va “df2” obyektlarini qo‘shamiz.

```
In [ ]: df1 + df2
```

Out[ ]:

	a	b	c	d
kartoshka	NaN	NaN	NaN	NaN
piyoz	NaN	13.0	15.0	NaN
pomidor	NaN	7.0	9.0	NaN
sabzi	NaN	1.0	3.0	NaN

`a` va `d` ustunlar hamda `kartoshka` va `sabzi` qatorlar ikkala `DataFrame`da bo‘lmagani sababli yuqoridagi NAN natija chiqdi.

Yuqoridagi qo‘shish amalini `.add()` metodi yordamida ham bajarish mumkin.

```
In [ ]: df1.add(df2)
```

Out[ ]:

	a	b	c	d
kartoshka	NaN	NaN	NaN	NaN
piyoz	NaN	13.0	15.0	NaN
pomidor	NaN	7.0	9.0	NaN
sabzi	NaN	1.0	3.0	NaN

Bunda maxsus `fill_value` parametri yordamida mos tushuvchi indekslardagi NaN qiymatlarni boshqa qiymat bilan almashtirish mumkin. Bu usulda yakuniy natijada ikkala df da ham mavjud bo‘lmagan indekslarda `NaN` qiymati qoladi.

```
In [ ]: df2.add(df1, fill_value=0)
```

Out[ ]:

	a	b	c	d
kartoshka	NaN	9.0	10.0	11.0
piyoz	6.0	13.0	15.0	8.0
pomidor	3.0	7.0	9.0	5.0
sabzi	0.0	1.0	3.0	2.0

Boshqa arifmetik amallar

- `add`, `radd` - Qo‘shish (+)
- `sub`, `rsub` - Ayirish (-)
- `div`, `rdiv` - Bo‘lish (/)
- `floordiv`, `rfloordiv` - Bo‘lish va butun qismini olis (//)
- `mul`, `rmul` - Ko‘paytirish (*)
- `pow`, `rpow` - Eksponenta (**)

Yuqoridagi amallarning barchasida `fill_value` parametri mavjud bo‘lib, `r` bilan boshlangan metodlarda `DataFrame` ifodaning o‘ng tarafida bo‘ladi.

Pandas obyekti elementlariga funksiyalarni qo‘llash

“df” nomli DataFrame yaratib olib va **.abs** funksiyasi foydalaniladi.

```
[ ]  df = pd.DataFrame(np.random.randn(3,4), index = list('abc'),
                       columns = ['sabzi', 'pomidor', 'piyoz', 'lavlagi'])
     df
```

Out[ ]:

	sabzi	pomidor	piyoz	lavlagi
a	-0.512258	-1.825022	-0.897561	-1.829579
b	0.276274	0.187842	-0.099863	-1.450242
c	1.145831	0.000722	0.584264	0.898253

1) **.abs** – barcha elementlarni absolyut qiymatini qaytaradi

```
In [ ]: np.abs(df)
```

Out[ ]:

	sabzi	pomidor	piyoz	lavlagi
a	0.512258	1.825022	0.897561	1.829579
b	0.276274	0.187842	0.099863	1.450242
c	1.145831	0.000722	0.584264	0.898253

2) **.round** – elementlarni yaxlitlash uchun foydalaniladi.

```
In [ ]: np.round(df)
```

Out[ ]:

	sabzi	pomidor	piyoz	lavlagi
a	-1.0	-2.0	-1.0	-2.0
b	0.0	0.0	-0.0	-1.0
c	1.0	0.0	1.0	1.0

3) **.mean** – har bir ustunning o‘rtacha qiymatini hisoblaydi.

```
In [ ]: df.mean()
```

Out[ ]:

	0
sabzi	0.303282
pomidor	-0.545486
piyoz	-0.137720
lavlagi	-0.793856

Tartiblash va Reytinglash

Series ma’lumotlar tuzilmasi elementlarini tartiblash uchun dastlab **“So”** nomli Series yaratib olinadi va uni ekranga chop qilish quyidagicha bo‘ladi.

```
In [ ]: So = pd.Series(range(5), index = ['a', 'd', 'e', 'b', 'c'])
        So
```

Out[ ]:

	0
a	0
d	1
e	2

1) **Series obyektini Indeks bo‘yicha tartiblash.**

```
In [ ]: So.sort_index()
```

Out[ ]:

	0
a	0
b	3
c	4

2) **Series obyektini qiymatlari bo‘yicha tartiblash.**

```
In [ ]: So.sort_values()
```

Out[ ]:

	0
a	0
d	1
e	2

DataFrame ma’lumotlar tuzilmasi elementlarini tartiblash uchun dastlab “df” nomli DataFrameni Githubdan yuklab olish kerak yoki tayyorini yuklash ta’lab etiladi.

Colabda ishlayotganlar uchun Dataframeni Colabning Fayl qismidan.csv fayli yuklanadi, keyin quyidagi kod bilan “df” Dataframe faollashtiriladi. Jupyter Notebookda ishlayotganlar DataFrameni manzili bilan to‘liq yozish kerak bo‘ladi.

```
df = pd.read_csv("vgsales new.csv")
```

DataFrameni qaysidir qismidagi ma’lumotlarni ko‘rish uchun quyidagi buruqdan foydalaniladi, bunda dastlabki 5 ta ma’lumotlar ko‘rinadi.

```
df.head()
```

```
In [103…  df = pd.read_csv("vgsales new.csv")
          df.head()
```

Out[103…

	Rank	Name	Platform	Year	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales
0	259	Asteroids	2600	1980.0	Shooter	Atari	4.00	0.26	0.0	0.05	4.31
1	545	Missile Command	2600	1980.0	Shooter	Atari	2.56	0.17	0.0	0.03	2.76
2	1768	Kaboom!	2600	1980.0	Misc	Activision	1.07	0.07	0.0	0.01	1.15
3	1971	Defender	2600	1980.0	Misc	Atari	0.99	0.05	0.0	0.01	1.05
4	2671	Boxing	2600	1980.0	Fighting	Activision	0.72	0.04	0.0	0.01	0.77

Agar DataFrame Githubda turgan bo‘lsa unda Dataframeni faollashtirish uchun Githubdagi manzili bilan chaqirish talab etiladi.

```
df = pd.read_csv("https://raw.githubusercontent.com/abroraxatov1/dataset1/refs/heads/main/vgsales%20new.csv")
df.head()
```

3) Indeks bo‘yicha tartiblash. Bunda `df.sort_index()` orqali amalga oshiriladi.

```
df.sort_index()
```

Rank	Name	Platform	Year	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales
1	Wii Sports	Wii	2006.0	Sports	Nintendo	41.49	29.02	3.77	8.46	82.74
2	Super Mario Bros.	NES	1985.0	Platform	Nintendo	29.08	3.58	6.81	0.77	40.24
3	Mario Kart Wii	Wii	2008.0	Racing	Nintendo	15.85	12.88	3.79	3.31	35.82
4	Wii Sports Resort	Wii	2009.0	Sports	Nintendo	15.75	11.01	3.28	2.96	33.00
5	Pokemon Red/Pokemon Blue	GB	1996.0	Role-Playing	Nintendo	11.27	8.89	10.22	1.00	31.37

4) Ustun nomi bo‘yicha tartiblash. Bunda `df.sort_index(axis=1)` orqali amalga oshiriladi.

```
df.sort_index(axis=1)
```

Rank	EU_Sales	Genre	Global_Sales	JP_Sales	NA_Sales	Name	Other_Sales	Platform	Publisher	Year
259	0.26	Shooter	4.31	0.0	4.00	Asteroids	0.05	2600	Atari	1980.0
545	0.17	Shooter	2.76	0.0	2.56	Missile Command	0.03	2600	Atari	1980.0
1768	0.07	Misc	1.15	0.0	1.07	Kaboom!	0.01	2600	Activision	1980.0
1971	0.05	Misc	1.05	0.0	0.99	Defender	0.01	2600	Atari	1980.0
2671	0.04	Fighting	0.77	0.0	0.72	Boxing	0.01	2600	Activision	1980.0
...	...	...	...	...	...	...	...	...	...	...
16310	0.00	Racing	0.01	0.0	0.01	Freaky Flyers	0.00	GC	Unknown	NaN
16330	0.00	Shooter	0.01	0.0	0.01	Inversion	0.00	PC	Namco Bandai Games	NaN

5) Teskari tartiblash, agar *axis=1* parametrini qo‘llasak ustun bo‘yicha teskari tartiblanadi.

6) DataFrame elementlarini biror ustun bo‘yicha tartiblash ham mumkin.

```
df.sort_values(by='Name')
```

Rank	Name	Platform	Year	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales
8359	.hack//G.U. Vol.1//Rebirth	PS2	2006.0	Role-Playing	Namco Bandai Games	0.00	0.00	0.17	0.00	0.17
7109	.hack//G.U. Vol.2//Reminisce	PS2	2006.0	Role-Playing	Namco Bandai Games	0.11	0.09	0.00	0.03	0.23
8604	.hack//G.U. Vol.2//Reminisce (jp sales)	PS2	2006.0	Role-Playing	Namco Bandai Games	0.00	0.00	0.16	0.00	0.16
8306	.hack//G.U. Vol.3//Redemption	PS2	2007.0	Role-Playing	Namco Bandai Games	0.00	0.00	0.17	0.00	0.17

.rank() metodi yordamida DataFrame qatorlarini reytingini aniqlash mumkin. Reyting aniqlashda aynan qaysi ko‘rsatkich (ustun) bo‘yicha reytingni aniqlash ko‘rsatiladi.

```
df['rank']=df['Global_Sales'].rank()
```

df

Rank	Name	Platform	Year	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales	rank
259	509	0	1980.0	8	53	4.00	0.26	0.0	0.05	4.31	16340.5
545	5958	0	1980.0	8	53	2.56	0.17	0.0	0.03	2.76	16054.0
1768	4734	0	1980.0	3	21	1.07	0.07	0.0	0.01	1.15	14827.5
1971	2013	0	1980.0	3	53	0.99	0.05	0.0	0.01	1.05	14622.5
2671	1000	0	1980.0	2	21	0.72	0.04	0.0	0.01	0.77	13927.5
...	...	...	...	...	...	...	...	...	...	...	...
16310	3311	7	NaN	6	530	0.01	0.00	0.0	0.00	0.01	309.5
16330	4396	13	NaN	8	347	0.01	0.00	0.0	0.00	0.01	309.5
16369	3877	17	NaN	1	530	0.01	0.00	0.0	0.00	0.01	309.5
16430	10772	7	NaN	7	530	0.01	0.00	0.0	0.00	0.01	309.5
16496	10103	2	NaN	0	530	0.00	0.01	0.0	0.00	0.01	309.5

Dataset statistikasi

Statistik ma’lumotlar bilan ishlash. Dastlab githubdan DataFrame yuklab olish yoki tayyorini yuqorida tushuntirilgani kabi yuklab olish kerak. Bir nechta statistik funksiyalar bilan tanishib chiqamiz.

1) **.sum()** funksiyasi ustun elementlarining summasini ko‘rsatadi.

```
summa = df.sum()
print(summa)
```

```
Name            9.614228e+07
Platform        2.622150e+05
Year            3.275860e+07
Genre           8.179700e+04
Publisher       4.889813e+06
NA_Sales        4.392950e+03
EU_Sales        2.434130e+03
JP_Sales        1.291020e+03
Other_Sales     7.977500e+02
Global_Sales    8.920440e+03
rank            1.377551e+08
dtype: float64
```

2) .**mean**() - o‘rta qiymatni hisoblash funksiyasi.

```
mean = df.mean()
print(mean)
```

```
Name            5792.401193
Platform          15.797988
Year            2006.406443
Genre              4.928124
Publisher        294.602542
NA_Sales           0.264667
EU_Sales           0.146652
JP_Sales           0.077782
Other_Sales        0.048063
Global_Sales       0.537441
rank            8299.500000
dtype: float64
```

3) .**idxmin**() minimum qiymatini topib beradi.

4) **argmin()** kerakli ustun bo‘yicha minimum qiymatni qidirish.

```
df['rank'].argmin()
```

```
df['rank'].argmin()
```

```
304
```

5) Boshqa kerakli funksiyalar

- count - NaN bo‘lmagan qiymatlar sonini aniqlash
- median - qiymatlar medianini hisbolash
- mad - mean absolute deviation(mutlaq chetlanishni anglatadi)
- prod - barcha qiymatlar ko‘paytmasi
- var - variance (dispersiya)
- std - standart deviation (standart chetlanish)
- skew – skewness (og‘ish)
- kurt - kurtosis
- cumsum - cumulative sum
- cummin va cummax- cumulative min and max
- cumprod - cumulative product
- diff - arifmetik farq (difference)
- pct_change - percent changes

Dataset statistikasi: Umumlashtiruvchi ma’lumotlar

Dastlab DataFrameni Githubdan yuklab olingandan so‘ng yoki tayyorini yuklangandan so‘ng DataFrameni datlabki 5 ta elementini .head() parametri orqali, oxirgi 5 ta elementini tail() parametri orqali, ixtiyoriy 5 ta elementini .sample parametri orqali chop qilish mumkin.

1) **.info()** DataFrame haqida umumiy ma’lumot olish.

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 16598 entries, 259 to 16496
Data columns (total 11 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Name          16598 non-null  int64
 1   Platform      16598 non-null  int64
 2   Year          16327 non-null  float64
 3   Genre         16598 non-null  int64
 4   Publisher     16598 non-null  int64
 5   NA_Sales      16598 non-null  float64
 6   EU_Sales      16598 non-null  float64
 7   JP_Sales      16598 non-null  float64
 8   Other_Sales   16598 non-null  float64
 9   Global_Sales  16598 non-null  float64
 10  rank          16598 non-null  float64
```

2) **.describe**() DataFrame haqida tavsiflovchi statistik ma’lumot olish funksiyasi.

```
df.describe()
```

	Name	Platform	Year	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales
count	16598.000000	16598.000000	16327.000000	16598.000000	16598.000000	16598.000000	16598.000000	16598.000000	16598.000000
mean	5792.401193	15.797988	2006.406443	4.928124	294.602542	0.264667	0.146652	0.077782	0.048063
std	3323.525731	8.392298	5.828981	3.762015	178.087732	0.816683	0.505351	0.309291	0.188588
min	0.000000	0.000000	1980.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	2921.250000	7.000000	2003.000000	1.000000	137.000000	0.000000	0.000000	0.000000	0.000000
50%	5839.500000	16.000000	2007.000000	5.000000	323.000000	0.080000	0.020000	0.000000	0.010000
75%	8699.750000	21.000000	2010.000000	8.000000	461.000000	0.240000	0.110000	0.040000	0.040000
max	11492.000000	30.000000	2020.000000	11.000000	578.000000	41.490000	29.020000	10.220000	10.570000

count - NaN qiymatga ega bo‘lmagan qatorlar soni;
mean - ko‘rsatilgan ustundagi qiymatlarning o‘rtacha qiymati (faqat sonli ustunlar uchun);
std - ustun uchun standart chetlanish;

min - ustun uchun minimum;
25/50/75% - 25/50/75% oraliqdagi qiymatlar;
max - ustun uchun maksimum.

Korrelyatsiya

Korrelyatsiya yordamida kerakli ustunlarning bir-biriga bog‘liqligini tekshirish mumkin. Bunda ma’lumotlar tahlili va bashorati uchun aynan qaysi maydon qaysi maydon bilan yaxshi yoki yomon bog‘langanligini bilish uchun korrelyatsiyani aniqlash kerak bo‘ladi. Bog‘liqlik juda ham past yoki juda ham yuqori bo‘lgani bashorat uchun halaqit beradi. “df2” nomli DataFrame yaratib unga .corr funksiyasi qo‘llanilish orqali korrelyatsiya aniqlanadi.

```
[147] data = {
          'xotira' : [500, 250, 128, 1000, 2000],
          'tezlik' : [120, 100, 75, 150, 200]
      }

      df2 = pd.DataFrame(data)
```

```
[148] df2['xotira'].corr(df2['tezlik'])
```

```
0.9813527136115692
```

Yuqoridagi df2 DataFrameda ikkita maydon bor, ya’ni avtomobilning yoshi bilan yili bu albatta bir-biriga bog‘liq bo‘lganligi sababli 98% chiqarayapti.

.corr funksiyasini o‘zimizni “df” DataFramega qo‘llasak quyidagi natijani olamiz.

```
In [10]: df['Global_Sales'].corr(df['Other_Sales'])

Out[10]: 0.7483308464077957
```

Bunda bog‘liqlik 74% da aniqlanmoqda.

Agar DataFramening barcha ustunlarini korrelyasiyasi kerak bo‘lsa quyidagicha yo‘l tutamiz. Bunda har bir maydonni boshqa maydonlar bilan bog‘liqlik jadvali shakllantiriladi, natijada kerakli maydonga mos bog‘langan maydonlarni aniqlash imkoniyati yaratiladi.

```
df.corr()
```

	Name	Platform	Year	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales
Name	1.000000	0.014418	-0.001370	-0.004987	0.026475	0.012277	0.006542	0.015945	-0.006785	0.010888
Platform	0.014418	1.000000	0.167823	0.029061	-0.017938	0.041536	0.047158	-0.078207	0.055061	0.028213
Year	-0.001370	0.167823	1.000000	-0.124994	0.029976	-0.091402	0.006014	-0.169316	0.041058	-0.074735
Genre	-0.004987	0.029061	-0.124994	1.000000	-0.052097	0.017435	0.016481	0.030085	0.009731	0.021671
Publisher	0.026475	-0.017938	0.029976	-0.052097	1.000000	0.002125	0.010671	0.049783	0.011902	0.015865
NA_Sales	0.012277	0.041536	-0.091402	0.017435	0.002125	1.000000	0.767727	0.449787	0.634737	0.941047
EU_Sales	0.006542	0.047158	0.006014	0.016481	0.010671	0.767727	1.000000	0.435584	0.726385	0.902836
JP_Sales	0.015945	-0.078207	-0.169316	0.030085	0.049783	0.449787	0.435584	1.000000	0.290186	0.611816
Other_Sales	-0.006785	0.055061	0.041058	0.009731	0.011902	0.634737	0.726385	0.290186	1.000000	0.748331
Global_Sales	0.010888	0.028213	-0.074735	0.021671	0.015865	0.941047	0.902836	0.611816	0.748331	1.000000

Hamma ustunni bitta ustunga nisbatan solishtirish uchun **.corrwith** funksiyasi ishlatiladi.

```
df.corrwith(df['Global_Sales'])
```

	0
Name	0.010888
Platform	0.028213
Year	-0.074735
Genre	0.021671
Publisher	0.015865
NA_Sales	0.941047
EU_Sales	0.902836
JP_Sales	0.611816
Other_Sales	0.748331
Global_Sales	1.000000

Yuqorida “Global_Sales” maydoni qolgan maydonlar bilan bilan bog‘liqlik darajasi aniqlandi.

Ma’lumotlarni filterlash

Pythonda DataFramedagi ma’lumotlarni filterlash imkoniyati yaratilgan. DataFramedagi istalgan ustunni qiymatlarini takrorlamasdan ajratish uchun **.unique** funksiyasidan foydalaniladi. Quyida price maydonini qiymatlarini takrorlanmasdan ajratib olish uchun quyidagi kodni kiritish kerak.

```
sale = df['Year'].unique()
sale
```

```
sale = df['Year'].unique()
sale
```

```
array([1980., 1981., 1982., 1983., 1984., 1985., 1986., 1987., 1988.,
       1989., 1990., 1991., 1992., 1993., 1994., 1995., 1996., 1997.,
       1998., 1999., 2000., 2001., 2002., 2003., 2004., 2005., 2006.,
       2007., 2008., 2009., 2010., 2011., 2012., 2013., 2014., 2015.,
       2016., 2017., 2020.,   nan])
```

Agar ustun qiymatlarini ajratish kerak bo‘lsa **.value_counts** funksiyasidan foydalaniladi.

```
sale = df['Genre'].value_counts()
print(sale)
```

```
sale = df['Genre'].value_counts()
print(sale)
```

```
Genre
0     3316
10    2346
3     1739
7     1488
8     1310
1     1286
6     1249
4      886
9      867
2      848
11     681
5      582
Name: count, dtype: int64
```

DataFrameni filterlash uchun dastlab filter yaratib olish kerak bo‘ladi. Bunda models o‘zgaruvchisiga 2 ta qiymat olinib bu qiymatni filtr o‘zgaruvchisiga bog‘lab olingan.

```
yil = [2005]
filtr = df['Year'].isin(yil)
print(filtr)
```

```
0        False
1        False
2        False
3        False
4        False
```

So‘ngra ushbu filtrni DataFramega qo‘llash quyidagicha bo‘ladi.

```
df[filtr]
```

Rank	Name	Platform	Year	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales
11	Nintendogs	DS	2005.0	Simulation	Nintendo	9.07	11.00	1.93	2.75	24.76
12	Mario Kart DS	DS	2005.0	Racing	Nintendo	9.81	7.57	4.13	1.92	23.42
20	Brain Age: Train Your Brain in Minutes a Day	DS	2005.0	Misc	Nintendo	4.75	9.26	4.16	2.05	20.22
28	Brain Age 2: More Training in Minutes a Day	DS	2005.0	Puzzle	Nintendo	3.44	5.36	5.32	1.18	15.30
42	Animal Crossing: Wild World	DS	2005.0	Simulation	Nintendo	2.55	3.52	5.33	0.88	12.27
...	...	...	...	...	...	...	...	...	...	...

Ma’lumotlarni filtrlashda **mantiqiy shartlardan ham foydalanish** mumkin.

```
df[df.Year > 1995]
```

Rank	Name	Platform	Year	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales
5	Pokemon Red/Pokemon Blue	GB	1996.0	Role-Playing	Nintendo	11.27	8.89	10.22	1.00	31.37
47	Super Mario 64	N64	1996.0	Platform	Nintendo	6.91	2.85	1.91	0.23	11.89
64	Mario Kart 64	N64	1996.0	Racing	Nintendo	5.55	1.94	2.23	0.15	9.87
117	Crash Bandicoot	PS	1996.0	Platform	Sony Computer Entertainment	3.23	2.35	0.94	0.30	6.82
153	Tekken 2	PS	1996.0	Fighting	Sony Computer Entertainment	2.26	1.89	1.36	0.23	5.74
...	...	...	...	...	...	...	...	...	...	...

Ikkita shartni bittada kiritishimiz mumkin.

```
df[(df.Year > 1995) & (df.Genre == 'Action')]
```

Rank	Name	Platform	Year	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales
202	Resident Evil	PS	1996.0	Action	Virgin Interactive	2.05	1.16	1.11	0.73	5.05
230	Tomb Raider	PS	1996.0	Action	Eidos Interactive	2.29	1.97	0.13	0.24	4.63
326	Resident Evil Director's Cut	PS	1996.0	Action	Virgin Interactive	1.82	1.24	0.47	0.25	3.77
575	Star Wars: Shadows of the Empire	N64	1996.0	Action	Nintendo	2.00	0.50	0.12	0.03	2.65
1998	Mega Man X4	PS	1996.0	Action	Virgin Interactive	0.45	0.30	0.22	0.07	1.04
...	...	...	...	...	...	...	...	...	...	...

Bunda istalgan qiymatdagi ma’ulotlarni filtrlash mumkin.

```
df[df.Platform == 'X360']
```

Rank	Name	Platform	Year	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales
843	Call of Duty 2	X360	2005.0	Shooter	Activision	1.81	0.05	0.01	0.15	2.02
1532	Need for Speed: Most Wanted	X360	2005.0	Racing	Electronic Arts	1.00	0.17	0.02	0.10	1.29
2661	Perfect Dark Zero	X360	2005.0	Shooter	Microsoft Game Studios	0.66	0.02	0.03	0.06	0.77
3369	PGR3 - Project Gotham Racing 3	X360	2005.0	Racing	Microsoft Game Studios	0.49	0.03	0.03	0.05	0.60
4195	Dead or Alive 4	X360	2005.0	Fighting	Tecmo Koei	0.30	0.03	0.10	0.03	0.47
...	...	...	...	...	...	...	...	...	...	...

Nazariy savollar

1. Pandas kutubxonasi va uning vazifasi?
2. Pandas kutubxonasi afzalliklari?
3. Series ma’lumotlar tuzilmasi?
4. DataFrame ma’lumotlar tuzilmasi?
5. DataFrame: Qayta indekslash?
6. DataFrame va Seriesda elementlarni tanlash?
7. Pandas kutubxonasida arifmetik amallarni bajarilishi?
8. Pandas obyekti elementlariga funksiyalarni qo‘llash?
9. Tartiblash va Reytinglash?
10. Series obyektini Indeks bo‘yicha tartiblash?
11. Series obyektini qiymatlari bo‘yicha tartiblash?
12. Indeks bo‘yicha tartiblash?
13. Ustun nomi bo‘yicha tartiblash?
14. Teskari tartiblash?
15. DataFrame elementlarini biror ustun bo‘yicha tartiblash?

16. Dataset statistikasi?
17. sum() funksiyasi va uning vazifasi?
18. mean() funksiyasi va uning vazifasi?
19. idxmin()funksiyasi va uning vazifasi?
20. argmin()funksiyasi va uning vazifasi?
21. info() funksiyasi va uning vazifasi?
22. discribe() funksiyasi va uning vazifasi?
23. Korrelyatsiya va uning vazifasi?
24. Ma'lumotlarni filterlash?
25. unique funksiyasi va uning vazifasi?
26. Ma'lumotlarni filtrlashda mantiqiy shartlardan foydalanish?

1.6. Mashinaviy o‘qitishda ma’lumotlarga ishlov berish

Reja:

Fayllar bilan ishlash;
Fayllarga ma’lumotlarni yozish;
HDF5 formati;
Web fayllar bilan ishlash;
Ma’lumotlarni vizuallashtirish.

Fayllar bilan ishlash

Fayldan ma’lumotlarni o‘qish. Ma’lumotlarga ishlov berish uchun Pandas kutubxonasida fayllar bilan ishlaydigan bir nechta funksiyalar mavjud.

- **read_csv** - vergul bilan ajratilgan jadval fayllarni o‘qish.
- **read_table** - probel (bo‘sh joy) bilan ajratilgan jadval fayllarni o‘qish.
- **read_excel** - xls yoki xlsx formatidagi **Excel** fayllarni o‘qish.
- **read_clipboard** - ko‘chirilgan (xotiradagi) jadvalni o‘qish.
- **read_hdf** - HDF5 formatidagi fayllarni o‘qish.
- **read_html** - HTML (web) sahifadagi barcha jadvallarni o‘qish.
- **read_json** - JSON ma'lumotlarni o‘qish.
- **read_pickle** - pickle formatidagi fayllarni o‘qish.
- **read_sql** - SQL query dan qaytgan ma'lumotlarni o‘qish (SQLAlchemy bilan ishlaydi).

Yuqoridagi funksiyalarning deyarli barchasining ishlashi juda o‘xshash, muhimi, to‘g‘ri formatdagi faylni ko‘rsatish va turli qo‘shimcha parametrlardan foydalangan holda jadvalni to‘g‘ri yaratib olish hisoblanadi. Tajriba olish uchun tayyor DataFramedan foydalanish mumkin, agar tayyor bo‘lmasa DataFrame hosil qilish kerak bo‘ladi. DataFrame hosil qilish uchun avval massivli yozuv quyidagicha hosil qilinadi.

```
import pandas as pd
data={'opxotira':[2,4,2,8],'yil':[2020,2021,2022
    ,2023], 'narxi':[1200,1400,1300,1600]}
```

Yuqoridagi kodda bitta kompyuter uchun 3 ta maydonli ma’lumotlar shakllantirilgan. Bu ma’lumotlarni DataFramega o‘tkazish uchun quyidagi kod yoziladi.

```
df=pd.DataFrame(data)
```

Yuqoridagi kod orqali DataFrame tayyor qilinadi, tayyor DataFrameni excel formatda yuklab olish uchun quyidagi kod yoziladi.

```
df.to_excel('dataset.xlsx', index=False)
```

Yuqorida `index=False` parametr DataFramening indekslangan qismini excel faylga o‘tkazmaydi.

O‘qib olinadigan fayl kompyuterda yoki onlayndagi fayl bo‘lishi mumkin. Bunda Githubda joylashgan turli fayllardan foydalanish yoki kerakli faylni kompyuterga yuklab olib, Jupyter Notebookda ochib foydalanish mumkin. Agar kerakli fayl Jupyter Notebook fayli joylashgan papkada bo‘lsa shunchaki nomini yozish yetarli hisoblanadi. Colabda ishlayotganlar uchun Dataframeni Colabning Fayl qismidan **.csv** fayli yuklanadi, keyin quyidagi kod bilan **“df”** Dataframe faollashtiriladi. Jupyter Notebookda ishlayotganlar DataFrameni manzili bilan to‘liq yozish kerak bo‘ladi.

```
df = pd.read_csv("Uy_baza.csv")
df=pd.read_excel("dataset.xlsx")
```

DataFrameni qaysidir qismidagi ma’lumotlarni ko‘rish uchun quyidagi buruqdan foydalaniladi, bunda dastlabki 3 ta ma’lumotlar ko‘rinadi.

```
df.head(3)
```

	opxotira	yil	narxi
0	2	2020	1200
1	4	2021	1400
2	2	2022	1300

Pandas yangi ochilgan jadvalning qatorlariga avtomat ravishda 0 dan boshlab indeks beradi. Agar siz indeks sifatida fayldagi jadvalning biror ustunini ishlatmoqchi bo‘lsangiz, ustun tartib raqamini **index_col** parametri yordamida ko‘rsatishingiz mumkin.

```
df=pd.read_excel("dataset.xlsx", index_col=2)
df.head()
```

	opxotira	yil
narxi		
1200	2	2020
1400	4	2021
1300	2	2022
1600	8	2023

.nrows parametri yordamida jadvaldan faqatgina sanoqli qatorlarni o‘qish mumkin. Bu katta jadvallar bilan ishlash uchun qulay.

```
df=pd.read_excel("dataset.xlsx", nrows=2)
df.head()
```

	opxotira	yil	narxi
0	2	2020	1200
1	4	2021	1400

Pandasda **fayllarni o‘qishda 50 ga yaqin parametrlarni ko‘rsatish** mumkin. To‘liq ro‘yxat bilan quyidagi sayt orqali tanishish mumkin: www.pandas.pydata.org.

Fayllarga ma’lumotlarni yozish

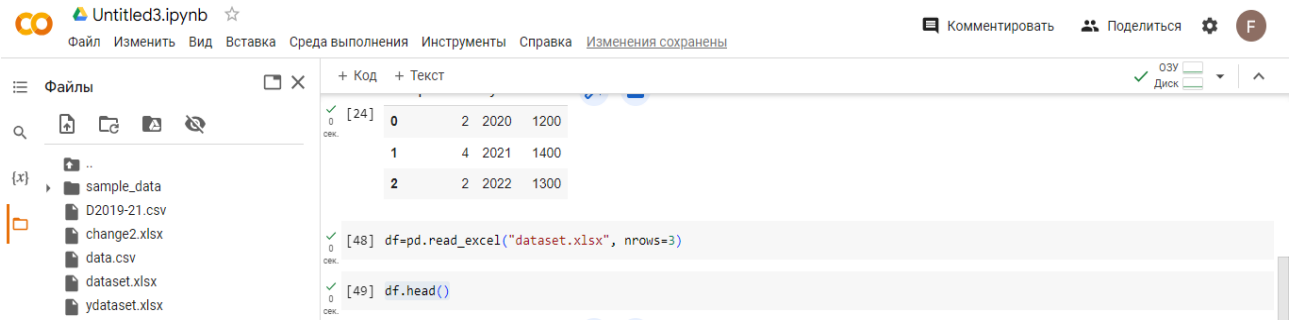
Fayllarga ma’lumotlarni yozish uchun dastlab birorta faylni chaqirib olish kerak bo‘ladi.

```
df=pd.read_excel("dataset.xlsx", nrows=3)
df.head()
```

	opxotira	yil	narxi
0	2	2020	1200
1	4	2021	1400
2	2	2022	1300

Faylni chaqirib olgandan so‘ng **“to”** buyrug‘i orqali biror faylga yoziladi, bunda yuqorida keltirilgan faqat 3 satrli ma’lumot ydata.xlsx nomli excel faylga yoziladi, va yangi fayl colabda fayllar qatoriga tushadi.

```
df.to_excel('ydataset.xlsx')
```



Series obyektini ham faylga yozishimiz mumkin. Dastlab, Series obyektini yaratib olamiz.

```
Ввод [37]: obj=df.country
           obj
```

```
Out[37]: 0        Macau
         1       Monaco
         2    Singapore
         3    Hong Kong
         4    Gibraltar
         5      Bahrain
         Name: country, dtype: object
```

```
Ввод [42]: obj.to_csv('country.csv')
```

```
Ввод [43]: df=pd.read_csv('country.csv')
           df
```

Out[43]:

	Unnamed: 0	country
0	0	Macau
1	1	Monaco
2	2	Singapore
3	3	Hong Kong
4	4	Gibraltar
5	5	Bahrain

Biz DataFrameni faylga saqlashda faqat kerakli ustunlarini saqlashimiz mumkin.

```
Ввод [48]: df.to_csv('colum.csv', columns=['country','pop2021','area'])
```

```
Ввод [50]: df=pd.read_csv('colum.csv')
           df
```

Out[50]:

	Unnamed: 0	country	pop2021	area
0	0	Macau	658.394	30
1	1	Monaco	39.511	2
2	2	Singapore	5896.686	710
3	3	Hong Kong	7552.810	1104
4	4	Gibraltar	33.698	6
5	5	Bahrain	1748.296	765

HDF5 formati

HDF5 formati ilmiy tadqiqodlar qilishda katta hajmdagi ma’lumotlarni saqlash uchun keng ishlatiladigan formatlardan hisoblanadi. **HDF5** fayllarni nafaqat Python, balki Java, Julia, Matlab va boshqa dasturlash tillarida ochish mumkin. HDF - **hierarchial data format** (ierarxiyali ma'lumotlar formati) hisoblanib, tarkibida bir nechta datasetlarni siqilgan holatda saqlashi mumkin. HDF formatida fayllarni saqlash uchun avval HDF obyketi yaratiladi.

```
hd=pd.HDFStore("data.h5")
```

“**hd**” fayliga **.put** metodi yordamida “**df**” va “**df2**” DataFramelarini mos ravishda ‘komp’ va ‘komp2’ kalit so‘zlari bilan yozish orqali, birlashtiriladi.

```
hd.put('komp',df)
hd.put('komp2',df2)
```

“hd” faylidagi DataFramega murojaat qilish uchun quyidagicha yo‘l tutamiz.

```
[63] hd['komp']
```

	opxotira	yil	narxi
0	2	2020	1200
1	4	2021	1400
2	2	2022	1300

```
[64] hd['komp2']
```

	dxotira	yil	narxi
0	200	2020	1250
1	400	2021	1450
2	250	2022	1350
3	500	2023	1650

Web fayllar bilan ishlash

Web sahifalardagi ma’lumotlarni intellektual qayta ishlash uchun, web ma’lumotlarni DataFramelarga o‘qib olish imkoniyati mavjud. Buning uchun **Read_html()** funksiyasi ishlatiladi. **Read_html()** - web sahifadagi barcha jadvallarni o‘qib, DataFramelardan iborat ro‘yxat ko‘rinishida qaytaradi.

```
data3=pd.read_html("https://en.wikipedia.org/
wiki/Registan")
data3
```

```
data3
```

	0	1	vteIslamic art.2
0	Lakes	Aydar Lake Lake Charvak	
1	National parks	Chatkalskiy State Nature Reserve Zaamin Nation...	
0	vteIslamic art	vteIslamic art.1	\
1	Architecture	Regional styles Abbasid Ayyubid Azerbaijani Ch...	
2	Regional styles	Abbasid Ayyubid Azerbaijani Chinese Fatimid Ha...	
3	Elements	Ablaq Banna'i Iwan Jali Mashrabiya Mihrab Mina...	
4	Arts	Regional styles Bangladeshi Persian (Early, Qa...	
5	Regional styles	Bangladeshi Persian (Early, Qajar, Safavid) Tu...	
6	Carpets	Gul Kilim Motifs Persian Turkish Prayer	
7	Pottery	Fritware Hispano-Moresque İznik Lustreware Min...	
8	Textiles	Batik Damask Ikat Embroidery Soumak Suzani	
9	Woodwork	Khatam Minbar	
10	Other media	Music Brass Damascus steel Enamelled glass Gla...	
11	Arts of the book	Miniatures Arabic Mughal Ottoman Persian Calli...	
12	Miniatures	Arabic Mughal Ottoman Persian	
13	Calligraphy	Arabic Diwani Indo-Muslim Kufic Muhaqqaq Naskh...	
14	Other arts	Muraqqa Hilya Ottoman illumination	
15	Decoration	Arabesque Geometric patterns Girih (tiles) Zel...	
16	The garden	Charbagh Mughal Ottoman Paradise Persian	
17	Museums, collections	Berlin Cairo Doha Ghazni Istanbul (Arts, Calli...	
18	Principles, influences	Islamic Art: Mirror of the Invisible World Ani...	

“data” tarkibida sahifadagi barcha jadvallar alohida DataFrame ko‘rinishida saqlangan. Jadvallar sonini ko‘rish uchun **len()** funksiyasiga murojaat qilamiz.

```
[74] data3=pd.read_html("https://en.wikipedia.org/wiki/Registan")

len(data3)

10
```

Yuqoridagi data tarkibida 10 ta DataFrame bor ekan. Alohida DataFramelarga indeks orqali murojaat qilishimiz mumkin. Ba'zida kerakli jadvallarni topguncha bir necha indekslarni qarab chiqish talab qilinishi mumkin.

Ma’lumotlarni vizuallashtirish

Qayta ishlanadigan ma’lumotlarni jadval, grafik yoki diagramma ko‘rinishida ifodalash uchun Pythonning Jupyter Notebook yoki Colabda amalga oshirish imkoniyatlari mavjud. Ma’lumotlarni jadval, grafik yoki diagramma ko‘rinishida vizuallashtirish uchun quyidagi dataframeni faollashtiramiz va shu dataframeni vizuallashtirish imkoniyatlarini qarab o‘tamiz.

```
data={'opxotira':[2,4,2,8],'yil':[2020,2021,2022
      ,2023], 'narxi':[1200,1400,1300,1600]}
                df=pd.DataFrame(data)
```

Yuqoridagi **df** dataframening ma’lumotlarini vizuallashtirish uchun df quyidagi kod orqali faollashtiriladi.

```
df
```

Natija quyidagicha bo‘ladi.

```
df
```

	opxotira	yil	narxi
0	2	2020	1200
1	4	2021	1400
2	2	2022	1300
3	8	2023	1600

Yuqorida dataframening tuzilmasi shakllantirildi. Bu dataframeni jadval ko‘rinishida ifodalash uchun quyidagi buyruq beriladi.

```
df
```

	opxotira	yil	narxi
0	2	2020	1200
1	4	2021	1400
2	2	2022	1300
3	8	2023	1600

Natijada ma’lumotni quyidagi jadval ko‘rinishi ifodalanadi.

1 to 4 of 4 entries Filter

index	opxotira	yil	narxi
0	2	2020	1200
1	4	2021	1400
2	2	2022	1300
3	8	2023	1600

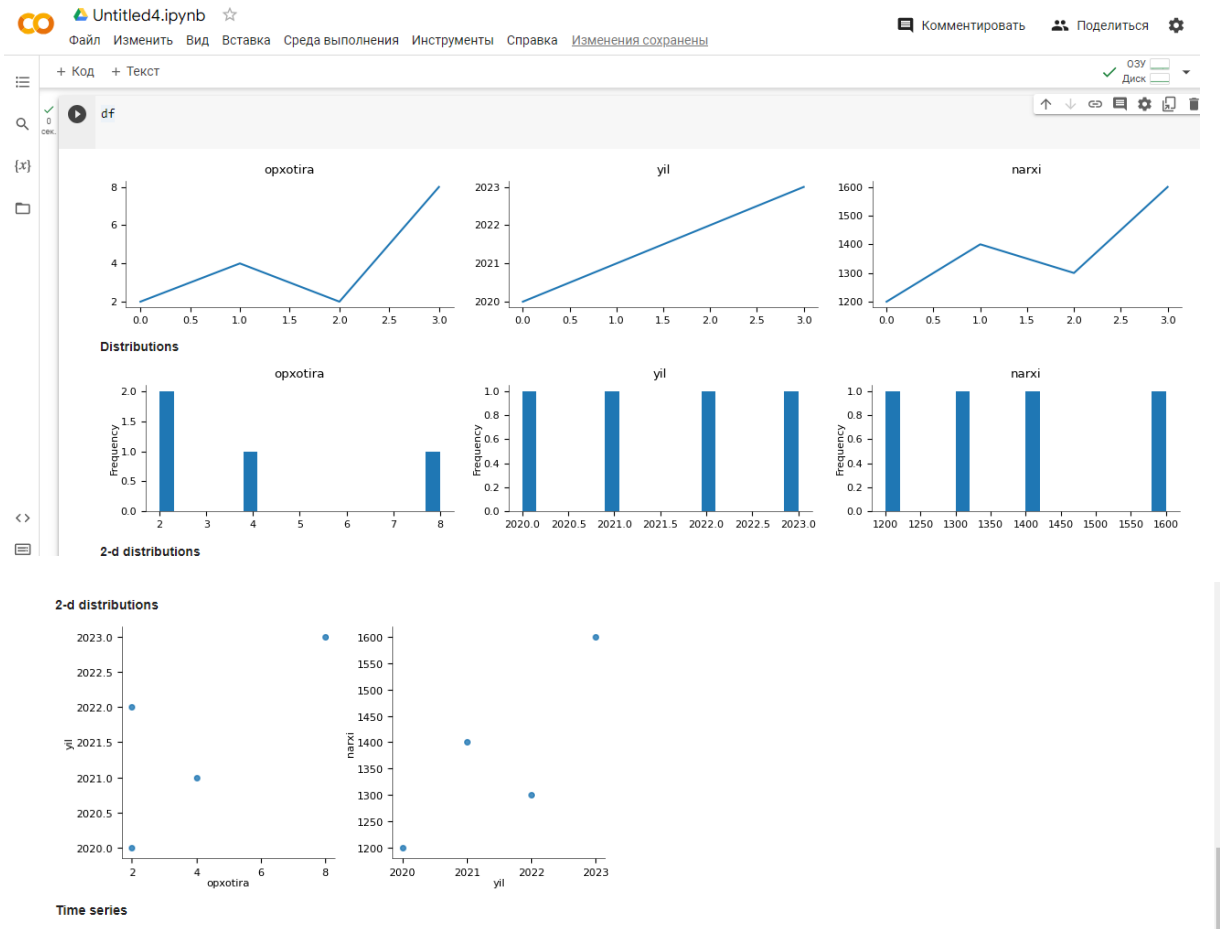
Show 25 per page

Like what you see? Visit the data table notebook to learn more about interactive tables.

Ma’lumotni diagramma va grafik ko‘risnishida ifodalash uchun quyidagi buyruq beriladi.

```
df
```

	opxotira	yil	narxi
0	2	2020	1200
1	4	2021	1400
2	2	2022	1300
3	8	2023	1600



Qayta ishlangan ma’lumotlarni yuqoridagi buyruqlar orqali to‘liq vizuallashtirish imkoniyati yaratiladi.

Nazariy savollar

1. Fayllar bilan ishlash jarayonini tushuntiring?

2. Fayllardan ma’lumot o‘qish jarayonini tushuntiring?

3. Fayllarga ma’lumotlarni yozish jarayonini tushuntiring?

4. Dataframe yaratish jarayonini tushuntiring?

3. HDF5 formatini tushuntiring?

4. Web fayllar bilan ishlash jarayonini tushuntiring?

5. Ma’lumotlarni vizuallashtirish jarayonini tushuntiring?

8. Ma’lumotlarni jadval ko‘rinishida ifodash jarayonini tushuntiring?

9. Ma’lumotlarni diagramma ko‘rinishida ifodash jarayonini tushuntiring?

10. Ma’lumotlarni grafik ko‘rinishida ifodash jarayonini tushuntiring?

II-BOB. MASHINAVIY O‘QITISH, MASHINAVIY O‘QITISHNING REGRESSIYA ALGORITMLARI

Mazkur bobda sun’iy intellekt uchun mashinaviy o‘qitish algoritmlari va ularni turlari keltirib o‘tilgan. Mashinaviy o‘qitish algoritmlari va ularni ishlash funksiyalarining tahlillari keltirib o‘tilgan.

2.1. Mashinaviy o‘qitish. Mashinaviy o‘qitish turlari

Reja:

Mashinaviy o‘qitish;
Nazoratli o‘qitish;
Regressiya;
Klassifikatsiyalash (Tasniflash);
Nazoratsiz mashinaviy o‘qitish;
Nazoratsiz o‘qitishning ishlash funksiyasi;
Nazoratsiz o‘qitish algoritmi turlari;
Nazoratsiz o‘qitishning afzalliklari;
Nazoratsiz o‘qitishning kamchiliklari.

Mashinaviy o‘qitish

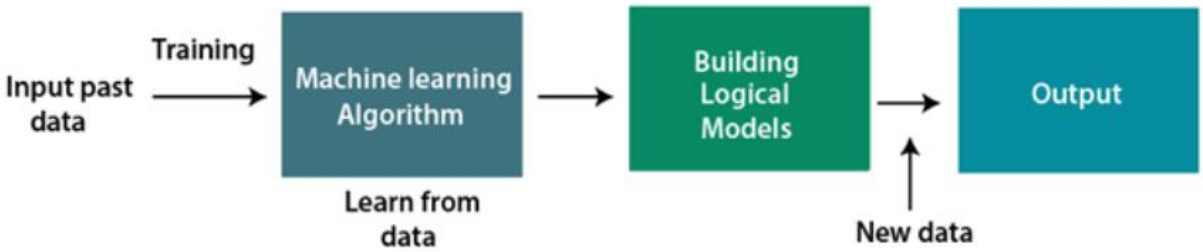
Mashinaviy o‘qitish - *bu kompyuterlarga mavjud ma'lumotlardan avtomatik ravishda o‘rganish asosida xulosa beruvchi o‘sib borayotgan texnologiya hisoblanadi.* Mashinaviy o‘qitish tarixiy ma'lumotlar yoki ma'lumotlardan foydalanib, matematik modellarni yaratish va bashorat qilish uchun turli xil algoritmlardan foydalanadi. Hozirda u tasvirni aniqlash , nutqni aniqlash , elektron pochtani filtrlash, tavsiyalar tizimi, iqtisodiy masalalarni bashoratlash va boshqalar kabi turli vazifalar uchun qo‘llanilmoqda.

Mashinaviy o‘qitish matematik statistika, optimallashtirish usullari va klassik matematik fanlarning birlashmasi, lekin hisoblash samaradorligi va qayta tayyorlash muammolari bilan bog‘liq o‘ziga xos xususiyatlarga ega bo‘ladi. Klassik statistik yondashuvlarga alternativ sifatida ko‘plab induktiv o‘qitish usullari ishlab chiqilgan. Ko‘pgina usullar ma’lumotlar qidirish bilan chambarchas bog‘liq (Data mining) bo‘ladi. Mashinaviy o‘qitish nafaqat matematik, balki amaliy, muhandislik fani hisoblanadi. Sof nazariya, qoida tariqasida, amalda qo‘llaniladigan usul va algoritmlarni darhol olib kelmaydi. Ularni yaxshi ishlashi uchun ehtimollar nazariyasida yuzaga kelgan nomuvofiqlikni haqiqiy muammolar sharoitlari bilan qoplash uchun qo‘shimcha evristik ixtiro qilish kerak bo‘ladi. Mashinaviy o‘qitish bo‘yicha deyarli har

qanday tadqiqot usulining amaliyligini tasdiqlovchi model yoki haqiqiy ma'lumotlarga nisbatan eksperimentsiz to‘liq bo‘lmaydi.

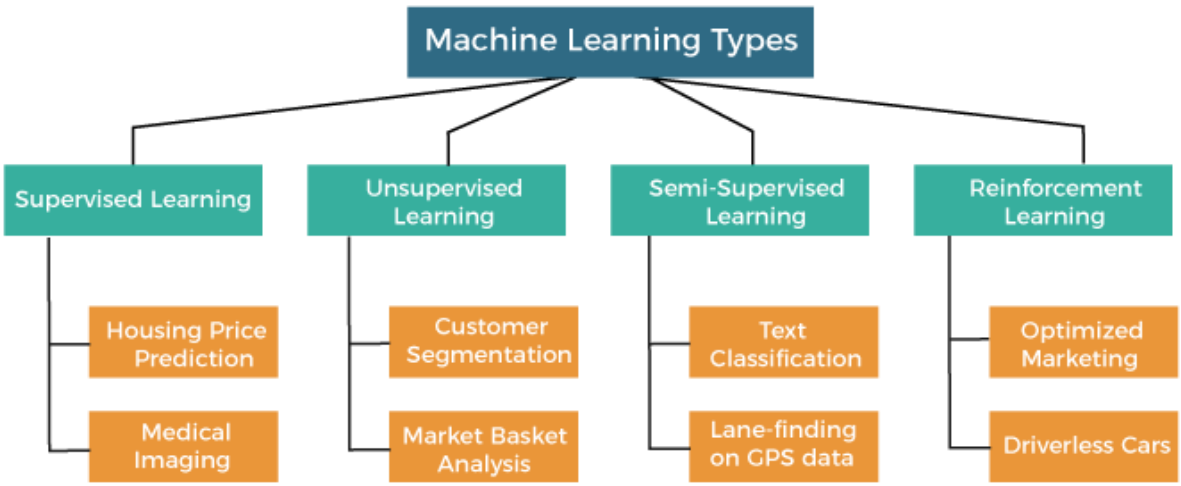
Mashinaviy o‘qitishni ishlash funksiyasi. Mashinaviy o‘qitish tizimi oldingi, daslabki ma'lumotlardan o‘qitiladi, bashorat modellarini quradi va har safar yangi ma’lumotlarni olganida, uning natijalarini bashorat qiladi. Prognoz qilingan chiqishning aniqligi ma’lumotlar miqdoriga bog‘liq bo‘ladi, chunki katta hajmdagi ma'lumotlar chiqishni aniqroq bashorat qiladigan yaxshiroq modelni yaratishga yordam beradi.

Aytaylik, bizda murakkab muammo bor, u yerda biz ba’zi bashoratlarni amalga oshirishimiz kerak, shuning uchun unga kod yozish o‘rniga, biz ma’lumotlarni umumiy algoritmlarga berishimiz kerak va bu algoritmlar yordamida mashina mantiqni o‘z ichiga oladi. Mashinaviy o‘qitish muammo haqida fikrlash tarzimizni o‘zgartirishga qodir bo‘ladi. Quyidagi blok sxemada mashinaviy o‘qitish algoritmining ishlash funksiyasi shakllantirilgan.



Mashinaning qay tarzda foydalanilishiga nisbatan mashinaviy o‘qitish 4 turga bo‘linadi.

- Nazoratli o‘qitish (Supervised Machine Learning)
- Nazoratsiz o‘qitish (Unsupervised Machine Learning)
- Yarim nazoratli o‘qitish (Semi-Supervised Machine Learning)
- Kuchaytirilgan o‘qitish (Reinforcement Learning)



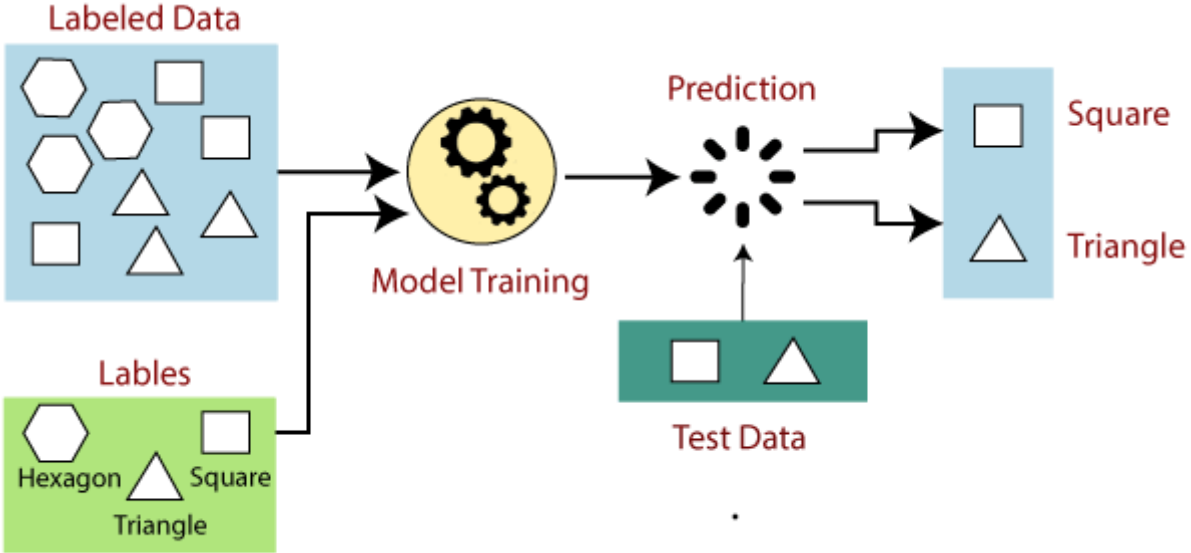
Nazoratli o‘qitish

Nazoratli o‘qitish - bu mashinaviy o‘qitish turlari bo‘lib, unda mashinalar yaxshi "shakllantirilgan" o‘quv ma'lumotlaridan foydalangan holda o‘qitiladi va shu ma'lumotlar asosida mashinalar natijani bashorat qiladilar.

Nazoratli o‘qitish - bu mashinaviy o‘qitish modeliga kirish ma'lumotlari va to‘g‘ri chiqish ma'lumotlarini taqdim etish jarayonini amalga oshirib beradi. Nazoratli o‘qitish algoritmining maqsadi kirish o‘zgaruvchisi (x), chiqish o‘zgaruvchisi (y) bilan taqqoslash uchun masshtablash funksiyasini topish hisoblanadi.

Nazoratli o‘qitishni ishlash funksiyasi. Nazoratli o‘qitishda modellar yaxshi "shakllantirilgan" ma'lumotlar to‘plamidan foydalangan holda o‘qitiladi, bunda model har bir ma'lumot turini o‘rganadi. O‘qitish jarayoni tugallangandan so‘ng, model test ma’lumotlari (o‘quv to‘plamining kichik to‘plami) asosida sinovdan o‘tkaziladi va keyin u natijani bashorat qiladi.

Nazoratli o‘qitishning ishlashini quyidagi misol va sxema orqali osongina tushunish mumkin.



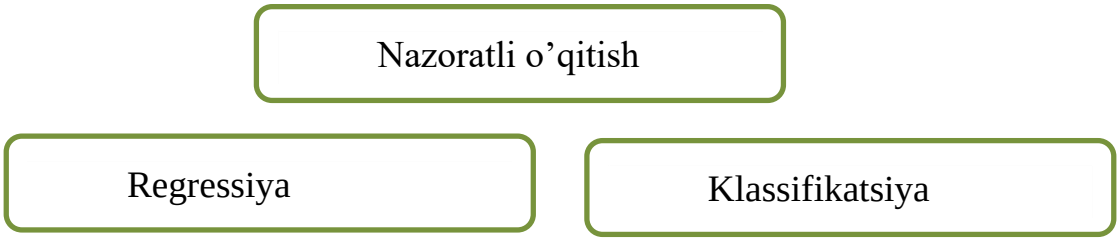
Aytaylik, bizda kvadrat, to‘rtburchak, uchburchak va ko‘pburchakni o‘z ichiga olgan har xil turdagi shakllarning ma'lumotlar to‘plami mavjud. Endi birinchi qadam - har bir shakl uchun modelni o‘rgatishimiz kerak.

- Agar berilgan shaklning to‘rt tomoni bo‘lsa va barcha tomonlari teng burchaklari 90 gradus bo‘lsa, u kvadrat sifatida bo‘ladi.
- Agar berilgan shaklning uchta tomoni bo‘lsa, u uchburchak sifatida belgilanadi.

- Agar berilgan shakl oltita teng tomonlarga ega bo‘lsa, u olti burchakli deb belgilanadi.
- Endi, mashg‘ulotdan so‘ng, biz test majmuasi yordamida modelimizni sinab ko‘ramiz.

Mashina barcha turdagi shakllar bo‘yicha allaqachon o‘rganilgan va u yangi shaklni topganda, u shaklni bir qator tomonlarning asoslari bo‘yicha tasniflaydi va chiqishni taxmin qiladi.

Nazoratli o‘qitishni yana ikki turdagi muammolarga bo‘lish mumkin:



Regressiya

Agar kirish o‘zgaruvchisi va chiqish o‘zgaruvchisi o‘rtasida bog‘liqlik mavjud bo‘lsa, regressiya algoritmlari qo‘llaniladi. U ob-havo prognozi, bozor tendensiyalari va boshqalar kabi uzluksiz o‘zgaruvchilarni bashorat qilish uchun ishlatiladi. Quyida nazoratli o‘qitishning ba’zi mashhur Regressiya algoritmlari keltirilgan:

- Oddiy chiziqli regressiya algoritmi (Simple Linear Regression Algorithm)
- Ko‘p o‘lchovli regressiya algoritmi (Multivariate Regression Algorithm)
- Qaror daraxti algoritmi (Decision Tree Algorithm)
- Lasso regressiyasi (Lasso Regression)

Klassifikatsiyalash (Tasniflash)

Tasniflash algoritmlari chiqish o‘zgaruvchisi kategorik bo‘lganda qo‘llaniladi, ya'ni Ha-Yo‘q, Erkak-Ayol, Haqiqiy-Noto‘g‘ri va boshqalar kabi ikkita sinf mavjud va ularga quyidagi algoritmlar kiradi.

- Tasodifiy o‘rmon algoritmi (Random Forest Algorithm)
- Qaror daraxti algoritmi (Decision Tree Algorithm)
- Logistik regressiya algoritmi (Logistic Regression Algorithm)
- Vektorli mashina algoritmini qo‘llab-quvvatlash (Support Vector Machine Algorithm)

Nazoratli o‘qitishning afzalliklari:

- Nazoratli o‘qitish yordamida model oldingi tajribalar asosida natijani bashorat qilishi mumkin.

- Nazoratli o‘qitishda biz ob’ektlar sinflari haqida aniq tasavvurga ega bo‘lishimiz mumkin.
- Nazoratli o‘qitish modeli bizga firibgarlikni aniqlash, spamni filtrlash va boshqalar kabi turli xil real muammolarni hal qilishda yordam beradi .

Nazoratli o‘qitishning kamchiliklari:

- Nazoratli o‘qitish o‘quv modellari murakkab vazifalarni bajarish uchun mos emas.
- Agar test ma’lumotlari o‘quv ma'lumotlar to‘plamidan farq qilsa, nazoratli o‘qitish to‘g‘ri natijani bashorat qila olmaydi.
- Trening ko‘p hisoblash vaqtlarini talab qildi.
- Nazoratli o‘qitishda bizga ob’ektlarning sinflari haqida yetarli bilim kerak.

Nazoratsiz mashinaviy o‘qitish

Biz nazoratli mashinaviy o‘qitishni tahlil qildik, unda modellar o‘quv ma’lumotlari nazorati ostida mavjud ma’lumotlardan foydalangan holda o‘qitiladi. Ammo bizda shakllantirilgan ma’lumotlar bo‘lmagan va berilgan ma'lumotlar to‘plamidan yashirin naqshlarni topish kerak bo‘lgan kabi ko‘p holatlar bo‘lishi mumkin. Shunday qilib, mashinaviy o‘qitishda bunday holatlarni hal qilish uchun bizga nazoratsiz o‘qitish usullari kerak bo‘ladi. Nomidan ko‘rinib turibdiki, nazoratsiz o‘qitish - bu mashinaviy o‘qitish usuli bo‘lib, unda modellar o‘quv ma’lumotlar to‘plami yordamida nazorat qilinmaydi. Buning o‘rniga, modellarning o‘zi berilgan ma'lumotlardan yashirin naqsh va tushunchalarni topadi. Uni yangi narsalarni o‘rganish paytida inson miyasida sodir bo‘ladigan o‘rganish bilan solishtirish mumkin. Buni quyidagicha aniqlash mumkin.

Nazoratsiz o‘qitishni regressiya yoki tasniflash muammosiga to‘g‘ridan-to‘g‘ri qo‘llash mumkin emas, chunki nazoratli o‘qitishdan farqli o‘laroq, bizda kirish ma’lumotlari mavjud, lekin tegishli chiqish ma’lumotlari yo‘q bo‘ladi. Nazoratsiz o‘qitishning maqsadi **ma’lumotlar to‘plamining asosiy tuzilishini topish, ma'lumotlarni o‘xshashliklari bo‘yicha guruhlash va ushbu ma'lumotlar to‘plamini siqilgan formatda taqdim etishdan iborat.**

Misol: Faraz qilaylik, nazoratsiz o‘qitish algoritmiga har xil turdagi mushuk va itlarning tasvirlarini o‘z ichiga olgan kirish ma’lumotlar to‘plami berilgan. Algoritm hech qachon berilgan ma’lumotlar to‘plamida o‘qitilmaydi, ya’ni u ma’lumotlar to‘plamining xususiyatlari haqida hech qanday tasavvurga ega emas. Nazoratsiz o‘qitish

algoritmining vazifasi tasvir xususiyatlarini o‘z-o‘zidan aniqlashdir. Nazoratsiz o‘qitish algoritmi ushbu vazifani tasvirlar o‘rtasidagi o‘xshashliklarga ko‘ra tasvir ma'lumotlar to‘plamini guruhlarga ajratish orqali amalga oshiradi.

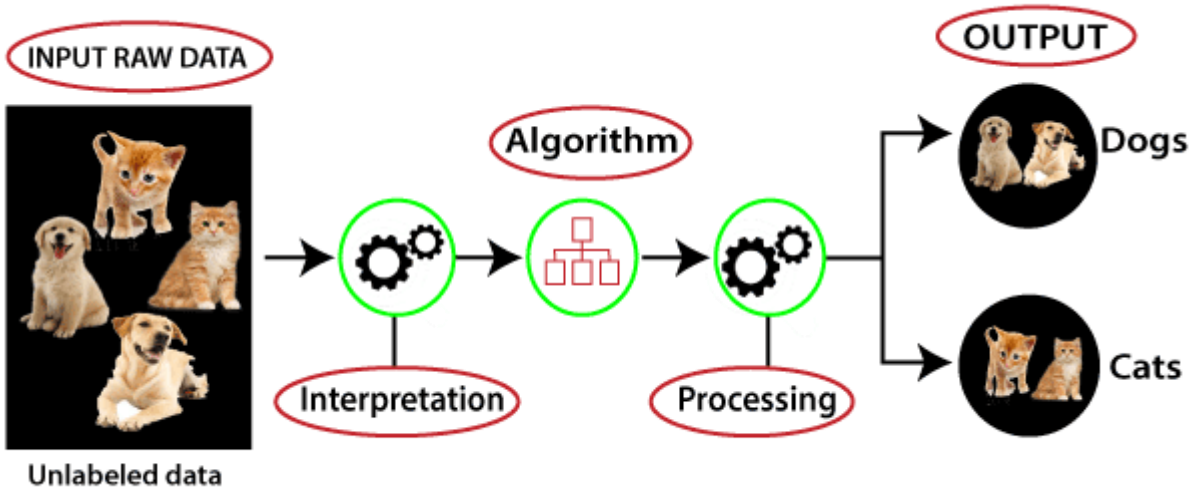
Nima uchun nazoratsiz o‘qitishdan foydalanish kerak?

Quyida nazoratsiz o‘qitishning ahamiyatini tavsiflovchi asosiy sabablar keltirilgan:

- Nazoratsiz o‘qitish ma'lumotlardan foydali tushunchalarni topishda yordam beradi.
- Nazoratsiz o‘qitish, odam o‘z tajribasi orqali fikrlashni o‘rganganiga o‘xshaydi, bu esa uni haqiqiy sun’iy intellektga yaqinlashtiradi.
- Nazoratsiz o‘qitish shakllantirilmagan va toifalanmagan ma’lumotlar ustida ishlaydi, bu esa nazoratsiz o‘qitishni muhimroq qiladi.
- Haqiqiy dunyoda bizda har doim ham mos keladigan chiqish bilan kirish ma'lumotlari mavjud emas, shuning uchun bunday holatlarni hal qilish uchun bizga nazoratsiz o‘qitish kerak bo‘ladi.

Nazoratsiz o‘qitishning ishlash funksiyasi

Nazoratsiz o‘qitishning ishlashini quyidagi sxema orqali tushunish mumkin:



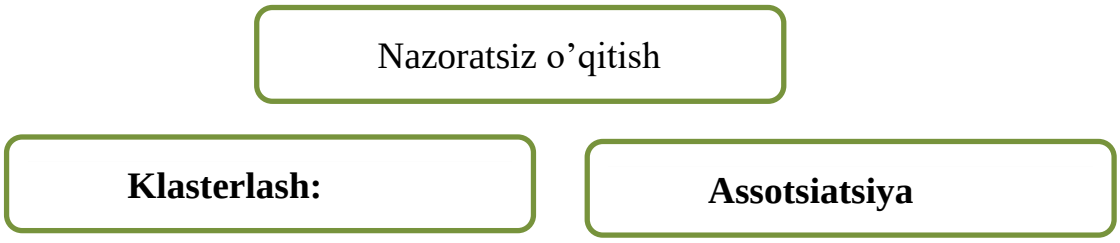
Bu yerda biz shakllantirilamagan kirish ma’lumotlarini oldik, ya’ni u toifalarga ajratilmagan va tegishli natijalar ham berilmagan. Endi ushbu shakllantirilamagan kirish ma’lumotlari uni o‘qitish uchun mashinaviy o‘qitish modeliga beriladi. Birinchidan, u ma'lumotlardan yashirin naqshlarni topish uchun yetarli bo‘lmagan ma’lumotlarni sharhlaydi va

keyin k-klasterlash, qarorlar daraxti va boshqalar kabi mos algoritmlarga qo‘llaniladi.

Tegishli algoritmni qo‘llaganidan so‘ng, algoritm ma’lumotlar ob’ektlarini ob’ektlar orasidagi o‘xshashlik va farqlarga ko‘ra guruhlarga ajratadi.

Nazoratsiz o‘qitish algoritmi turlari

Nazoratsiz o‘qitish algoritmini yana ikkita turdagi muammolarga bo‘lish mumkin:



Klasterlash: Klasterlash - bu obyektlarni klasterlarga guruhlash usuli bo‘lib, eng ko‘p o‘xshashliklarga ega bo‘lgan obyektlar guruhda qoladi va boshqa guruh ob’ektlari bilan kamroq yoki umuman o‘xshashliklarga ega emas obyektlar ajratiladi. Klaster tahlili ma’lumotlar obyektlari o‘rtasidagi umumiylikni topadi va ularni ushbu umumiyliklarning mavjudligi va yo‘qligi bo‘yicha toifalarga ajratadi.

Assotsiatsiya : Assotsiatsiya qoidasi - bu katta ma'lumotlar bazasidagi o‘zgaruvchilar o‘rtasidagi munosabatlarni topish uchun foydalaniladigan nazoratsiz o‘qitish usuli. U ma'lumotlar to‘plamida birgalikda yuzaga keladigan elementlar to‘plamini aniqlaydi.

Nazoratsiz o‘qitish algoritmlari:

Nazoratsiz o‘qitish algoritmlari berilgan ma’lumotlar ichidan qonuniyatlarni o‘zi topishga harakat qiladi.

Nazoratsiz o‘qitish algoritmlari quyidagi muammolarni hal qilish imkonini beradi.

Strukturalanmagan ma’lumotlarni sinflashtirish;

Tabiiy tilni qayta ishlash;

Tasvirlarni sinflashtirish;

Ma’lumotlarga asoslanib qonuniyatlarni aniqlash kabi masalalarni hal qiladi.

Quyida ba'zi mashhur nazoratsiz o‘qitish algoritmlari ro‘yxati keltirilgan:

- K-klasterlash (K-means clustering);
- K-eng yaqin qo‘shnilar **(k-nearest neighbors);**
- Ierarxal klasterlash (Hierarchal clustering);
- Anomaliyani aniqlash (Anomaly detection);

- Neyron tarmoqlari.

Nazoratsiz o‘qitishning afzalliklari

Nazoratsiz o‘qitish nazoratli o‘qitishga nisbatan murakkabroq vazifalar uchun ishlatiladi, chunki nazoratsiz o‘qitishda bizda shakllantirilgan(yorliqli) kirish ma’lumotlari mavjud emas.

- Nazoratsiz o‘qitish afzalroq, chunki etiketli ma’lumotlarga nisbatan yorliqsiz ma'lumotlarni olish oson.

Nazoratsiz o‘qitishning kamchiliklari

- Nazoratsiz o‘qitish nazoratli o‘qitishga qaraganda ancha qiyin, chunki u tegishli natijalarga ega emas.
- Nazoratsiz o‘qitish algoritmining natijasi kamroq aniq bo‘lishi mumkin, chunki kirish ma'lumotlari etiketlanmagan va algoritmlar aniq chiqishni oldindan bilmaydi.

Nazoratli o‘qitish va nazoratsiz o‘qitish o‘rtasidagi asosiy farqlar quyida keltirilgan:

Nazoratli o‘qitish	Nazoratsiz o‘qitish
Nazoratli o‘qitish algoritmlari etiketli ma’lumotlardan foydalangan holda o‘qitiladi.	Nazoratsiz o‘qitish algoritmlari etiketlanmagan ma'lumotlardan foydalangan holda o‘qitiladi.
Nazoratli o‘qitish modeli to‘g‘ri natijani bashorat qilish yoki yo‘qligini tekshirish uchun to‘g‘ridan-to‘g‘ri fikr-mulohazalarni oladi.	Nazoratsiz o‘qitish hech qanday fikr-mulohazalarni qabul qilmaydi.
Nazoratli o‘qitish modeli natijani bashorat qiladi.	Nazoratsiz o‘qitish ma'lumotlardagi yashirin naqshlarni topadi.
Nazoratli o‘qitishda modelga kirish ma'lumotlari chiqish bilan birga taqdim etiladi.	Nazoratsiz o‘qitishda modelga faqat kirish ma'lumotlari taqdim etiladi.
Nazoratli o‘qitishning maqsadi yangi ma'lumotlar berilganda natijani bashorat qila oladigan modelni o‘rgatishdir.	Nazoratsiz o‘qitishning maqsadi noma'lum ma'lumotlar to‘plamidan yashirin naqshlarni va foydali tushunchalarni topishdir.

Nazoratli o‘qitish modelni o‘rgatish uchun nazoratga muhtoj.	Nazoratsiz o‘qitish modelini o‘rgatish uchun hech qanday nazoratga muhtoj emas.
Nazoratli o‘qitishni **Tasniflash** va **Regressiya** muammolariga ajratish mumkin.	Nazoratsiz o‘qitishni **Klasterlash** va **Assotsiatsiyalar** muammolarida tasniflash mumkin .
Nazoratli o‘qitish biz kirish va tegishli natijalarni biladigan holatlar uchun ishlatilishi mumkin.	Nazoratsiz o‘qitish bizda faqat kirish ma'lumotlariga ega bo‘lgan va tegishli chiqish ma'lumotlari bo‘lmagan holatlar uchun ishlatilishi mumkin.
Nazoratli o‘qitish haqiqiy sun'iy intellektga yaqin emas, chunki bunda biz avval har bir ma'lumot uchun modelni o‘rgatamiz va shundan keyingina u to‘g‘ri natijani bashorat qila oladi.	Nazoratsiz o‘qitish haqiqiy sun'iy intellektga yaqinroqdir, chunki u xuddi bola kundalik narsalarni o‘z tajribasidan o‘rgangandek o‘rganadi.

Yuqorida keltirilgan mashinaviy o‘qitish algoritmlari asosida sun’iy intellektning ba’zi muammolarini hal qilish imkoniyati yaratiladi.

Nazariy savollar

1. Mashinaviy o‘qitish va uning vazifasini tushuntiring?
2. Nazoratli o‘qitish va uning vazifasini tushuntiring?
3. Regressiya va uning vazifasini tushuntiring?
4. Klassifikatsiyalash (Tasniflash) va uning vazifasini tushuntiring?
5. Nazoratsiz mashinaviy o‘qitish va uning vazifasini tushuntiring?
6. Nazoratsiz o‘qitishning ishlash funksiyasini tushuntiring?
7. Nazoratsiz o‘qitish algoritmi turlari va ularning vazifasini tushuntiring?
8. Nazoratsiz o‘qitishning afzalliklarini tushuntiring?

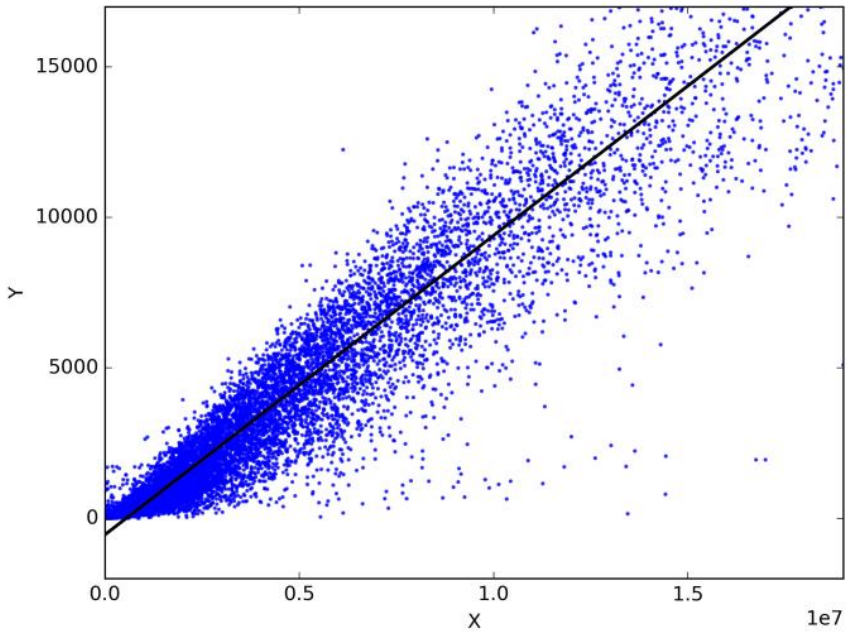
2.2. Nazorat ostida o‘qitish. Bir o‘zgaruvchili chiziqli regressiya

Reja:

Mashinaviy o‘qitishda regressiya tahlili;
Regressiya turlari;
Chiziqli regressiya;
Model aniqligini baholash;
Mashinaviy o‘qitishda oddiy chiziqli regressiya;
Oddiy chiziqli regressiya modeli;
Test va o‘qitish to‘plami;
Korrelyatsiya.

Mashinaviy o‘qitishda regressiya tahlili

Regression tahlil - bir yoki bir nechta mustaqil o‘zgaruvchilarga ega bo‘lgan bog‘liq (maqsadli) va mustaqil (bashoratchi) o‘zgaruvchilar o‘rtasidagi munosabatlarni modellashtirishning statistik usuli hisoblanadi. Aniqroq aytganda, regressiya tahlili boshqa mustaqil o‘zgaruvchilar o‘zgarmas bo‘lganda, bog‘liq o‘zgaruvchining qiymati mustaqil o‘zgaruvchiga mos ravishda qanday o‘zgarishini tushunishga yordam beradi. Masalan **harorat, yosh, ish haqi, narx va** boshqalar kabi doimiy/real qiymatlarni bashorat qilishi mumkin.



Quyidagi misol yordamida regressiya tahlili tushunchasini tushunish mumkin.

Misol: Faraz qilaylik, A reklama kompaniyasi bor, u har yili turli reklamalar qilish bilan shug‘ullanadi va undan foyda oladi. Quyidagi ro‘yxatda kompaniyaning so‘nggi 5 yil ichida qilgan reklamasi va tegishli foydalari ko‘rsatilgan:

MAHSULOT REKLAMASI	MAHSULOT SAVDOSI
900 000	**90 000 000**
150 000	**110 000 000**
180 000	**130 000 000**
200 000	**150 000 000**
210 000	**160 000 000**
250 000	**?**

Endi kompaniya 2023-yilda 250 minglik reklama **qilishni istaydi va bu yilgi sotuvlar haqidagi bashorat natijalarini bilmoqchi**. Shunday qilib, mashinaviy o‘qitish bunday turdagi bashorat qilish muammolarini hal qilish uchun regressiya tahlili kerak bo‘ladi.

Regressiya - bu o‘zgaruvchilar o‘rtasidagi korrelyatsiyani topishga yordam beradigan va bir yoki bir nechta bashorat qiluvchi o‘zgaruvchilar asosida uzluksiz chiqish o‘zgaruvchisini bashorat qilish imkonini beradigan **nazoratli o‘qitish usuli hisoblanadi.**U asosan **bashorat qilish, vaqt seriyalarini modellashtirish va o‘zgaruvchilar o‘rtasidagi sabab-ta'sir** munosabatlarini aniqlash uchun ishlatiladi**.**

Regressiyada biz berilgan ma’lumotlar nuqtalariga eng mos keladigan o‘zgaruvchilar o‘rtasida grafik chiziladi, ushbu sxemadan foydalanib, mashinaviy o‘qitish modeli ma’lumotlar haqida bashorat qilishi mumkin. Oddiy so‘z bilan aytganda, regressiya maqsadli bashorat qiluvchi grafikdagi barcha ma'lumotlar nuqtalaridan o‘tadigan chiziq yoki egri chiziqni shakllantiradi, shunda ma'lumotlar nuqtalari va regressiya chizig‘i orasidagi vertikal masofa minimal bo‘ladi.

Ma’lumotlar nuqtalari va chiziq o‘rtasidagi masofa model kuchli munosabatlarga ega yoki yo‘qligini ko‘rsatadi.

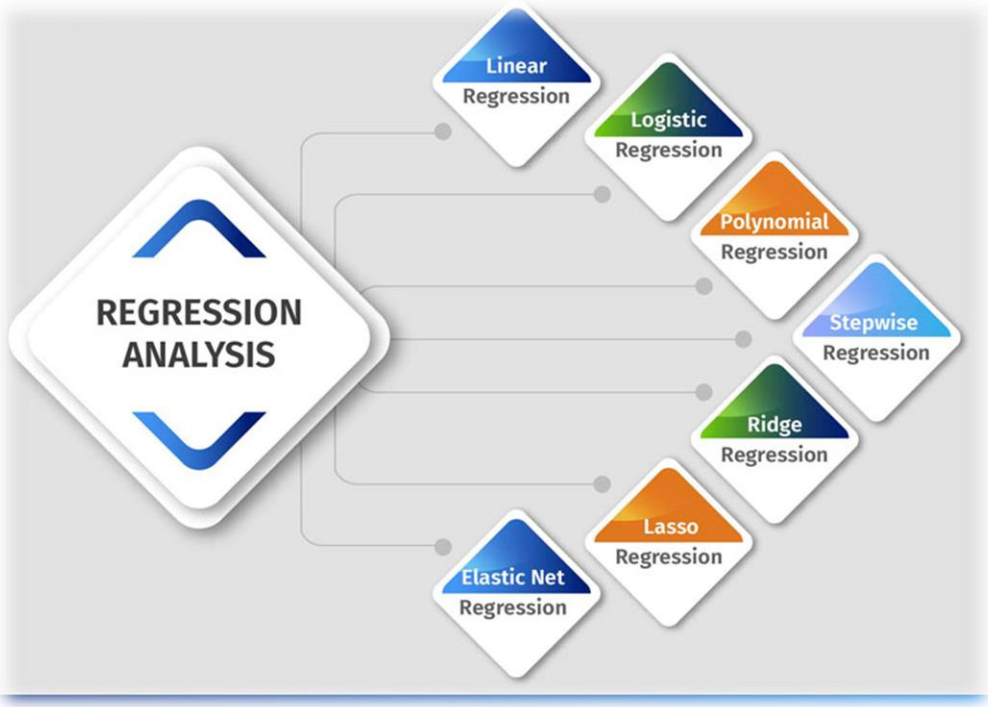
Regressiya tahlili bilan bog‘liq atamalar:

- **Bog‘liq o‘zgaruvchi:** bashorat qilmoqchi bo‘lgan yoki tushunmoqchi bo‘lgan regressiya tahlilidagi asosiy omilga bog‘liq o‘zgaruvchiga aytiladi. **U maqsadli o‘zgaruvchi** deb ham ataladi .
- **Mustaqil o‘zgaruvchi:** bog‘liq o‘zgaruvchilarga ta’sir qiluvchi yoki bog‘liq o‘zgaruvchilar qiymatlarini bashorat qilish uchun foydalaniladigan omillar mustaqil o‘zgaruvchi deb ataladi, shuningdek, **bashorat qiluvchi** deb ham ataladi .

- **Chiqib ketishlar:** chiqib ketish - bu boshqa kuzatilgan qiymatlarga nisbatan juda past yoki juda yuqori qiymatni o‘z ichiga olgan kuzatish hisoblanadi. Chiqib ketish natijaga xalaqit berishi mumkin, shuning uchun undan voz kechish kerak bo‘ladi.
- **Multikollinearlik:** Agar mustaqil o‘zgaruvchilar boshqa o‘zgaruvchilarga qaraganda bir-biri bilan yuqori korrelyatsiya bo‘lsa, bunday holat **multikollinearlik** deb ataladi. U ma’lumotlar to‘plamida bo‘lmasligi kerak, chunki u eng ko‘p ta’sir qiluvchi o‘zgaruvchini tartiblashda muammo yaratadi.
- **Noto‘g‘ri moslashtirish va haddan tashqari moslashtirish:** Agar bizning algoritmimiz o‘quv ma’lumotlar to‘plami bilan yaxshi ishlasa, lekin test ma’lumotlar to‘plami bilan yaxshi ishlamasa, bunday muammo **Overfitting** ya’ni moslashib qolish deb ataladi . Agar bizning algoritmimiz hatto o‘quv ma’lumotlar to‘plamida ham yaxshi ishlamasa, bunday muammo **underfitting** deyiladi.

Regressiya turlari

Ma’lumotlar ilmi fanida va mashinaviy o‘qitishda qo‘llaniladigan turli xil regressiya turlari mavjud. Har bir turning turli senariylarda o‘ziga xos ahamiyati bor, lekin asosiyda barcha regressiya usullari mustaqil o‘zgaruvchining bog‘liq o‘zgaruvchilarga ta’sirini tahlil qiladi.



Quyida biz keltirilgan bir necha muhim regressiya turlarini qarab o‘tamiz:

- Chiziqli regressiya
- Logistik regressiya

- Polinomli regressiya
- Vektor mashinasini qo‘llab-quvvatlash
- Qaror daraxtining regressiyasi
- Tasodifiy o‘rmon regressiyasi
- Ridge regressiyasi
- Lasso regressiyasi:

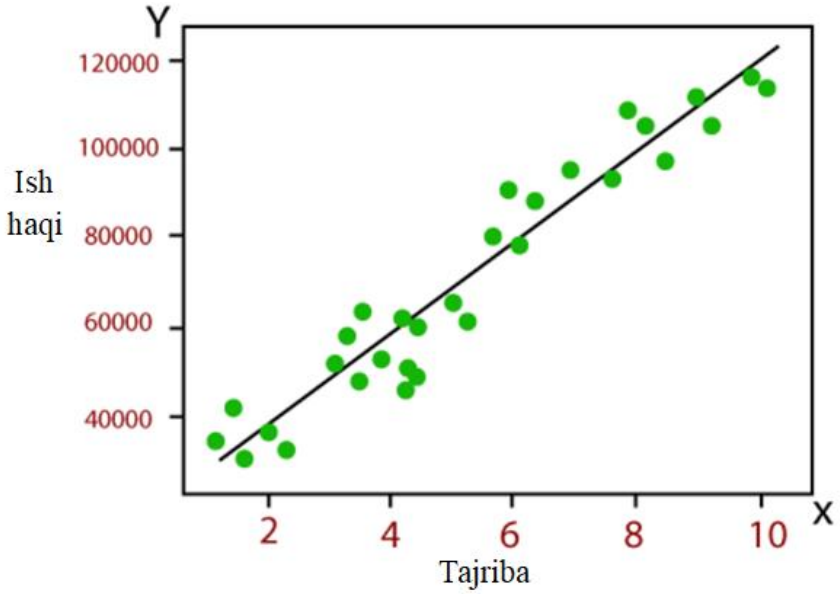
Chiziqli regressiya

Chiziqli regressiya - bu bashoratli tahlil uchun ishlatiladigan statistik regressiya usuli hisoblanadi. Bu regressiya ustida ishlaydigan va uzluksiz o‘zgaruvchilar orasidagi munosabatni ko‘rsatadigan juda oddiy va oson algoritmlardan biridir. U mashinaviy o‘qitishda regressiya muammosini hal qilish uchun ishlatiladi.

Chiziqli regressiya mustaqil o‘zgaruvchi (***x*** o‘qi) va bog‘liq o‘zgaruvchi (***y*** o‘qi) o‘rtasidagi chiziqli munosabatni ko‘rsatadi, shuning uchun **chiziqli regressiya deb ataladi.**

Agar faqat bitta (***x***) kirish o‘zgaruvchisi bo‘lsa, unda bu **oddiy chiziqli regressiya** deyiladi . Agar bir nechta kirish o‘zgaruvchisi bo‘lsa, bunday chiziqli regressiya **ko‘p chiziqli regressiya** deyiladi .

Chiziqli regressiya modelidagi o‘zgaruvchilar o‘rtasidagi munosabatni quyidagi rasm yordamida tushunish mumkin. **Bu erda xodim tajribasi** asosida xodimning **ish haqini** taxmin qilishni ko‘rish mumkin.



Quyida chiziqli regressiya uchun matematik tenglama keltirilgan:

$$y = bx + a \qquad (1)$$

Bu erda *y* bog‘liq o‘zgaruvchilar (maqsadli o‘zgaruvchilar), *x* mustaqil o‘zgaruvchilar (bashoratchi o‘zgaruvchilar), *a* va *b* - chiziqli koeffitsientlar.

Chiziqli regressiya bilan ishlaganda asosiy maqsad ma’lumotlarni ifoda etuvchi eng yaxshi chiziqni topish. Agarda ushbu regressiya chizig‘i 1-formula orqali ifoda etilayotgan bo‘lsa, u holda *a* va *b* koeffitsiyentlarni topishimiz kerak. Bunda ushbu konstantalar orqali prognoz qilingan natija ma’lumotlar to‘plamida keltirilgan haqiqiy qiymatga imkon qadar yaqin bo‘lishi, ya’ni prognoz qilingan qiymat va haqiqiy qiymat imkon qadar yaqin bo‘lishi, ya’ni ular orasidagi tafovut imkon qadar kam bo‘lishi kerak. Ushbu tafovut xatolik deb ataladi.

Model aniqligini baholash

Aniqlikni baholashning turli usullari bor, regressiya algoritmlar uchun odatda **o‘rtacha kvadrat xatolik** (Root Mean Square Error - RMSE) ko‘p ishlatiladi:

$$F_{RMS}(x,h) = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(h(x^{(i)}) - y^{(i)})^2}$$

Bu yerda:

m-ma’lumotlar to‘plamidagi elementlar soni;

$y^{(i)}$ - ma’lumotlar to‘plami;

i-qatorning haqiqiy qiymati;

$h(x^{(i)})$ - i qatordagi prognoz qilingan qiymat.

Aniqlikni baholashning yana bir usuli, **o‘rtacha absolyut xatolik** (mean absolute error - MAE).

$$F_{MAE}(x,h) = \frac{1}{m}\sum_{i=1}^{m}| h(x^{(i)}) - y^{(i)} |$$

Prognoz qilingan qiymat haqiqiy qiymatdan qanchalik uzoqda bo‘lsa, o‘rta kvadratik xatolik funksiyasi ham shuncha katta bo‘ladi.

Mashinaviy o‘qitishda oddiy chiziqli regressiya

Oddiy chiziqli regressiya - bu bog‘liq o‘zgaruvchi va bitta mustaqil o‘zgaruvchi o‘rtasidagi munosabatlarni modellashtiruvchi regressiya algoritmlarining bir turi hisoblanadi. Oddiy chiziqli regressiya modelida ko‘rsatilgan munosabatlar chiziqli yoki qiya to‘g‘ri chiziqdan iborat bo‘ladi, shuning uchun u oddiy chiziqli regressiya deb ataladi.

Oddiy chiziqli regressiyadagi asosiy nuqta shundaki, **bog‘liq o‘zgaruvchi doimiy/real qiymatli bo‘lishi kerak** . Biroq, mustaqil

o‘zgaruvchini uzluksiz yoki kategorik qiymatlar bo‘yicha o‘lchash mumkin bo‘ladi.

Oddiy chiziqli regressiya algoritmi asosan ikkita maqsadda ishlatiladi:

1. **Ikki o‘zgaruvchi o‘rtasidagi munosabatni modellashtirish.** Masalan, daromad va xarajatlar, tajriba va ish haqi o‘rtasidagi bog‘liqlik va shu kabi jarayonlarni amalga oshiradi.
2. **Yangi kuzatuvlarni bashorat qilish.** Masalan, harorat bo‘yicha ob-havo prognozi, bir yildagi investitsiyalar bo‘yicha kompaniyaning daromadi va shu kabi jarayonlarni amalga oshiradi.

Oddiy chiziqli regressiya modeli

Oddiy chiziqli regressiya modeli quyidagi tenglama yordamida ifodalanadi:

$$y = a_0 + a_1 x + \varepsilon$$

a_0 va a_1 konstantalar regressiya chizig‘ining yo‘nalishini belgilab beradi.

ε **xatolik (yaxshi model uchun bu ahamiyatsiz bo‘ladi).**

Python dasturlash tili yordamida oddiy chiziqli regressiya algoritmini amalga oshirish jarayoni quyidagicha amalga oshiriladi. Buning uchun dastlab qisqa hududlardagi uylarning xarakteristikasi bo‘yicha narxlari keltirilgan ma’lumotlar to‘plamini faollashtirib olish kerak bo‘ladi. Quyidagi kod orqali **df** nomli ma’lumotlar to‘plami shakllantirib olinadi.

```
import pandas as pd
data={'Hudud':['Samarqand','Urgut','Qorasuv','Qo
rasuv','Qorasuv','Qorasuv','Qorasuv','Urgut','Is
htixon','Qorasuv'],'Xona':[3,2,2,3,3,1,1,2,2,1],
'Maydon':[57,52,42,65,70,28,30,32,51,30],'Qavat'
:[4,4,4,1,3,1,2,5,3,1],'Uy_Qavat':[4,5,4,4,5,4,4
,5,4,4],'Narx':[52000,56000,37000,49500,55000,25
500,21200,20000,26200,22200]};
df=pd.DataFrame(data)
df
```

	Hudud	Xona	Maydon	Qavat	Uy_Qavat	Narx
0	Samarqand	3	57	4	4	52000
1	Urgut	2	52	4	5	56000
2	Qorasuv	2	42	4	4	37000
3	Qorasuv	3	65	1	4	49500
4	Qorasuv	3	70	3	5	55000
5	Qorasuv	1	28	1	4	25500
6	Qorasuv	1	30	2	4	21200
7	Urgut	2	32	5	5	20000
8	Ishtixon	2	51	3	4	26200
9	Qorasuv	1	30	1	4	22200

Bunda yuqoridagi uy narxlariga oid ma’lumotlar to‘plami bor undan ikkita o‘zgaruvchiga ega bo‘lgan ma’lumotlar to‘plami qaraladi: **Maydon** va **Narx**. Ushbu muammoning maqsadlari quyidagicha bo‘ladi:

- Dastlab ikki o‘zgaruvchi o‘rtasida korrelyatsiya bor-yo‘qligi aniqlanadi;
- Ma’lumotlar to‘plami uchun eng mos chiziq topiladi;
- Mustaqil o‘zgaruvchini o‘zgartirish orqali bog‘liq o‘zgaruvchi qanday o‘zgarishi tahlil qilinadi.

Ushbu ikki o‘zgaruvchi o‘rtasidagi munosabatni ifodalash va eng mos chiziqni topish uchun oddiy chiziqli regressiya modelini yaratiladi.

Python yordamida mashinaviy o‘qitishda oddiy chiziqli regressiya modelini amalga oshirish uchun quyidagi amallar bajariladi:

Ma'lumotlarni oldindan qayta ishlash. Oddiy chiziqli regressiya modelini yaratish uchun birinchi qadam ma’lumotlarni oldindan qayta ishlash hisoblanadi, ya’ni yuqoridagi kabi ma’lumotlar to‘plami shakllantiriladi. Ma’lumotlar shakllantirilgandan so‘ng, birinchidan, uchta muhim kutubxonani import qilish kerak, bu ma’lumotlar to‘plamini yuklash, grafiklarni chizish va oddiy chiziqli regressiya modelini yaratish uchun kerak bo‘ladi.

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

Yuqorida keltirilgan kod yozilgandan so‘ng, ma’lumotlar to‘plami yuklanmagan bo‘lsa yuklash kerak, yuklangan bo‘sa shu ma’lumotlar to‘plami bilan ish olib boriladi.

Bizda yuklangan **df** ma’lumotlar to‘plamidan, Qorasuv tumanidagi uylarning narxini bashorat qilib ko‘ramiz. Ma’lumotlar to‘plamidan Qorasuv tumaniga tegishli ma’lumotar ajratib olinadi.

```
Uy=df[df.Hudud=='Qorasuv']
Uy.head()
```

	Hudud	Xona	Maydon	Qavat	Uy_Qavat	Narx
2	Qorasuv	2	42	4	4	37000
3	Qorasuv	3	65	1	4	49500
4	Qorasuv	3	70	3	5	55000
5	Qorasuv	1	28	1	4	25500
6	Qorasuv	1	30	2	4	21200

X o‘zgaruvchiga maydon ustunidagi qiymatlarni, *Y* o‘zgaruvchiga narx ustunidagi qiymatlarni massiv ko‘rinishida ta’minlanadi.

```
X=Uy['Maydon'].to_numpy()
X
```

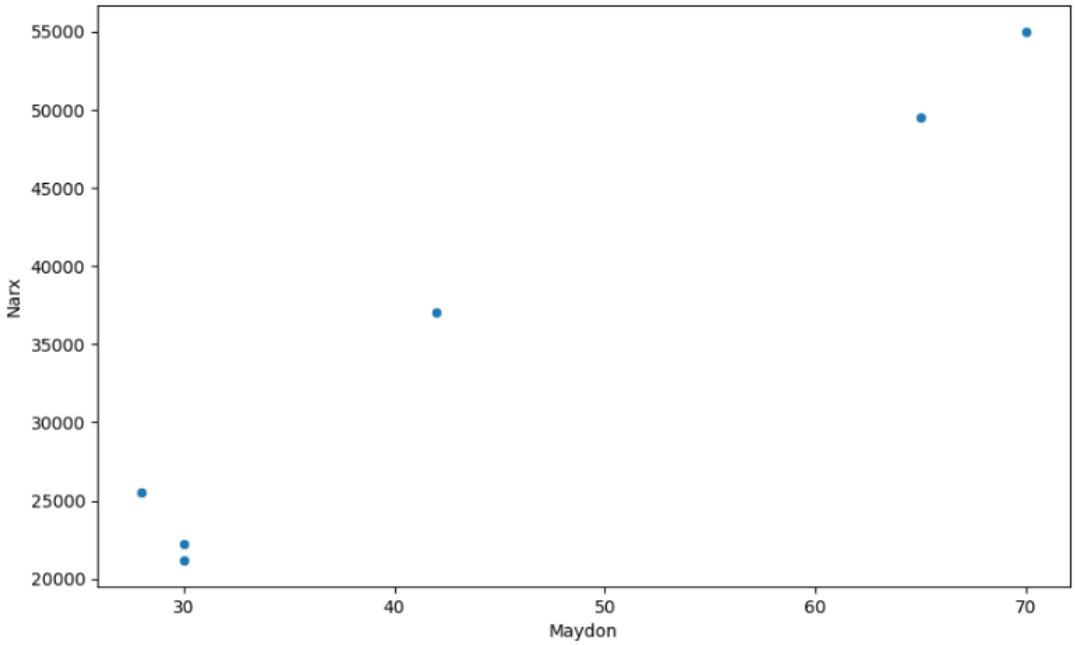
```
array([42, 65, 70, 28, 30, 30])
```

```
Y=Uy['Narx'].to_numpy()
Y
```

```
array([37000, 49500, 55000, 25500, 21200, 22200])
```

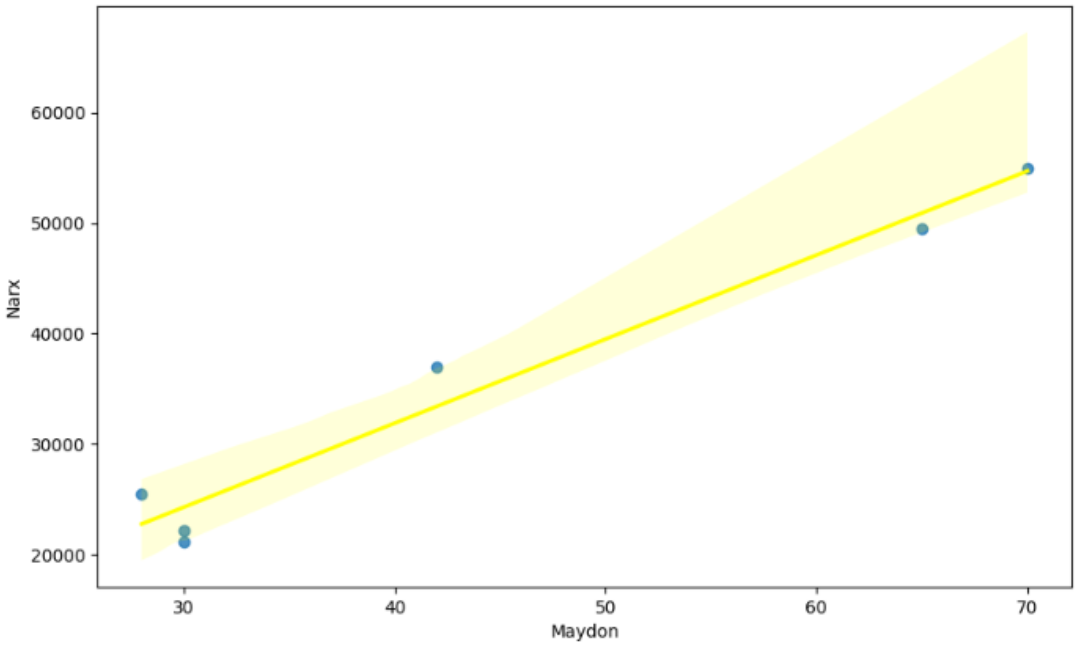
Narx va uy maydoni o‘rtasida chiziqli bog‘liqlik bor ekanini tekshirib quyidagi kod orqali aniqlanadi:

```
plt.figure(figsize=(10,6))
sns.scatterplot(data=Uy, x='Maydon', y='Narx')
plt.show()
```



Quyidagi kod qatorini bajarib, belgilangan ma’lumotlar to‘plamida bashorat chizig‘i mavjud ekanini aniqlash mumkin.

```
plt.figure(figsize=(10,6))
sns.regplot(data=Uy, x='Maydon', y='Narx',
        line_kws={"color":"yellow"})
plt.show()
```



Yuqoridagi grafikda ko‘k nuqtalar haqiqiy qiymatlarni ifodalaydi va bashorat qilingan qiymatlar sariq regressiya chizig‘i bilan qoplangan. Regressiya chizig‘i bog‘liq va mustaqil o‘zgaruvchilar o‘rtasidagi korrelyatsiyani ko‘rsatadi. Chiziqning yaxshi mosligini haqiqiy qiymatlar va bashorat qilingan qiymatlar o‘rtasidagi farqni

hisoblash orqali kuzatish mumkin. *x* va *y* o‘zgaruvchining qiymatlari va quyidagi formuladan foydalangan holda *a* va *b* koeffisientlarni hisoblash mumkin.

$$b = \frac{\sum_{i=1}^{n}(x_i - \tilde{x})(y_i - \tilde{y})}{\sum_{i=1}^{s}(x_i - \tilde{x})^2}$$

$$a = \tilde{y} - b\tilde{x}$$

Bu yerda $\tilde{x}$ - *x* ustun uchun o‘rtracha qiymat, $\tilde{y}$- *y* ustun uchun o‘rtacha qiymat.

Dastlab, *a* va *b* koeffisientlarni hisoblab olinadi.

```
import numpy as np
Xmean=np.mean(X)
Ymean=np.mean(Y)
Xmean,Ymean
```

(44.166666666666664, 35066.666666666664)

```
b=sum((X-Xmean)*(Y-Ymean))/sum((X-Xmean)**2)
print(f"{b=}")
```

```
b=760.2939790822576
```

```
a=Ymean-b*Xmean
print(f"{a=}")
```

```
a=1487.0159238669585
```

Test to‘plamini tashkil qilish uchun “Maydon”ga tegishli ma’lumotlardan ixtiyoriy 3 ta qiymat va mos ravishda narxlarini ham ajratib olinadi, ma’lumotlar to‘plami katta bo‘lish kerak va test to‘plami ham katta bo‘lsa natija yaxshi bo‘ladi.

```
x_test=Uy.sample(3,
random_state=6)['Maydon'].to_numpy()
print(f"{x_test=}")
y_test=Uy.sample(3,
random_state=6)['Narx'].to_numpy()
print(f"{y_test=}")
```

```
x_test=array([30, 30, 42])
y_test=array([21200, 22200, 37000])
```

a va *b* koeffisientlar yordamida ajratilgan 3 ta qiyma asosida narxlar bashorat qilinadi.

```
y_bashorat=a+b*x_test
print(f"{y_bashorat=}")
y_bashorat=array([24295.83529633,
24295.83529633, 33419.36304532])
```

Natijalardan ko‘rinib turibdiki haqiqiy narxlardan bashorat qilingan narxlar farq qilib turibdi. Natijalarning aniqligini bilish uchun ular o‘rtasidagi o‘rtacha kvadrat xatolik va o‘rtacha absolyut xatoliklari hisoblanadi.

```
MAE=np.sum(np.absolute(y_bashorat-
y_test))/len(y_test)
print(f"{MAE=}")
```

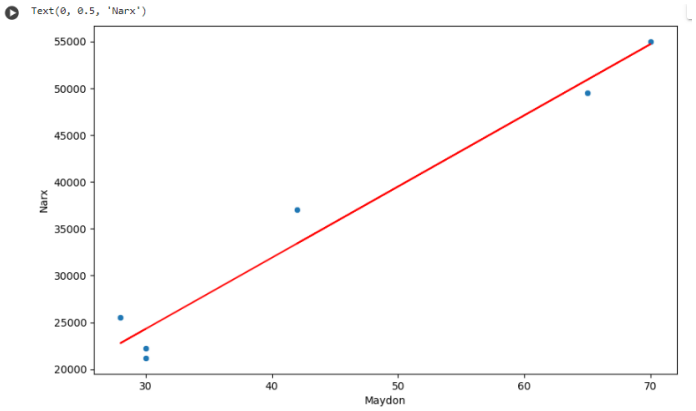
Natija: `MAE=2924.10251578253`

```
RMSE=np.sqrt(np.sum((y_bashorat-
y_test)**2)/len(y_test))
print(f"{RMSE=}")
```

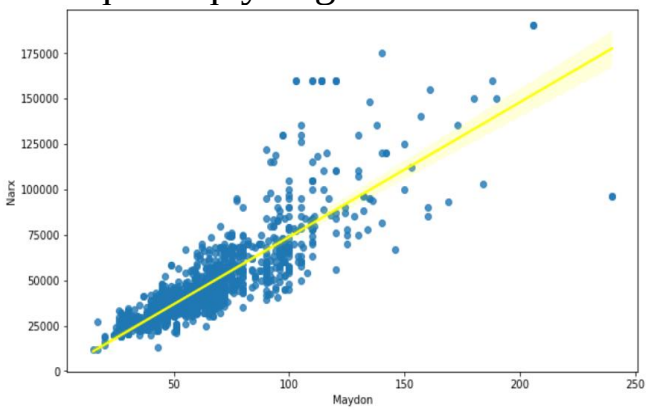
Natija: `RMSE=2988.7390190848764`

Topilgan *a* va *b* koeffisientlar yordamida $y = ax + b$ tenglama uchun regressiya chizig‘i chiziladi.

```
plt.figure(figsize=(10,6))
sns.scatterplot(data=Uy, x='Maydon', y='Narx')
plt.plot(X,b*X+a,'-r')
plt.xlabel("Maydon")
plt.ylabel("Narx")
```



Demak yuqoridagi amallar asosida chiziqli regressiyani amalga oshirish mumkin hamda “Maydon” va “Narx” maydonlaridan foydalanib bashorat natijalarini olish imkoniyati mavjud. Yuqorida biz qaragan ma’lumotlar to‘plamida ma’lumotlar kam bo‘lganligi sababli nuqtalar ham kam tasvirlangan, ma’lumotlar yetarli darajada ko‘p bo‘lsa natija ham yaxshi bo‘ladi va nuqtalar quyidagi tasvir ko‘rinishida bo‘ladi.



Test va o‘qitish to‘plami

Mashinaviy o‘qitish uchun ma’lumotlarni ikki qismga ajratib olish kerak bo‘ladi.

Train set – o‘qitish to‘plami, bu to‘plam model yaratish uchun kerak bo‘ladi.

Test set – test to‘plam, bu to‘plam model aniqligini tekshirish uchun kerak bo‘ladi.

Ma’lumotlarni test va o‘qitish to‘plamiga ajratish yuqorida oddiy chiziqli regressiya modelini qurishda ishlatilgan. Yuqorida oddiy chiziqli regressiya modelini qurishda to‘plamda ma’lumot kam bo‘lganligi sababli 6 tadan 3 tasi test to‘plam sifatida ajratib olingan. Ma’lumotlarning 80% train o‘qitish, 20% test to‘plam uchun ajratish maqsadga muvofiq bo‘ladi.Buning uchun **scikit-learn** tarkibidagi tayyor **train_test_split** funskyasiga murojaat qilish mumkin. Funskiyaga parametr sifatida dataset (df), test set hajmi (0.2 ya'ni 20%) va tasodifiy sonlar generatori uchun qiymat (**random_seed**) beriladi. **Random_seed**ning vazifasi **train_test_split** funksiyani ishga tushirganda doim bir xil tasodifiy qiymatlar olishni amalga oshiradi. Bu esa, mashinaviy o‘qitishning model yaratish jarayonida **test_set** doim yashirin qolishini ta’minlaydi. Ma’lumotlarni o‘qitish va test to‘plamlariga ajratish uchun quyidagi kodni keltirish mumkin.

```
from sklearn.model_selection import
                train_test_split
train_set,test_set=train_test_split(Uy,
    test_size=0.2, random_state=6)
```

Ajratilgan o‘qitish to‘plamini quyidagi kod orqali ko‘rish mumkin.

```
train_set.head()
```

	Hudud	Xona	Maydon	Qavat	Uy_Qavat	Narx
2	Qorasuv	2	42	4	4	37000
5	Qorasuv	1	28	1	4	25500
3	Qorasuv	3	65	1	4	49500
4	Qorasuv	3	70	3	5	55000

Ajratilgan test to‘plamini quyidagi kod orqali ko‘rish mumkin.

	Hudud	Xona	Maydon	Qavat	Uy_Qavat	Narx
6	Qorasuv	1	30	2	4	21200
9	Qorasuv	1	30	1	4	22200

Ajratilgan o‘qitish va test to‘plamlari asosida mashinaviy o‘qitishning modelini yaratish va modelni aniqligini baholash imkoni yaratiladi.

Korrelyatsiya

Qaralayotgan ma’lumotlar to‘plamidagi ma'lumotlar orasida uyning narxiga ta’sir qiluvchi parametrlarni topish kerak bo‘ladi. Buni amalga oshirish uchun maydonlarning bir – biriga bog‘liqligini tekshirishda korrelyatsiyani aniqlash kerak bo‘ladi.

```
Uy.corr()
```

	Xona	Maydon	Qavat	Uy_Qavat	Narx
Xona	1.000000	0.985983	0.321634	0.581318	0.987856
Maydon	0.985983	1.000000	0.260599	0.672864	0.982715
Qavat	0.321634	0.260599	1.000000	0.387298	0.345529
Uy_Qavat	0.581318	0.672864	0.387298	1.000000	0.671078
Narx	0.987856	0.982715	0.345529	0.671078	1.000000

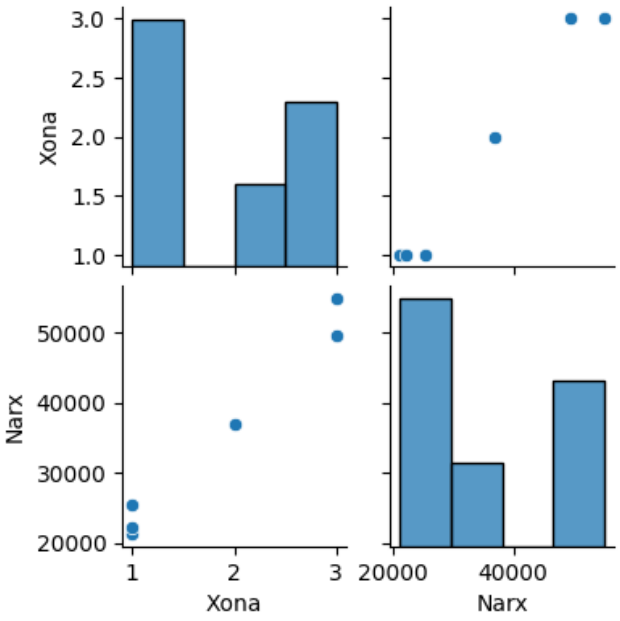
Bitta ustundagi qiymatlarni, qolgan ustundagi qiymatlar bilan bog‘liqligini quyidagi kod orqali solishtirish mumkin.

```
Uy.corrwith(Uy['Xona'])
```

```
Xona        1.000000
Maydon      0.985983
Qavat       0.321634
Uy_Qavat    0.581318
Narx        0.987856
dtype: float64
```

Seabron tarkibidagi pairplot funksiyasi yordamida korrelyasiya qiymatlarini grafik ko‘rinishida chiqarishimiz ham mumkin.

```
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
maydon=['Xona', 'Narx']
sns.pairplot(Uy[maydon], height=2)
plt.show()
```



Natijada har bir maydon qolgan maydonlar bilan qanday ta’sir qilayotganligini aniqlash mumkin. Bitta ustunni qolgan ustundagi qiymatlar bilan solishtirish ham mumkin.

Nazariy savollar

1. Mashinaviy o‘qitishda regressiya tahlilini tushuntirib bering?

2. Regressiya va uning turlarini tushuntirib bering?
3. Chiziqli regressiyani tushuntirib bering?
4. Model aniqligini baholashni tushuntirib bering?
5. Xatoliklarni aniqlash va uning usullarini tushuntirib bering?
6. Mashinaviy o‘qitishda oddiy chiziqli regressiyani tushuntirib bering?
7. Oddiy chiziqli regressiya modelini qurish ketma-ketligini tushuntirib bering?
8. Oddiy chiziqli regressiya modeli asosida olingan ma’lumotlar xatoligini aniqlashni tushuntirib bering?
9. Test va o‘qitish to‘plami hamda ularni shakllantirish bosqichlarini tushuntirib bering?
10. Korrelyatsiya va uning vazifasini tushuntirib bering?

2.3. Mashinaviy o‘qitishda ko‘p chiziqli regressiya

Reja:

Ko‘p chiziqli regressiya;

Ma'lumotlarni oldindan qayta ishlash;

Ko‘p chiziqli regressiya modelini o‘quv to‘plamiga moslashtirish;

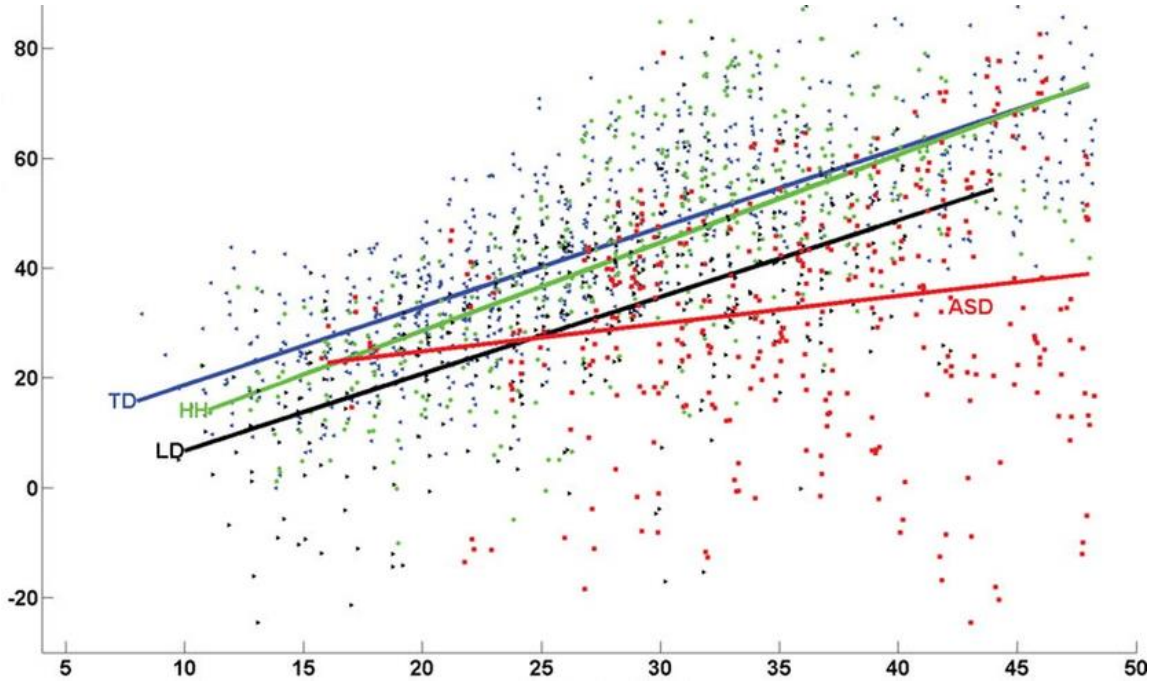
Test to‘plami natijalarini bashorat qilish.

Ko‘p chiziqli regressiya

Oldingi mavzudagi oddiy chiziqli regressiyada javob o‘zgaruvchisini modellashtirish uchun bitta o‘zgaruvchidan foydalanilgan. Lekin shunday masalalar borki, javob o‘zgaruvchisiga bir nechta bashorat qiluvchi o‘zgaruvchilar ta'sir qiladigan turli holatlar bo‘lishi mumkin; bunday holatlar uchun Ko‘p chiziqli regressiya algoritmi qo‘llaniladi.

Bundan tashqari, bir nechta chiziqli regressiya oddiy chiziqli regressiyaning kengaytmasi hisoblanadi, chunki javob o‘zgaruvchisini bashorat qilish uchun bir nechta bashorat qiluvchi o‘zgaruvchilar kerak bo‘ladi. Natijada bu jarayonni quyidagicha ta’riflash mumkin.

Ko‘p chiziqli regressiya – bitta bog‘liq doimiy o‘zgaruvchi va bir nechta mustaqil o‘zgaruvchilar o‘rtasidagi chiziqli munosabatni modellashtiradigan muhim regressiya algoritmlaridan biridir.



Ko‘p chiziqli regressiyada maqsadli o‘zgaruvchi (*Y*) bir nechta bashorat qiluvchi o‘zgaruvchilarning chiziqli kombinatsiyasi $x_1, x_2, x_3, ..., x_n$ lardan tashkil topgan. Bu oddiy chiziqli regressiyaning yaxshilanganligi sababli, ko‘p chiziqli regressiya tenglamasi uchun ham xuddi shunday qo‘llaniladi va tenglama quyidagicha ifodalanadi:

$$Y = a_0 + a_1 x_1 + a_2 x_2 + a_3 x_3 + ... + a_n x_n$$

Bu yerda $x_1, x_2, x_3, ..., x_n$ o‘zaruvchilar, $a_0, a_1, a_2, ..., a_n$ lar esa koeffitsiyentlar hisoblanadi.

Python dasturlash tili yordamida bir nechta chiziqli regressiya modelini amalga oshirish mumkin. Yuqoridagi mavzuda uy narxlariga oid ma’lumotlar to‘plami bor, undan chiziqli regressiya modeli uchun faqat ikkita **Maydon** va **Narx** o‘zgaruvchiga ega bo‘lgan ma'lumotlar to‘plami olingan. Ko‘p chiziqli regressiya uchun qolgan ma’lumotlar to‘plamidan ham foydalaniladi. Masalan **Maydon, Xona, Qavat, Uy_qavati, Narx** kabi ma’lumotlar to‘plamidan foydalaniladi.

Narxni topishimiz kerak bo‘lgani uchun maydon bog‘liq o‘zgaruvchidir, qolgan to‘rtta o‘zgaruvchi esa mustaqil o‘zgaruvchilardir. Quyida ko‘p chiziqli regressiya modelini qo‘llashning asosiy bosqichlari keltirilgan:

- Ma'lumotlarni oldindan qayta ishlash;
- Ko‘p chiziqli regressiya modelini o‘quv to‘plamiga moslashtirish;
- Test to‘plamining natijasini bashorat qilish.

Ma'lumotlarni oldindan qayta ishlash

Birinchi qadam - bu ma'lumotlarni oldindan qayta ishlash, ya’ni ma’lumotlar to‘plamini shakllantirish, bu jarayon oldingi mavzularda keltirib o‘tilgan. Ushbu jarayonda kutubxonalar import qilinadi va tayyorlangan yoki tanlangan barcha o‘zgaruvchilarni o‘z ichiga olgan ma’lumotlar to‘plami import qilinadi. Quyida uning kodi:

```
import pandas as pd
data={'Hudud':['Samarqand','Urgut','Qorasuv','Qo
rasuv','Qorasuv','Qorasuv','Qorasuv','Urgut','Is
htixon','Qorasuv'],'Xona':[3,2,2,3,3,1,1,2,2,1],
'Maydon':[57,52,42,65,70,28,30,32,51,30],'Qavat'
:[4,4,4,1,3,1,2,5,3,1],'Uy_Qavat':[4,5,4,4,5,4,4
```

```
,5,4,4],'Narx':[52000,56000,37000,49500,55000,25
500,21200,20000,26200,22200]};
df=pd.DataFrame(data)
```

Qorasuv tumanidagi uylarning narxini bashorat qilish uchun ma’lumotlar to‘plamidan Qorasuv tumaniga tegishli ma’lumotar ajratib olinadi.

```
Uy=df[df.Hudud=='Qorasuv']
Uy.head()
```

Keyingi qadamda quyidagi kod orqali ma’lumotlar to‘plami o‘quv va test to‘plamlariga ajratiladi. Bunda 20% test uchun, 80% o‘qitish uchun ajratilishi tavsiya etiladi.

```
from sklearn.model_selection import
train_test_split
train_set,test_set=train_test_split(Uy,
test_size=0.2, random_state=6)
```

O‘qitishni amalga oshirish uchun ma’lumotlar to‘plami yuqoridagi buyruqlar orqali tayyorlandi.

Ko‘p chiziqli regressiya modelini o‘quv to‘plamiga moslashtirish

Regressiya modeli o‘quv to‘plamiga moslashtiriladi va ko‘p chiziqli regressiya formulasidagi $a_1, a_2, a_3, \dots, a_n$ koeffitsiyentlar topiladi. $a_1, a_2, a_3, \dots, a_n$ koeffitsiyentlarni hisoblash jarayoni mavjud to‘plam asosida tenglamalar sitemasi orqali hisoblanadi. Biz bu jarayonni dasturlash asosida kod orqali amalga oshirishni bajaramiz.

```
from sklearn import linear_model
MLR_model=linear_model.LinearRegression()
x_train=np.asanyarray(train_set[['Xona','Maydon'
,'Qavat','Uy_Qavat']])
y_train=np.asanyarray(train_set[['Narx']])
MLR_model.fit (x_train,y_train)
print('Koeffitsentlar=',MLR_model.coef_)
print('a0=',MLR_model.intercept_)
```

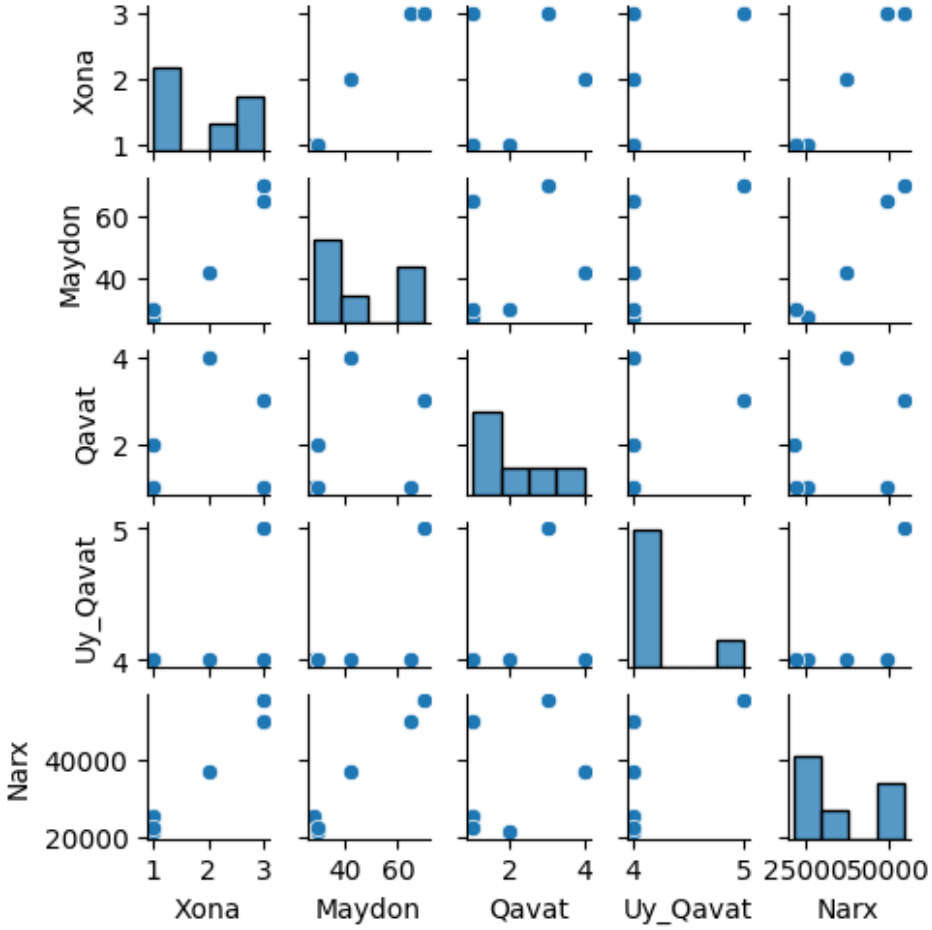
Natija:

```
Koeffitsentlar= [[-148.86039886  656.6951567
818.37606838  579.77207977]]
a0= [4123.93162393]
```

Natijada yuqoridagi kod orqali ko‘p chiziqli regressiyani koeffitsiyentlari topildi.

Seabron tarkibidagi **pairplot** funksiyasi yordamida korrelyasiya qiymatlarini grafik ko‘rinishida chiqarishi ham mumkin. “Maydon, Xona, Qavat, Uy_qavat, Narx” ustunlari o‘rtasidagi bog‘liqlikni quyidagi kod orqali tasvirlash ham mumkin:

```
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
maydon=['Xona', 'Maydon','Qavat',
'Uy_Qavat','Narx']
sns.pairplot(Uy[maydon], height=1)
plt.show()
```



Test to‘plami natijalarini bashorat qilish

Qurilgan ko‘p chiziqli regressiya modelining korrektligini aniqlash uchun bu modelning ishlashini tekshirish kerak. Buning uchun test to‘plam asosida bashorat qiymatlari aniqlanadi. Bashorat qilish uchun **predict** funksiyasidan foydalaniladi**.**

```
x_test=np.asanyarray(test_set[['Xona','Maydon','
Qavat','Uy_Qavat']])
y_test=np.asanyarray(test_set[['Narx']])
y_bashorat=MLR_model.predict(x_test)
print(y_bashorat)
```

Natija:

```
[[27631.76638177]
 [26813.39031339]]
```

Chiqarilgan natija x_test to‘plamga mos y_bashorat natijasi hisoblanadi, aslida x_test to‘plamga mos y_test narxlar to‘plami bor edi, y_test va y_bashorat o‘rtasidagi farqqa qarab modelni korrektligini baholash mumkin.

```
print('y_bashorat=', y_bashorat)
print('y_test=', y_test)
```

Natija:

```
y_bashorat= [[27631.76638177]
 [26813.39031339]]
y_test= [[21200]
     [22200]]
```

Haqiqiy narxlardan bashorat qilingan narxlar farq qilib turibdi, shu sababli modelning bashorat qilingan qiymatlar va test to‘plam qiymatlari orqali ular o‘rtasidagi o‘rtacha kvadrat xatolik va o‘rtacha absolyut xatolikni hisoblab bahosi olinadi.

```
from sklearn.metrics import mean_absolute_error,
mean_squared_error
MAE=mean_absolute_error(y_test, y_bashorat)
RMSE=np.sqrt(mean_squared_error(y_test,
y_bashorat))
print(f"{MAE=}")
print(f"{RMSE=}")
```

Natija:

```
MAE=5522.5783475783555
RMSE=5596.918302660132
```

O‘rtacha kvadrat xatolik taqriban 5597 ga, o‘rtacha absolyut xatolik esa 5523 ga teng ekan. Shuningdek, quyidagi kod orqali o‘quv ma’lumotlar to‘plami va test ma’lumotlar to‘plami uchun ball ko‘rsatkichni tekshirish ham mumkin.

```
print('Uquv tuplam
uchun=',MLR_model.score(x_train, y_train))
print('Test tuplam
uchun=',MLR_model.score(x_test, y_test))
```

Yuqoridagi ball shuni ko‘rsatadiki, qurilgan modelning o‘quv ma’lumotlar to‘plami bilan va test ma'lumotlar to‘plami bilan nechchi % aniqlilikda ishlayotganligi ko‘rinadi.

Nazariy savollar

1. Ko‘p chiziqli regressiya va uning mazmunini tushuntiring?

2. Ma'lumotlarni oldindan qayta ishlash bosqichi, ya’ni ma’lumotlar to‘plamini shakllantirish bosqichini tushuntiring?

3. Ma’lumotlarni test va o‘quv to‘plamalariga ajratish jarayonini tushuntiring?

4. Ko‘p chiziqli regressiya modelini qurish va o‘quv to‘plamiga moslashtirishni tushuntiring?

5. Ko‘p chiziqli regressiya modelining koeffitsentlarini topish jarayonini tushuntiring?

6. Ko‘p chiziqli regressiya modeli asosida test to‘plami natijalarini bashorat qilish jarayonini tushuntiring?

7. Ko‘p chiziqli regressiya modeli asosida bashorat qilingan qiymatlarning xatoliklarini aniqlash jarayonini tushuntiring?

8. Ko‘p chiziqli regressiya modelining o‘quv va test to‘plamlariga nisbatan aniqlilik ko‘rsatkichini baholash jarayonini tushuntiring?

2.4. Mashinaviy o‘qitishda polinomial regressiya

Reja:

Polinomial regressiya;
Chiziqli regressiya modelini qurish;
Polinomial regressiya modelini qurish.

Polinomial regressiya

Mashinaviy o‘qitish algoritmlari tarkibida polinomial regressiyasi algoritmi ham mavjud, bu algoritm ham, jarayonni keyingi bosqichini aniqlash uchun xizmat qiladi. Polinomial regressiya – bog‘liq va mustaqil o‘zgaruvchi o‘rtasidagi munosabatni *n*-darajali ko‘phad sifatida modellashtiruvchi regressiya algoritmi hisoblanadi. Polinom regressiya tenglamasi quyida ifoda orqali shakllantiriladi.

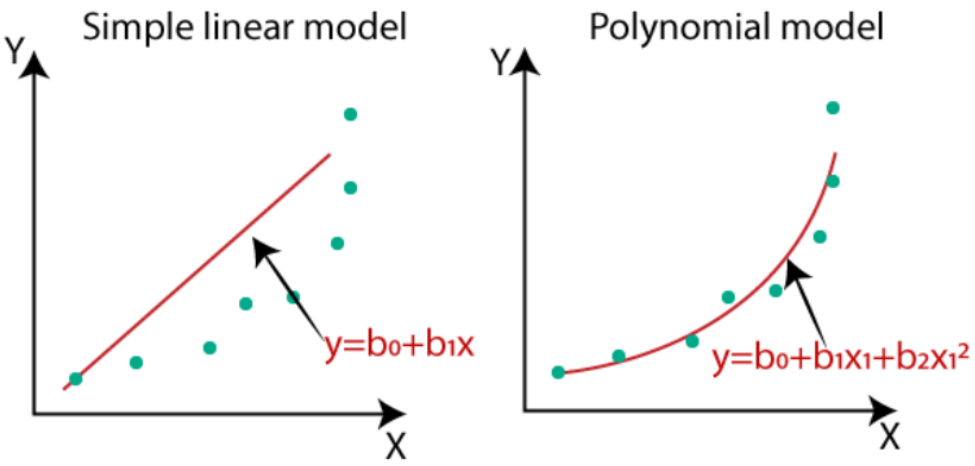
$$Y = a_0 + a_1 x_1 + a_2 x_1^2 + a_3 x_1^3 + ... + a_n x_1^n \quad (1)$$

O‘qitish uchun polinom regressiyasida ishlatiladigan ma’lumotlar to‘plami chiziqli bo‘lmagan xususiyatga ega bo‘ladi. U murakkab va chiziqli bo‘lmagan funksiyalar va ma’lumotlar to‘plamlariga mos keladigan chiziqli regressiya modelidan foydalanadi.

Polinomli regressiyaga ehtiyoj. Mashinaviy o‘qitishda polinomial regressiyaga bo‘lgan ehtiyojni quyidagi fikrlar orqali tushunish mumkin.

Agar chiziqli model **chiziqli ma’lumotlar to‘plamiga** qo‘llansa, u oddiy chiziqli regressiyadagidek yaxshi natija beradi, lekin agar **chiziqli bo‘lmagan ma’lumotlar to‘plamida** bir xil modelni hech qanday o‘zgartirishlarsiz qo‘llanilsa xatolik darajasi yuqori bo‘ladi va aniqlik pasayib ketadi.

Shunday qilib, ma’lumotlar chiziqli bo‘lmagan tarzda joylashtirilgan bo‘lsa bunday holatlar uchun **polinomial regressiya** modeli kerak bo‘ladi. Chiziqli ma’lumotlar to‘plami va chiziqli bo‘lmagan ma'lumotlar to‘plamining quyidagi taqqoslash grafigi orqali yaxshiroq tushunish mumkin.



Yuqoridagi rasmda chiziqli bo‘lmagan tarzda joylashtirilgan ma’lumotlar to‘plami ifodalangan. Shunday qilib, agar bu nuqtalar chiziqli model bilan qoplansa, u hech qanday ma’lumot nuqtasini deyarli qamrab olmasligi aniq ko‘rinib turibti. Egri chiziq Polinomial modeldagi ma’lumotlar nuqtalarining ko‘pini qoplash uchun mos keladi.

Demak, agar ma’lumotlar to‘plami chiziqli bo‘lmagan tarzda joylashtirilgan bo‘lsa, oddiy chiziqli regressiya o‘rniga Polinomli regressiya modelidan foydalanish taklif etiladi.

Python tilida polinomial regressiyasini amalga oshirish

Bu yerda Python yordamida polinomial regressiyasini amalga oshirishda Polinomial regressiya modelini oddiy chiziqli regressiya modeli bilan solishtirish orqali tushuntiriladi.

Polinomli regressiyani amalga oshirishning asosiy bosqichlari quyidagicha bo‘ladi:

- Ma'lumotlarni oldindan qayta ishlash;
- Chiziqli regressiya modelini yaratish va uni ma'lumotlar to‘plamiga moslash;
- Polinomial regressiya modelini yaratish va uni ma'lumotlar to‘plamiga moslash;
- Chiziqli regressiya va polinomial regressiya modeli uchun natijani vizualizatsiya qilish;
- Natijani bashorat qilish;

Ma’lumotlarni oldindan qayta ishlash bosqichi. Ma'lumotlarni oldindan qayta ishlash bosqichi oldingi regressiya modellaridagi kabi qoladi, ba’zi o‘zgarishlar bundan mustasno. Polinomli regressiya modelida ma'lumotlar to‘plamini o‘quv va test to‘plamiga ajratmaymiz. Buning ikkita sababi bor:

- Ma’lumotlar to‘plamida juda kam ma’lumot mavjud bo‘lib, u sinov va o‘quv to‘plamiga bo‘lish uchun mos kelmaydi, aks holda bizning modelimiz ma’lumotlar o‘rtasidagi bog‘liqlikni topa olmaydi.
- Quyidagi modelda ish haqi bo‘yicha juda aniq bashorat qiymat olish uchun, model yetarli ma’lumotga ega bo‘lishi kerak.

Dastlab quyidagi kod orqali ma’lumotlar to‘plami yuklab olinadi.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
df=pd.read_csv("/content/Ice_cream selling
data.csv")
df.head(5)
```

	Temperature (°C)	Ice Cream Sales (units)
0	-4.662263	41.842986
1	-4.316559	34.661120
2	-4.213985	39.383001
3	-3.949661	37.539845
4	-3.578554	32.284531

Ma’lumotlar to‘plamida, ikkita ustunni o‘z ichiga olgan “Ice_cream selling” ma’lumotlari shakllantirilgan. Keyingi qadamda ma’lumotlar ajratib olinadi va normalizasiya qilinadi.

```
import numpy as np
x=np.asanyarray(df[['Temperature
(°C)']]/df['Temperature (°C)'].max())
y=np.asanyarray(df[['Ice Cream Sales
(units)']]/df['Ice Cream Sales (units)'].max())
```

Chiziqli regressiya modelini qurish

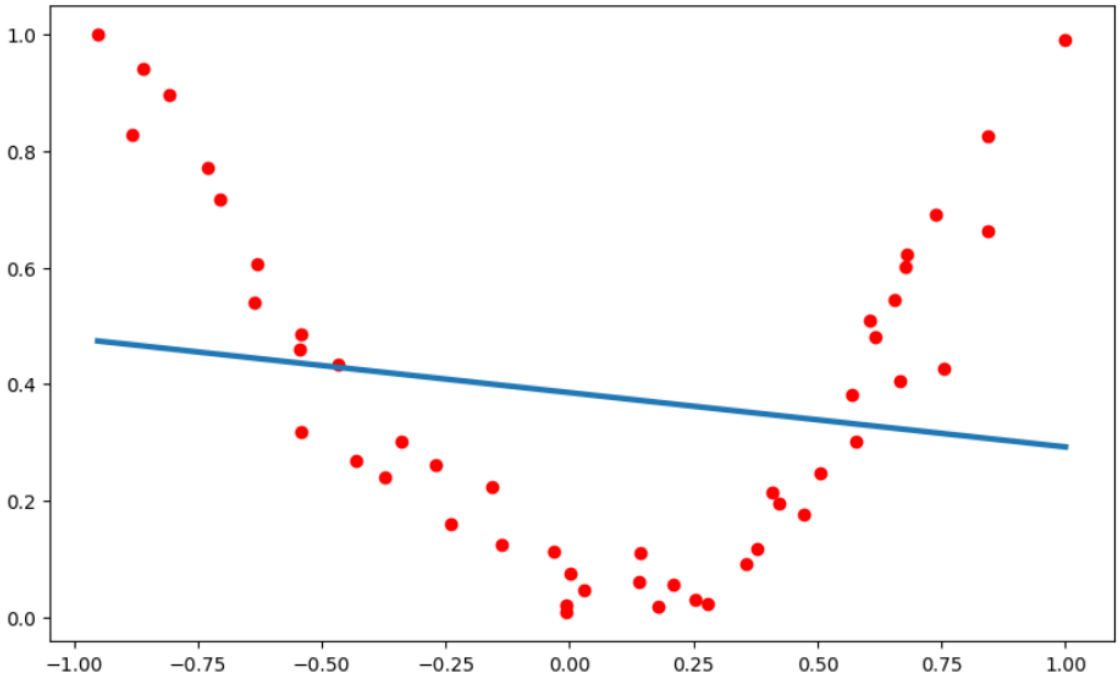
Keyingi qadamda chiziqli regressiya modeli quriladi va ma’lumotlar to‘plamiga moslashtiriladi. Polinomli regressiyani qurishda avval chiziqli

regressiya modeli uchun mos yozuvlar sifatida qabul qilinadi va ikkala natija taqqoslanadi.

```
from sklearn.linear_model import
LinearRegression
model=LinearRegression()
model.fit(x,y)
yhat=model.predict(x)
```

Keyingi qadamda chiziqli regressiya orqali bashorat chizig‘i natijasi vizualizatsiya qilinadi.

```
plt.figure(figsize=(10,6))
plt.plot(x,y,'ro‘,label='data')
plt.plot(x,yhat,linewidth=3.0, label='fit')
plt.show()
```



Yuqoridagi rasmga e’tibor berilsa, chiziqli regressiya chizig‘i ma'lumotlar to‘plamidan juda uzoqda ekanligi va ma’lumotlar to‘g‘ri chiziqda emasligi aniq bo‘ldi. Demak, to‘g‘ri chiziqdan boshqa, ma'lumotlar to‘plamiga mos keladigan egri chiziqli model qurish kerak bo‘ladi.

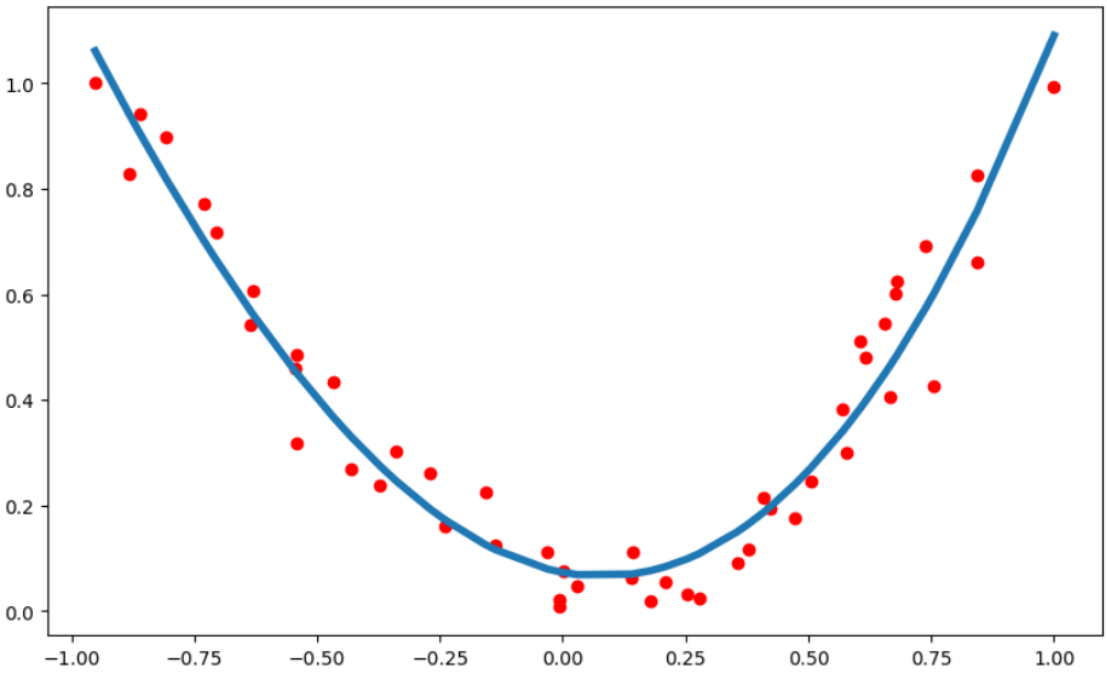
Polinomial regressiya modelini qurish

Polinomial regressiya modelini qurish uchun Python dasturlash tilida mavjud tayyor kutubxonadan foydalaniladi.

```
from sklearn.preprocessing import
PolynomialFeatures
poly_features=PolynomialFeatures(degree=3,
include_bias=False)
x_poly=poly_features.fit_transform(x)
model.fit(x_poly,y)
yhat=model.predict(x_poly)
```

Polinomli regressiya orqali olingan natija quyidagi kod orqali vizualizatsiya qilinadi.

```
plt.figure(figsize=(10,6))
plt.plot(x,y,'ro‘,label='data')
plt.plot(x,yhat,linewidth=4.0, label='fit')
plt.show()
```



Polinomial regressiya uchun absolut xatolik quyidagicha hisoblanadi.

```
print("MAE:", mean_absolute_error(yhat,y))
```

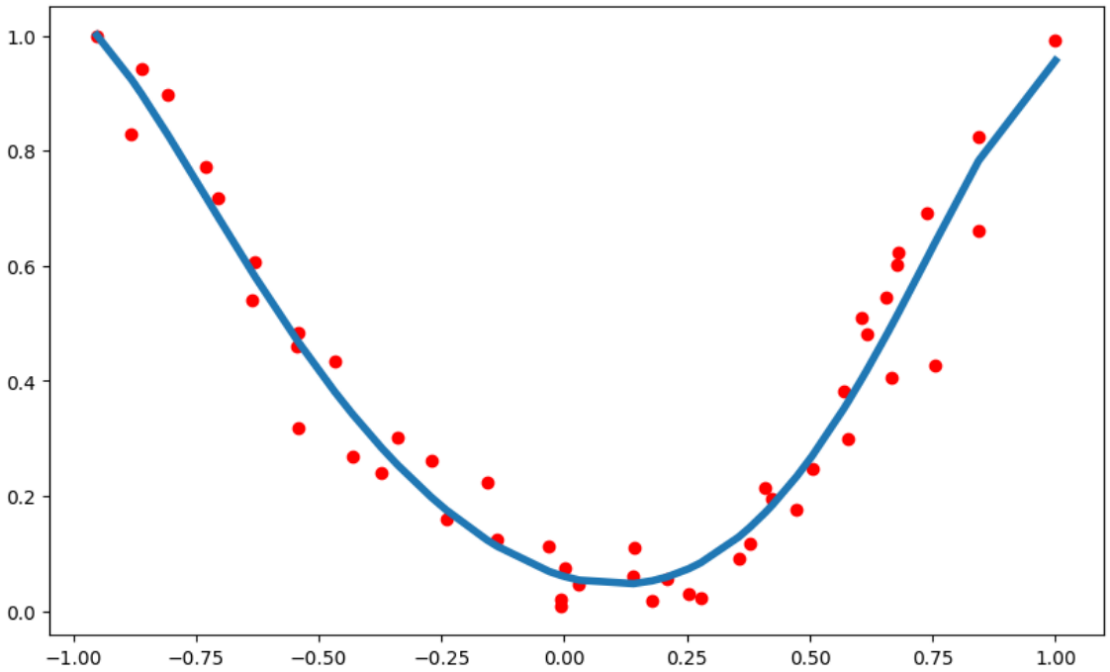
```
MAE: 0.06181689921211453
```

Polinomial regressiya tenglamasida daraja=3 uchun edi, agar darajani daraja=6 ga oshirsak natija yaxshilanishi mumkin. Darajani oshirish quyidagicha amalga oshiriladi.

```
from sklearn.preprocessing import
PolynomialFeatures
poly_features=PolynomialFeatures(degree=6,
include_bias=False)

x_poly=poly_features.fit_transform(x)

model.fit(x_poly,y)

yhat=model.predict(x_poly)
```

Polinomli regressiyaning oshirilgan darajasini natijasi quyidagi kod orqali vizualizatsiya qilinadi.

```
plt.figure(figsize=(10,6))
plt.plot(x,y,'ro‘,label='data')

plt.plot(x,yhat,linewidth=4.0, label='fit')

plt.show()
```



Polinomial regressiyaning yuqori darajasi ya’ni daraja=6 uchun xatolik quyidagicha hisoblanadi.

```
print("MAE:", mean_absolute_error(yhat,y))
```

```
MAE: 0.05401271200947631
```

Polinomial regressiyada darajani oshirish yuzaga keladigan xatolikni kamaytirishi ko‘rinib turibdi. Lekin darajani oshirish haddan tashqari moshlashish (overfitting) muammosiga ham olib kelishi mumkin.

Nazariy savollar

1. Polinomial regressiya, polinomial regressiya tenglamasini tushuntiring?

2. Chiziqli regressiya modelini qurishni tushuntiring?

3. Polinomial regressiyaga bo‘lgan ehtiyojni tushuntiring?

4. Polinomial regressiya modelini qurishni tushuntiring?

5. Polinomial regressiya natijasini yaxshilash jarayonini tushuntiring?

6. Polinomial regressiyada xatoliklarni baholash jarayonini tushuntiring?

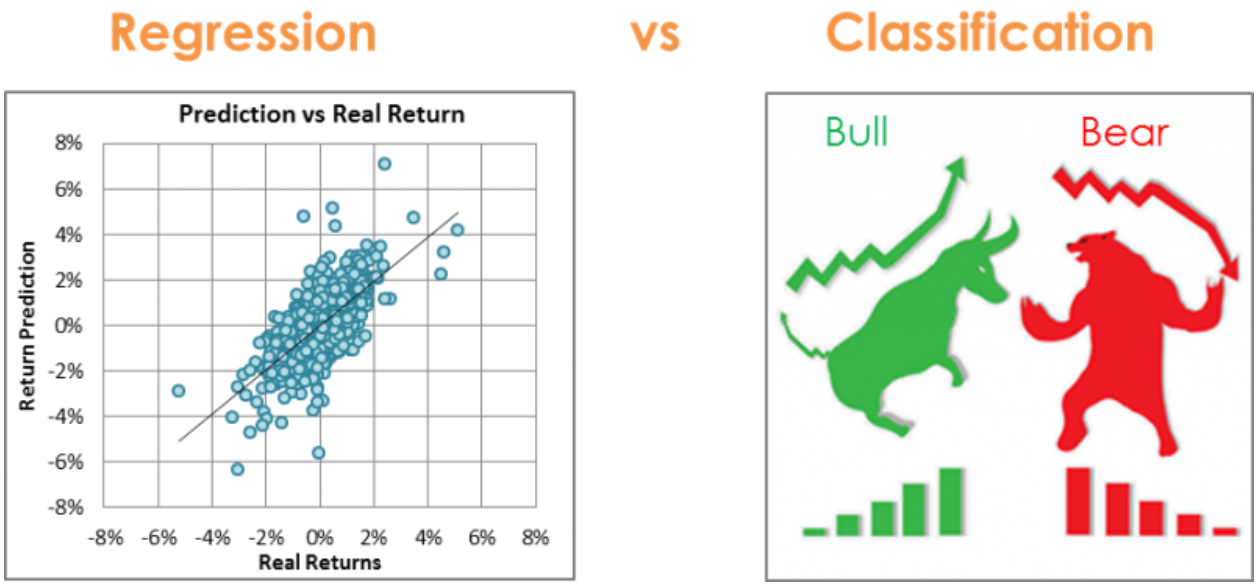
III-BOB. MASHINAVIY O‘QITISHDA KLASSIFIKATSIYA ALGORITMLARI

3.1. Mashinaviy o‘qitishda klassifikatsiya(tasniflash) algoritmi

Reja:

Klassifikatsiya algoritmi;
Mashinaviy o‘qitishda klassifikatsiya algoritmlarining turlari;
Mashinaviy o‘qitishda klassifikatsiya modelini baholash;
Aniqlik va eslab qolish(Precision va Recall);
F-1 Score;
Klassifikatsiya algoritmlaridan foydalanish holatlari.

Nazoratli mashinaviy o‘qitish algoritmini keng ma’noda regressiya va klassifikatsiya algoritmlariga ajratish mumkin. Regressiya algoritmlarida biz uzluksiz qiymatlar uchun chiziqni davomini aniqlash uchun bashorat natijalarini oldik, ammo kategorik qiymatlarni bashorat qilish uchun klassifikatsiya algoritmlaridan foydalanish kerak bo‘ladi. Regressiya va klassifikatsiya algoritmlarini farqini quyidagi rasm orqali aniqlash mumkin.



Klassifikatsiya algoritmi

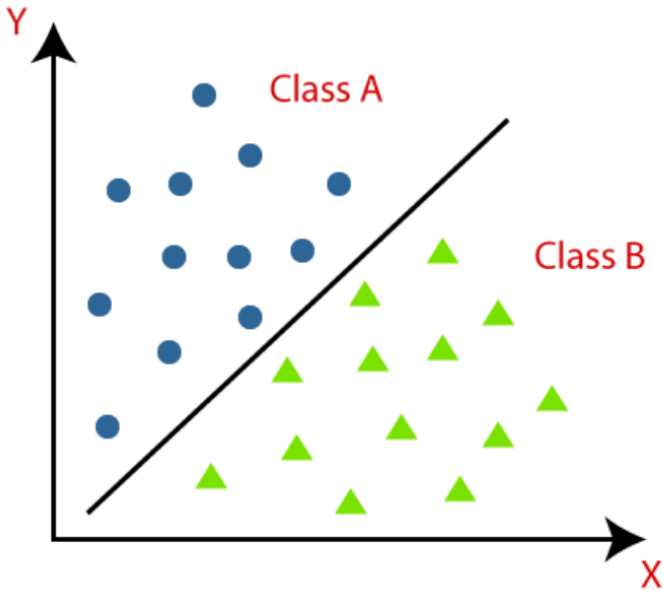
Klassifikatsiya - bu ma’lumotlar to‘plamini turli parametrlar asosida sinflarga bo‘lishda yordam beradigan funksiyani topish jarayoni

hisoblanadi. Klassifikatsiyada kompyuter dasturi o‘quv ma’lumotlar to‘plamida o‘qitiladi va shu o‘qitish asosida ma’lumotlarni turli sinflarga ajratadi. Masalan, Ha yoki Yo‘q, 0 yoki 1, Spam yoki Spam emas, mushuk yoki it va hokazo. Sinflarni maqsadlar/yorliqlar yoki toifalar deb atash mumkin.

Regressiyadan farqli o‘laroq, klassifikatsiyaning chiqish o‘zgaruvchisi "Yashil yoki ko‘k", "meva yoki hayvon" va boshqalar kabi qiymat emas, balki toifalar hisoblanadi. Klassifikatsiya algoritmi nazorat ostida o‘rganish usuli bo‘lgani uchun, u etiketli kirish ma’lumotlarini oladi, bu u tegishli chiqish bilan kirishni o‘z ichiga oladi degan ma’noni anglatadi.

Klassifikatsiya algoritmining asosiy maqsadi ma’lum bir ma'lumotlar to‘plamining toifasini aniqlashdir va bu algoritmlar asosan kategorik ma’lumotlarning chiqishini bashorat qilish uchun ishlatiladi.

Klassifikatsiya algoritmlarini quyidagi grafik orqali yaxshiroq tushunish mumkin. Quyidagi grafikda ikkita sinf, A sinf va B sinf mavjud. Bu sinflar bir-biriga o‘xshash va boshqa sinflarga o‘xshamaydigan xususiyatlarga ega.



Ma’lumotlar to‘plamida klassifikatsiyani amalga oshiradigan algoritm tasniflagich sifatida tanilgan. Klassifikatsiyaning ikki turi mavjud:

- **Binary klassifikator.** Agar klassifikatsiya muammosi faqat ikkita mumkin bo‘lgan natijaga ega bo‘lsa, u ikkilik klassifikator deb

ataladi.

Misollar: ha yoki yo‘q, erkak yoki ayol, spam yoki spam emas, mushuk yoki it va hokazo.

- **Multi-Class klassifikator.** Agar klassifikatsiya muammosi ikkitadan ortiq natijaga ega bo‘lsa, u ko‘p sinfli tasniflagich deb ataladi.

Misol: Ekin turlarining tasnifi, musiqa turlarining tasnifi va hokazo.

Mashinaviy o‘qitishda klassifikatsiya algoritmlarining turlari

Mashinaviy o‘qitishda klassifikatsiya algoritmlarini asosan ikkita toifaga ajratish mumkin:

1.Chiziqli modellar:

- Logistik regressiya;
- Vektorli mashinalarni qo‘llab-quvvatlash.

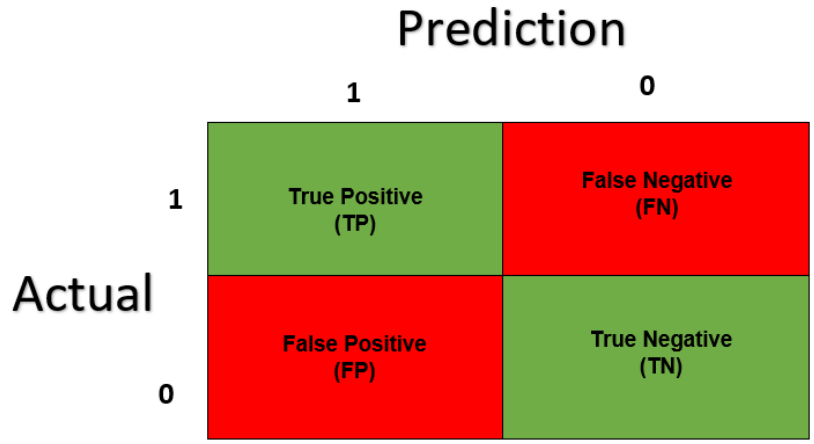
2.Chiziqli bo‘lmagan modellar:

- K-Eng yaqin qo‘shnilar;
- SVM;
- Naif Bayes;
- Qarorlar daraxti tasnifi;
- Tasodifiy o‘rmon tasnifi.

Mashinaviy o‘qitishda klassifikatsiya modelini baholash

Model tugallangandan so‘ng, uning ishlashini baholash kerak bo‘ladi. Har qanday mashinaviy o‘qitish modelini yaratishda eng muhim vazifa uning ishlashini baholash hisoblanadi. Shunday qilib, savol tug‘iladi: mashinaviy o‘qitish modelining muvaffaqiyatini qanday o‘lchash mumkin? O‘qitish va baholashni qachon to‘xtatish va qachon yaxshi deb atash kerakligini qayerdan bilamiz?

Klassifikatsiyada odatda to‘rt xil holat bo‘lishi mumkin, ya’ni kasallik mavjudlik bo‘yicha qaror qabul qilish masalasi bo‘yicha quyidagi rasmdagi holatlar bo‘lishi mumkin.



Kuzatishning haqiqiy va bashorat qilingan sinfi bilan bog‘liq to‘rtta toifasi quyidagicha bo‘lishi mumkin:

To‘g‘ri - ijobiy(TP): Berilgan kuzatishning to‘g‘ri – ijobiy qiymatida aslida kasallik mavjud, model ham mavjud javobini beradi.

Noto‘g‘ri - ijobiy (FP): Berilgan kuzatishning noto‘g‘ri – ijobiy qiymatida aslida kasallik mavjud, model mavjud emas javobini beradi.

To‘g‘ri - salbiy (TN): Berilgan kuzatishning to‘g‘ri – salbiy qiymatida aslida kasallik mavjud emas, model mavjud javobini beradi.

Noto‘g‘ri - salbiy (FN): Berilgan kuzatishning noto‘g‘ri – salbiy qiymatida aslida kasallik mavjud emas, model ham mavjud emas javobini beradi.

FP va FN tasniflash xatolaridir. **Statistikada FPlar I turdagi xatolar, FNlar esa II turdagi xatolar deb ataladi.** Ba’zi hollarda II turdagi xatolar xavfli bo‘lishi mumkin.

Misol uchun, agar bizning tasniflagichimiz uyda yong‘in sodir bo‘lishini bashorat qilsa, I turdagi xatolik noto‘g‘ri signal hisoblanadi.

Boshqa tomondan, II turdagi xato uy yonib ketayotganini va o‘t o‘chirish bo‘limi xabardor emasligini anglatadi.

Chalkashlik matritsasi (Confusion matrix). Chalkashlik matritsasi bizga chiqish sifatida matritsa/jadval beradi va modelning ishlashini tavsiflaydi.

Matritsa to‘g‘ri bashorat va noto‘g‘ri bashoratlarning umumiy soniga ega bo‘lgan umumlashtirilgan shakldagi bashorat natijalaridan iborat bo‘ladi. Matritsa quyidagi jadvalga o‘xshaydi:

n=165	**Predicted: NO**	**Predicted: YES**	
Actual: NO	TN = 50	FP = 10	60
Actual: YES	FN = 5	TP = 100	105
	55	110	

Shunday qilib, tasniflash modelini baholash usullari quyida keltirib o‘tilgan.

Aniqlilik(Accuracy)

Accuracy - eng oddiy va tushunarli tasniflash ko‘rsatkichi bo‘lib, **to‘g‘ri tasniflangan kuzatuvlarning ulushini o‘lchaydi**, va quyidagi formula orqali ifodalanadi:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{N}$$

Bunda, N – modelning aniqligini hisoblayotgan to‘plamdagi elementlat soni.

$$N = TP + TN + FP + FN.$$

Oddiy qilib aytganda, u xato darajasining yo‘qligini o‘lchaydi. 90% aniqlilik 100 ta kuzatishdan 90 tasi to‘g‘ri tasniflanganligini bildiradi. Ushbu o‘lchov ma’lumotlar to‘plamidagi pozitiv va negativ nishonlarning qiymatlari teng bo‘lganda o‘rinlidir.

Chalkashlik matritsasi nuqtai nazaridan o‘ylab, biz ba'zan "barcha namunalar bo‘yicha diagonal" deymiz. Oddiy qilib aytganda, u xato darajasining yo‘qligini o‘lchaydi. 90% aniqlik 100 ta kuzatishdan 90 tasi to‘g‘ri tasniflanganligini bildiradi.

Garchi 90% aniqlik dastlab juda yaxshi bo‘lib tuyulsa ham, nomutanosib ma’lumotlar to‘plamida aniqlilik o‘lchovidan foydalanish noto‘g‘ri bo‘lishi mumkin.

Aniqlik va eslab qolish(Precision va Recall)

Aniqlilik (Precision) - ijobiy deb belgilangan va aslida ijobiy bo‘lgan holatlarning ulushi. Boshqacha qilib aytganda, aniqlilik “bizning tasniflagichimiz natijalari qanchalik foydali" o‘lchovlari hisoblanadi va u quyidagi formula orqali ifodalanadi:

$$\Pr ecision = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{F}P}$$

Masalan, 90% aniqlilikni bizning tasniflagichimiz elektron pochtani spam deb belgilaganida, u haqiqatan ham 100 martadan 90 tasi spam ekanligini bildiradi.

TPlarni aniqlashning yana bir usuli - eslab qolishdan foydalanish.

Eslab qolish (**Recall)** - ijobiy deb belgilangan haqiqiy ijobiy holatlarning ulushi. U “natijalar qanchalik to‘liqligini” o‘lchaydi, ya’ni haqiqiy ijobiylarning qaysi foizi ijobiy deb taxmin qilinadi va u quyidagi formula orqali ifodalanadi:

$$Recall = \frac{TP}{TP + FN}$$

Ya’ni, 90% ni chaqirib olish, klassifikator barcha spam elektron pochta xabarlarining 90% ni to‘g‘ri belgilashini anglatadi, shuning uchun 10% spam emas deb belgilangan bo‘ladi.

F-1 Score

F-1 Score Precision va Recallning garmonik o‘rtacha ko‘rsatkichidir. Bu I va II turdagi xatolarga teng ahamiyat beradi va u quyidagi formula orqali ifodalanadi:

$$F - 1Score = \frac{2 * Precision * Recall}{Precision + Recall}$$

Ma’lumotlar to‘plamida pozitiv va negativ nishonlar notekis taqsimlanganda F-1 Score eng aniq o‘lchovlardan biri hisoblanadi.

Klassifikatsiya algoritmlaridan foydalanish holatlari

Klassifikatsiya algoritmlari turli joylarda ishlatilishi mumkin. Quyida tasniflash algoritmlaridan keng foydalaniladigan holatlar keltirilgan:

- Elektron pochta spamlarini aniqlash;

- Nutqni aniqlash;
- Saraton o‘simta hujayralarini aniqlash;
- Dori vositalarining tasnifi;
- Biometrik identifikatsiya va boshqalar.

Nazariy savollar

1.Nazoratli mashinaviy o‘qitish algoritmlari?

2.Klassifikatsiya algoritmi va uning vazifasi?

3.Mashinaviy o‘qitishda klassifikatsiya algoritmlarining turlari va uning vazifalari?

4.Chiziqli va chiziqli bo‘lmagan modellarni tushuntiring?

5.Mashinaviy o‘qitishda klassifikatsiya modelini baholash jarayonini tushuntiring?

6.Chalkashlik matritsasi va uning vazifasini tushuntiring?

7.Aniqlik va eslab qolish(Precision va Recall) baholash usuli va uning vazifasini tushuntiring?

8.F-1 Score baholash usuli va uning vazifasini tushuntiring?

9.Klassifikatsiya algoritmlaridan foydalanish holatlarini tushuntiring?

3.2. Mashinaviy o‘qitishning klassifikatsiya algoritmlaridan K-Eng yaqin qo‘shnilar (KNN - K-Nearest Neighbor) algoritmi

Reja:

K-NN algoritmi haqida;

K-NN algoritmi;

K-NN algoritmida K qiymatini tanlash;

K-NN algoritmini Python dasturlash tili orqali amalga oshirish;

K-NN algoritmini amalga oshirish bosqichlari;

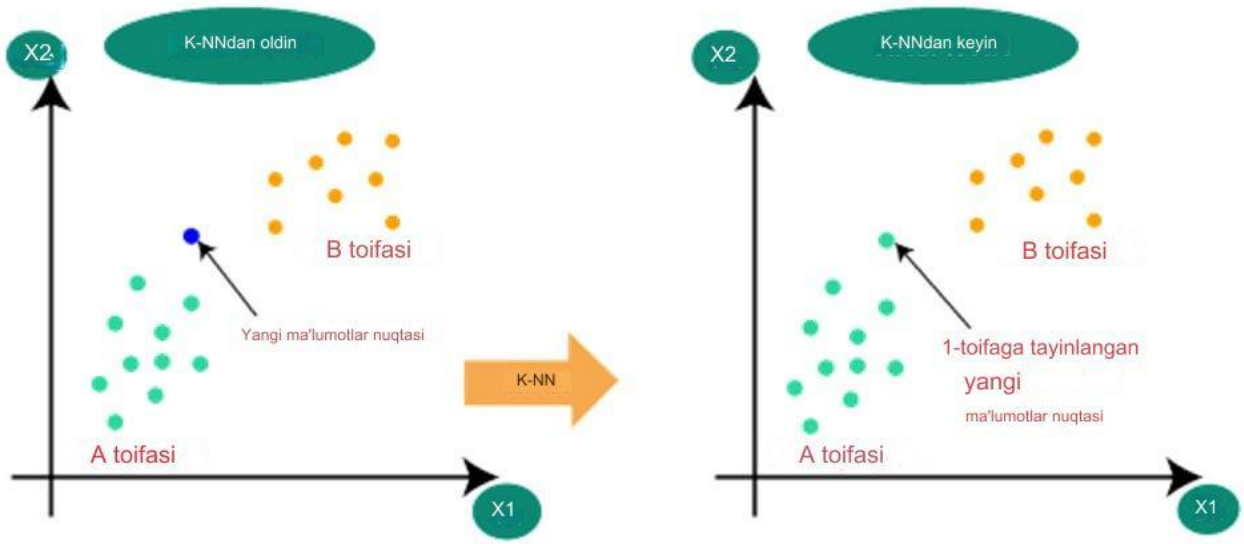
K ning eng yaxshi qiymatini aniqlash.

K-NN algoritmi haqida

KNN algoritmi oddiy, amalga oshirish oson, nazorat ostidagi mashinaviy o‘qitish algoritmi bo‘lib, u ham tasniflash, ham regressiya muammolarini hal qilish uchun ishlatilishi mumkin bo‘lgan algoritm hisoblanadi.

- K-NN algoritmi yangi ma’lumotlar va mavjud ma’lumotlar o‘rtasidagi o‘xshashlikni aniqlaydi va yangi ma’lumotni mavjud toifalar orasidan eng o‘xshash toifaga kiritishni amalga oshiradi.
- K-NN algoritmi regressiya uchun ham, tasniflash uchun ham ishlatilishi mumkin, lekin u asosan tasniflash muammolari uchun ishlatiladi.
- K-NN parametrik bo‘lmagan algoritm hisoblanib, u ma'lumotlarning taqsimlanishi haqida hech qanday asosiy bashoratlar qilmaydi.

K-NN algoritmi nima uchun kerak. Aytaylik, ikkita toifa, ya’ni A va B toifalari mavjud va bizda yangi ma’lumotlar bor, bu ma’lumotlar ushbu toifalarning qaysi birida joylashishini aniqlashimiz kerak. Ushbu turdagi muammolarni hal qilish uchun bizga K-NN algoritmi kerak bo‘ladi. K-NN algoritmi yordamida ma’lum bir ma’lumotlar to‘plamining toifasi yoki sinfini osongina aniqlash mumkin. Bu jarayonni quyidagi grafik orqali tushinib olish mumkin.



K-NN algoritmi

K-NN algoritmining ishlashini quyidagi qadamlar asosida tushunish mumkin:

1-qadam: K-NN uchun eng yaxshi K tanlash;

2-qadam: K qo‘shnilarning Evklid masofasini hisoblash;

3-qadam: Hisoblangan Evklid masofasiga ko‘ra K eng yaqin qo‘shnilarni olish;

4-qadam: Ushbu K qo‘shnilar orasida har bir toifadagi ma'lumotlar nuqtalarining sonini hisoblash;

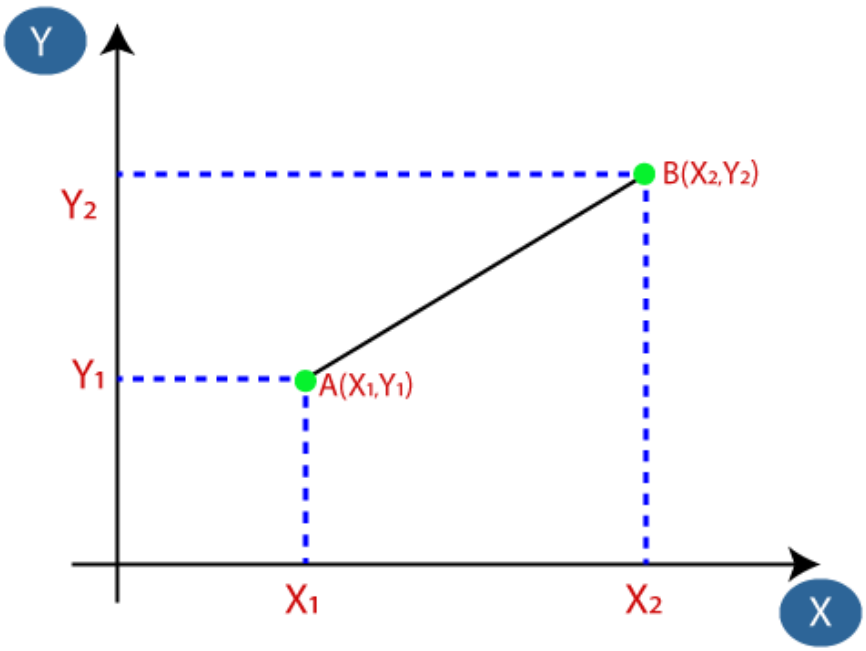
5-qadam: Yangi ma’lumotlar nuqtalarini qo‘shni soni maksimal bo‘lgan toifaga belgilash.

6-qadam: Modelni tayyorlash.

Dastlab, qo‘shnilar soni tanlanadi, keltirilgan misol uchun k = 5 tanlanadi. K-NN algoritmining asosiy e’tibor beriladigan joyi ham shu K eng yaqin qo‘shnilarni aniqlash hisoblanadi. Dastlab keltiriladigan misolda faqat K=5 sifatida foydalaniladi, keyinchalik K qiymat sikl orqali aniqlash ham qarab o‘tiladi.

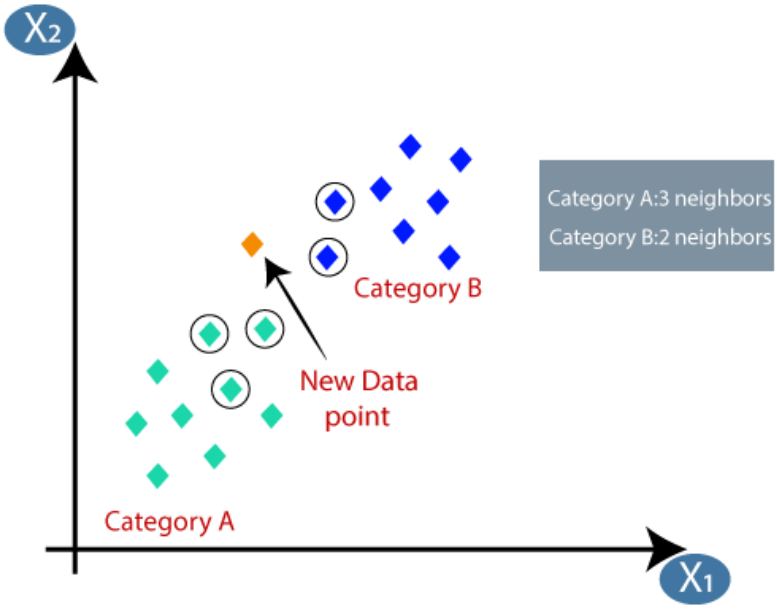
Keyinchalik, ma’lumotlar nuqtalari orasidagi Evklid masofasi hisoblanadi.

Bu jarayon quyidagicha amalga oshirilishi mumkin:



Masofani hisoblash formulasi an’anaviy ikki nuqta orasidagi masofani hisoblash: $A_1B_1 = \sqrt{(X_2 - X_1)^2 + (Y_2 - Y_1)^2}$

Evklid masofasi hisoblanadi va eng yaqin qo‘shnilarni, ya'ni A toifadagi uchta eng yaqin qo‘shni va B toifadagi ikkita eng yaqin qo‘shnilar olinadi. Bu jarayonni quyidagi grafik orqali ko‘rish mumkin.



Ko‘rib turganimizdek, eng yaqin uchta qo‘shni A toifasiga kiradi, shuning uchun bu yangi ma’lumotlar nuqtasi A toifasiga tegishli bo‘lishi kerak.

K-NN algoritmida K qiymatini tanlash

K uchun eng yaxshi qiymatni aniqlashning alohida usuli mavjud emas, shuning uchun ulardan eng yaxshisini topish uchun ba’zi qiymatlarni sinab ko‘rish kerak bo‘ladi. K uchun eng maqbul qiymat 5 ga teng.

K = 1 yoki K = 2 kabi qiymatlar K uchun juda past qiymat bo‘lishi mumkin. K uchun katta qiymatlar yaxshi, lekin u ba’zi qiyinchiliklarga ham olib kelishi mumkin.

K-NN algoritmini Python dasturlash tili orqali amalga oshirish

Mashinaviy o‘qitishda klassifikatsiya (tasniflash) algoritmlarini amaliy jarayonlarda qo‘llash uchun avval kerakli ma’lumotlar to‘plami (dataset) tanlab olinadi. Ushbu qaralayotgan ma’lumotlar to‘plami bemorlarning ma’lumotlari asosida olingan. Maqsad tashxislash o‘lchovlari asosida ayol bemorda insult bor-yo‘qligini aniqlashdan iborat.

Qaralayotgan ma’lumotlar to‘plamidagi ustunlar va ularning mazmuni:

- Sex:- Jinsi (0-ayol,1-erkak)
- age: Yoshi(butun sonda)
- hypertension: Gipertenziya (0 - yo‘q, 1 - bor)
- heart_disease: Yurak kasalligi (0 - yo‘q, 1 - bor)
- ever_1: Oldingi kasallik tarixi bo‘lganmi (0 - yo‘q, 1 - bor)
- work_type: Ish turi(butun sonda)
- Residence_type: Yashash joyi turi (1-Shahar, 0-Qishloq)
- avg_glucose_level: O‘rtacha qondagi shakar miqdori
- bmi: Tana massasi indeksi (haqiqiy sonda)
- smoking_status: Chekish holati (1-chekadi, 0- yo‘q)
- Stress Levels: Stress darajasi (haqiqiy sonda)
- Marital Status: Oilaviy holati (0-bo‘ydoq, 1-oilaviy, 2-ajrashgan)
- Heart Disease: Yurak kasalligi (0-yo‘q, 1-bor)
- Physical Activity: Jismoniy faollik darajasi (butun sonda)
- Stroke: Insult diagnostik natijasi (0 - yo‘q, 1 - bor)

K-NN algoritmini amalga oshirish bosqichlari

K-NN algoritmini amalga oshirish bosqichi quyidagi qadamlarda amalga oshiriladi.

- Ma’lumotlarni oldindan qayta ishlash bosqichi, ma’lumotlar to‘plamini shakllantirish;
- K-NN algoritmini o‘quv to‘plamiga moslashtirish;
- Sinov natijasini bashorat qilish;
- Natijaning to‘g‘riligini tekshirish;
- Test to‘plamining natijasini vizualizatsiya qilish.

1.Ma'lumotlarni oldindan qayta ishlash bosqichi:

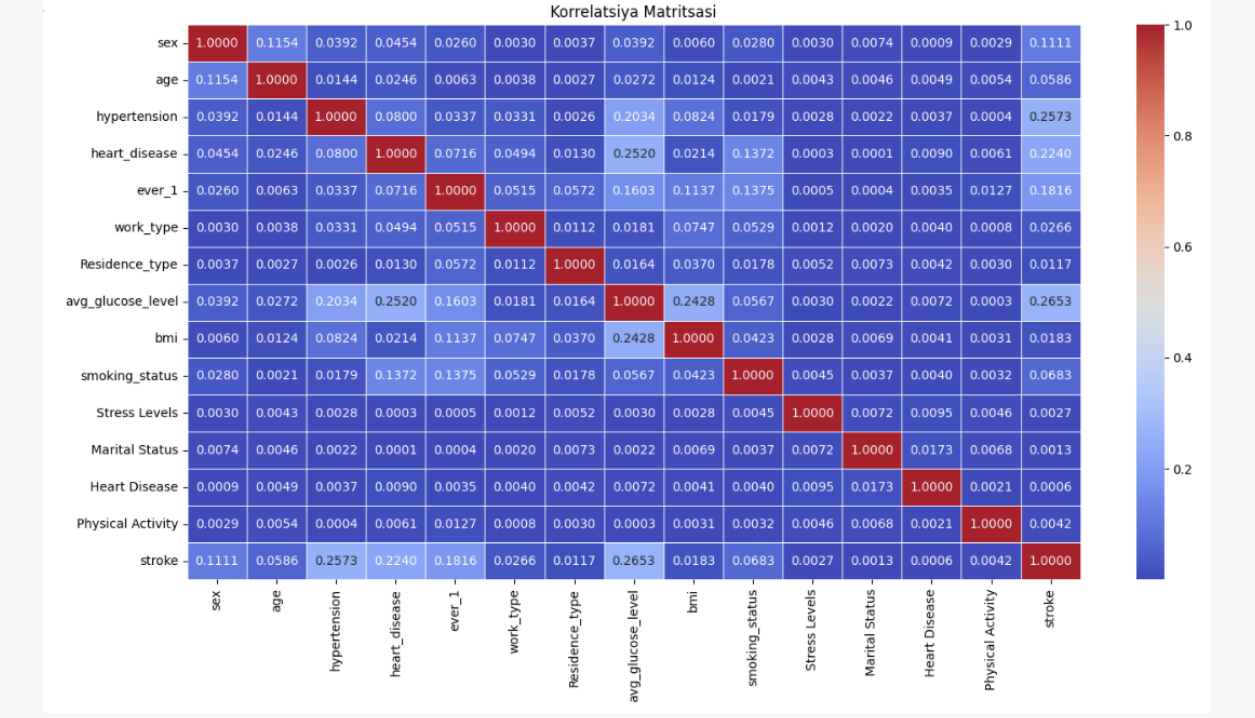
1) Dastlab kerakli **pandas va numpy** kutubxonalari faollashtiriladi va tayyorlangan ma’lumotlar to‘plami chaqirib olinadi.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
df=pd.read_csv("StrokeData.csv")

df.head(5)
```

	sex	age	hypertension	heart_disease	ever_1	work_type	Residence_type	avg_glucose_level	bmi	smoking_status	Stress Levels	Marital Status	Heart Disease	Physical Activity	stroke
0	1	63	0	1	1	4	1	228.69	36.6	1	3.48	1	1	2	1
1	1	42	0	1	1	4	0	105.92	32.5	0	1.73	0	0	1	1
2	0	61	0	0	1	4	1	171.23	34.4	1	7.31	1	1	3	1
3	1	41	1	0	1	3	0	174.12	24.0	0	5.35	1	0	2	1
4	1	85	0	0	1	4	1	186.21	29.0	1	6.84	2	1	1	1

Korrelyasiya. Keyingi qadamda berilgan ma’lumotlar to‘plamida “**stroke**” ustuniga tasir qiluvchi ustunlarni topish uchun korrelyatsiya jarayoni hisoblanadi.

```
corr_matrix=df.corr().abs()
corr_matrix.style.background_gradient(cmap='cool warm')
```



Yuqoridagi natijadan ko‘rinib turibdiki ‘stroke’ ustuniga ‘avg_glucose_level’ ustuni nisbatan ko‘proq tasir qilmoqda.

Ma’lumotlar to‘plamidagi ma’lumotlarni ‘X’ (‘stroke’ ustunidan boshqa barcha ustundagi qiymatlar) va ‘Y’ (‘stroke’ ustunidagi qiymatlar) o‘zgaruvchilarga ta’minlanadi.

```
x = df.drop('stroke', axis=1).values
y = df['stroke']
```

Keyingi qadamda **StandartScaler** funksiyasi yordamida ma’lumotlarni standart (kichik oraliqga) holatga keltirib olinadi.

```
from sklearn.preprocessing import StandardScaler
scaler=StandardScaler()
x=scaler.fit_transform(x)
```

Keyingi qadamda K-NN klassifikatorini o‘quv ma’lumotlariga moslashtiriladi. Buning uchun **SklearnNeighbor** kutubxonasining **KNeighborsClassifier** sinfi import qilinadi. Sinf import qilgandan so‘ng sinfda K-NN ob’ekti yaratiladi. Ushbu sinf quyidagi parametrlardan iborat bo‘ladi:

- **n_neighbours**: Algoritmning kerakli qo‘shnilarini aniqlash uchun ishlatiladi. Odatda, bu 5 qiymatini oladi.

- **metric=”minkowski”**: Bu ham standart parametr bo‘lib, nuqtalar orasidagi masofani belgilaydi(quyida bu parametrdan foydalanilmgan).

Keyingi qadamda klassifikator o‘quv ma’lumotlariga moslashtiriladi va ma’lumotlar to‘plamidan 20%ni test va 80%ni o‘quv to‘plamalariga ajratiladi.

```
from sklearn.model_selection import
train_test_split

x_train,x_test,y_train,y_test=train_test_split(x
,y, test_size=0.2, random_state=12)

from sklearn.neighbors import
KNeighborsClassifier

KNN=KNeighborsClassifier(n_neighbors=5)

KNN.fit(x_train, y_train)
```

Out[9]:

```
▾ KNeighborsClassifier
KNeighborsClassifier()
```

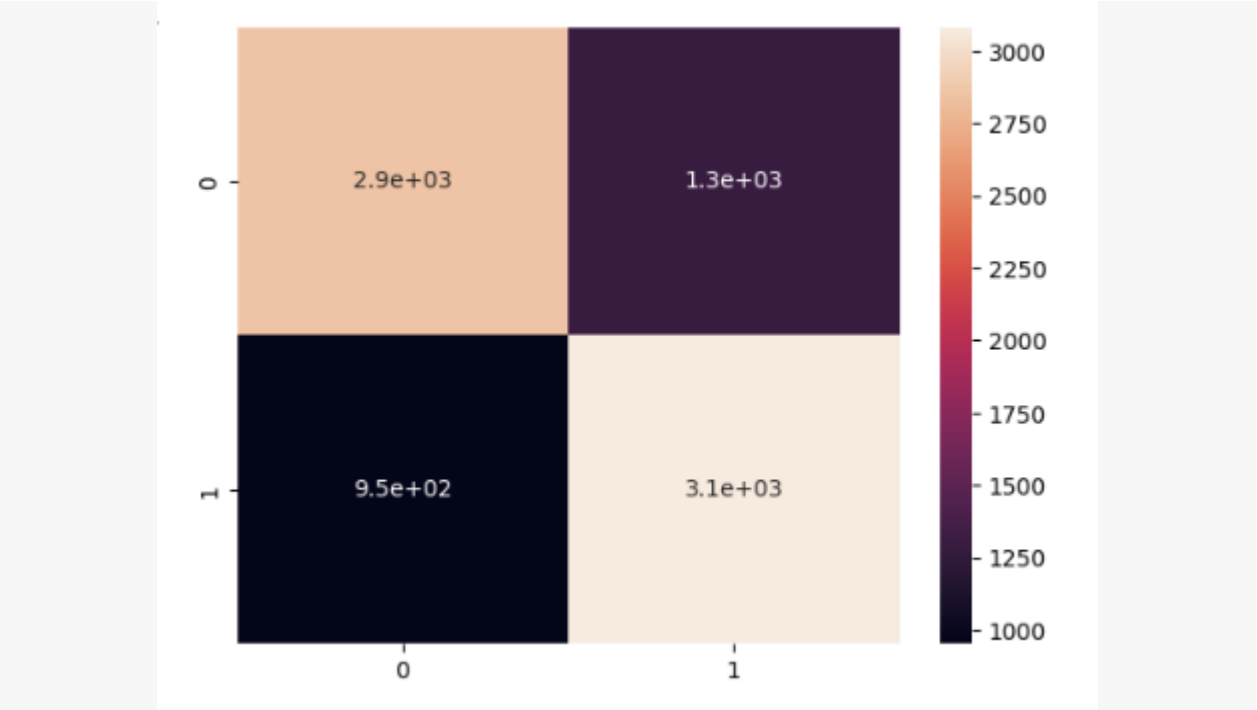
Keyingi qadamda **sinov natijasini bashorat qilish**: Test to‘plami natijasini bashorat qilish uchun **y_predict** vektori yaratiladi.

```
y_predict=KNN.predict(x_test)
```

Bashorat natijasi y_predict vektorida saqlanadi.

Chalkashlik (Confusion) matritsasini yaratish. Keyingi qadamda KNN modeli uchun klassifikatorning aniqligini ko‘rish uchun **confusion** matritsasi yaratiladi.

```
from sklearn.metrics import confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt
sns.heatmap(confusion_matrix(y_test, y_predict),
annot= True)
plt.show()
```



Keyingi qadamda bashorat natijasi olingandan so‘ng, uni yuqoridagi mavzuda tushuntirilgan **Accuracy, Precision Recall, F-1 Score** yordamida baholash amalga oshiriladi.

```
from sklearn.metrics import precision_score,
recall_score, f1_score, accuracy_score

precision= precision_score(y_test,y_predict)
recall=recall_score(y_test,y_predict)
f1=f1_score(y_test,y_predict)
accuracye=accuracy_score(y_test,y_predict)

print(f"{precision=}\n{recall=}\n{f1=}\n{accurac
ye=}")
```

Natija.

```
precision=0.7034923533439854
recall=0.7640059494298463
f1=0.7325014854426619
accuracye=0.7249511241446726
```

Yuqoridagi dastur kodida **accuracy** bergan natijadan ko‘rinib turiptiki **K-NN algoritmi** asosida yaratilgan dasturning natijasi `72.49`% aniqlikda ishlamoqda.

Yuqoridagi modelni ishlab chiqishda K-NN algoritmi uchun K ning qiymatini 5 ga teng deb olingan, lekin **K ning eng yaxshi qiymatini** aniqlash uchun quyidagi usullarni taqdim etiladi.

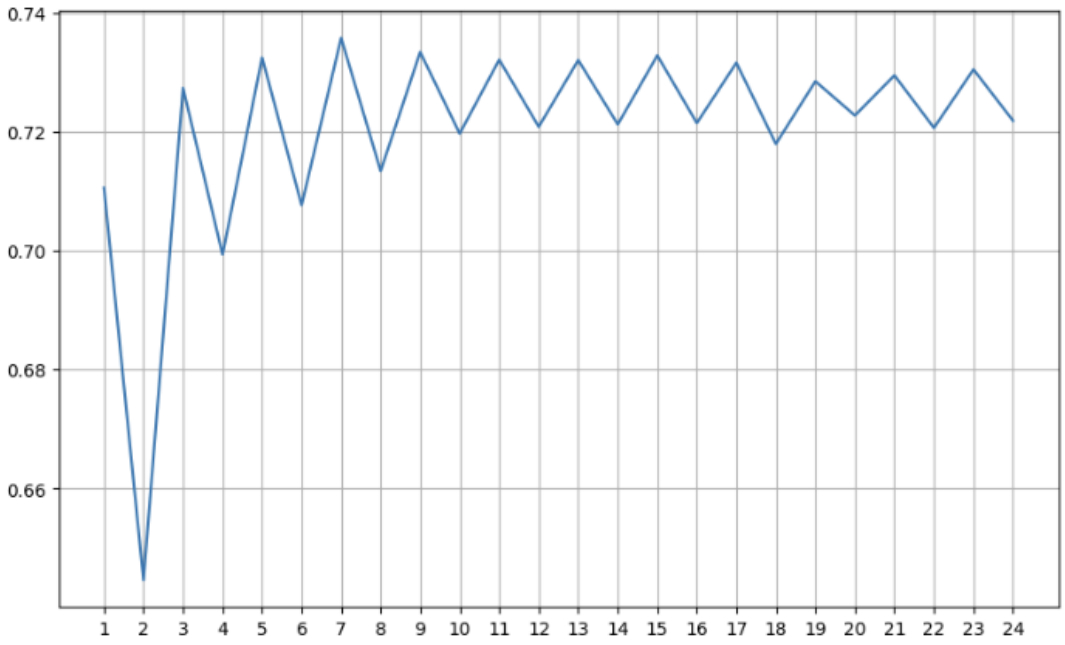
K ning eng yaxshi qiymatini aniqlash

K ning eng yaxshi qiymatini aniqlashning ikki xil **for operatori** va **GridSearchCV funksiyasi** orqali amalga oshirish mumkin.

1.**For operatori yordamida:** for operatori yordamida K ning qiymatlarini 1 dan 25 oraliqda natijasini solishtirib chiqiladi.

```
f1=[]
for k in range(1,25):
  KNN=KNeighborsClassifier(n_neighbors=k)
  KNN.fit(x_train, y_train)
  y_predict=KNN.predict(x_test)
  f1.append(f1_score(y_test,y_predict))
plt.figure(figsize=(10,6))
plt.plot(range(1,25),f1)
plt.xticks(range(1,25))
plt.grid()
plt.show()
```

Yuqoridagi dastur kodi ishlagandan so‘ng hosil bo‘lgan grafik orqali **K ning eng yaxshi qiymatini** aniqlash mumkin.



Bu yerda baholash **F-1 Score** orqali aniqlanmoqda, ko‘rinib turibdiki K=5,7, 9 qiymatlarga teng bo‘lganda aniqlik yuqori baholanmoqda. Modelni qurishda yuqoridagi kodda `KNN=KNeighborsClassifier(n_neighbors=7)` qilib kiritilsa natija quyidagicha yaxshilanganligini ko‘rish mumkin.

```
precision=0.7055745164960182
recall=0.7687159147248389
f1=0.7357930952663424
accuracye=0.7278836754643206
```

K ni qiymati yaxshilangandan so‘ng **K-NN algoritmi** asosida yaratilgan dasturning natijasi `72.78`% aniqlikda ishlamoqda.

2. K ning eng yaxshi qiymatini topish uchun **sklearn** kutubxonasi tarkibida tayyor **GridSearchCV** funksiyasi ham mavjud, u orqali K ni qiymati quyidagicha aniqlanadi.

```
from sklearn.model_selection import GridSearchCV
param_grid={'n_neighbors': np.arange(1,25)}
KNN_gscv=GridSearchCV(KNN, param_grid, cv=5)
KNN_gscv.fit(x,y)
```

Yuqoridagi kodning natijasi asosida **Best_params_** funksiyasi orqali K ning eng yaxshi qiymati aniqlanadi.

```
KNN_gscv.best_params_
```

Natija.

```
{'n_neighbors': 7}
```

Natijadan ko‘rinib turibdiki funksiya ham K ning eng yaxshi qiymatini 7 ekanligini ko‘rsatmoqda.

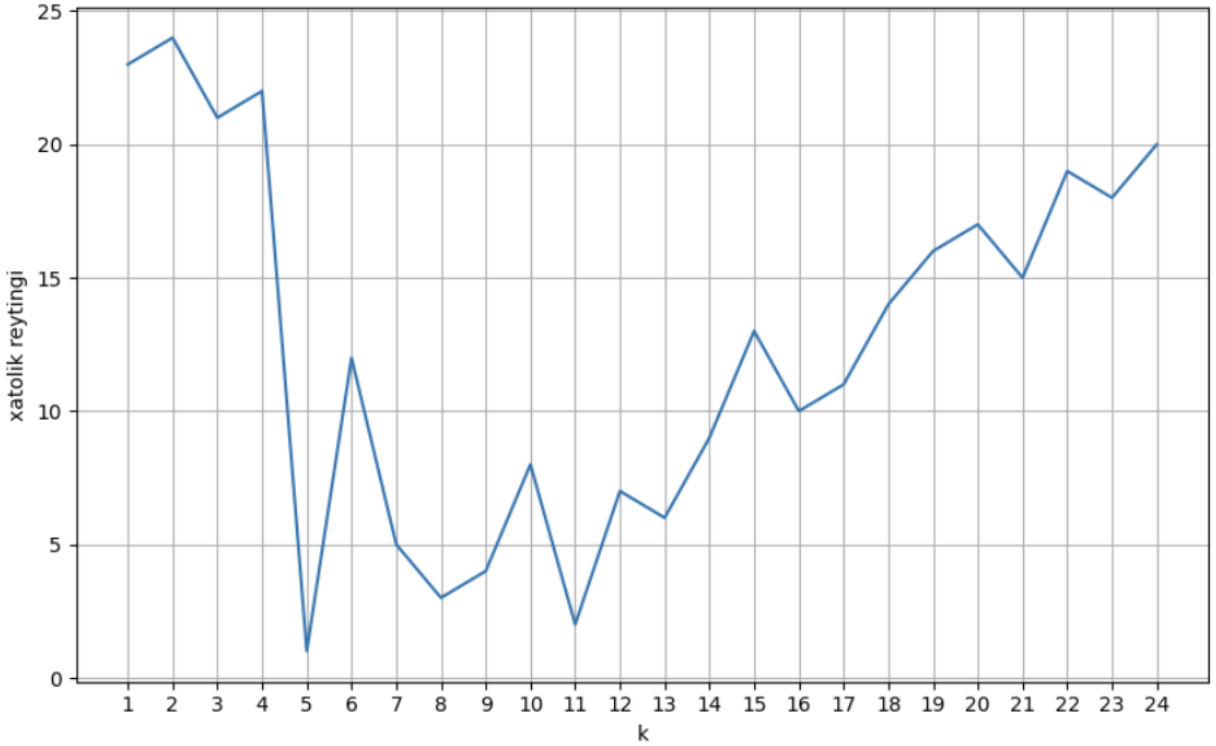
Best_score_ funksiyasi orqali esa K ning eng yaxshi 7 ga teng qiymatida model aniqliligini baholash mumkin.

```
KNN_gscv.best_score_
```

Natija: `0.7278836754643206`

Natijadan ko‘rinib turibdiki funksiya K=7 bo‘lganda aniqlilik 72.78% ni tashkil etganligi aniqlandi. Bu natijani xatolik grafigi orqali ham tasvirlash mumkin.

```
plt.figure(figsize=(10,6))
plt.plot(param_grid['n_neighbors'],
KNN_gscv.cv_results_['rank_test_score'])
plt.xticks(param_grid['n_neighbors'])
plt.xlabel("k")
plt.ylabel("xatolik reytingi")
plt.grid()
plt.show()
```



Yuqoridagi grafikda modelning xatolik darajasi ko‘rsatib o‘tilgan, K ning qiymati 5 ga teng bo‘lganda eng kam xatolikga erishilmoqda.

Natijada K-NN algoritmi asosida qurilgan mashinaviy o‘qitish modelida K ning eng yaxshi qiymati 5 ga teng bo‘ldi. Endi yuqoridagi ma’lumotlar to‘plamiga mos ixtiyoriy bitta yozuvli ma’lumotlar x_test o‘rniga o‘qitilsa, kasallik bor yoki yo‘q ekanligini aniqlash mumkin bo‘ladi.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
df=pd.read_csv("StrokeData.csv")
x = df.drop('stroke', axis=1).values
y = df['stroke']
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y, test_size=0.2,
random_state=12)
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y, test_size=0.2,
random_state=12)
from sklearn.neighbors import KNeighborsClassifier
KNN=KNeighborsClassifier(n_neighbors=5)
KNN.fit(x_train, y_train)

data1={'sex':[1],'age':[55],
'hypertension':[1],'heart_disease':[0],'ever_1':[0],'work_type':[4],'Resid
ence_type':[0],'avg_glucose_level':[121],'bmi  ':[28],'smoking_status  ':[
0],'Stress Levels  ':[5.35],'Marital Status  ':[1],'Heart
Disease':[5.35],'Physical Activity':[3]};
df1=pd.DataFrame(data1)
x_test1=df1
y_predict=KNN.predict(x_test1)
y_predict
```

Natija.

array([0])

Demak *array([0])* yuqoridagi dastur kodi natijasi bo‘yicha yangi kiritilgan ma’lumotlar asosida kasallik yo‘q ekanligi aniqlandi.

Nazariy savollar

1. K-NN algoritmi va uning vazifasini tushuntiring?

2. K-NN algoritmi va uning vazifasinigrafik orqali tushuntiring?

3. K-NN algoritmining qadamlarini tushuntiring?

4. K-NN algoritmida K qiymatning vazifasi va uni tanlashni tushuntiring?

5. K-NN algoritmini Python dasturlash tili orqali amalga oshirish bosqichlari va dasturini ifodalang?
6. K-NN algoritmini amalga oshirish bosqichlari;
7. Chalkashlik (Confusion) matritsasini yaratish va uning vazifasini tushuntiring?
8. Bashorat natijasi Accuracy, Precision Recall, F-1 Score yordamida baholash jarayonini tushuntiring?
9. K ning eng yaxshi qiymatini aniqlash usullarini tushuntiring?
10. For operatori yordamida K ni eng yaxshi qiymatini aniqlash ni tushuntiring?

3.3. Qaror daraxti (Decision Tree) klassifikatsiya algoritmi

Reja:

Qaror daraxti (Decision Tree);

Qarorlar daraxtidan foydalanish sababi;

Qaror daraxtining afzalliklari;

Qaror daraxtining kamchiliklari;

Qaror daraxti algoritmini Python dasturlash tilida amalga oshirilishi;

Qaror daraxti algoritmini amalga oshirish bosqichlari.

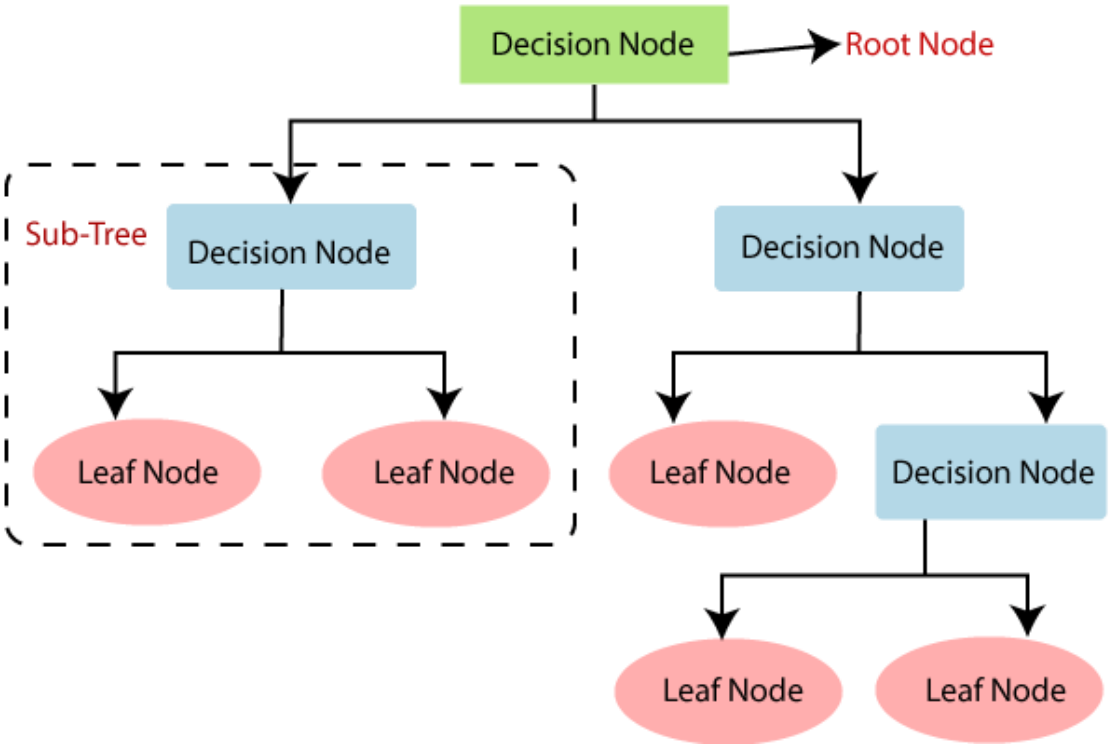
Qaror daraxti (Decision Tree)

Nazorat ostidagi mashinaviy o‘qitishning yana bir klassifikatsiya algoritmlaridan biri qaror daraxti algoritmi hisoblanadi.

Qaror daraxti (Decision Tree) - bu tasniflash va regressiya muammolari uchun ishlatilishi mumkin bo‘lgan nazoratli o‘qitish usuli, lekin asosan tasniflash muammolarini hal qilish uchun yaxshi natijalarni ko‘rsatadi.

Qaror daraxti muammoni hal qilish uchun daraxt tasviridan foydalanadi. U qaror daraxti deb atalishini sababi, daraxtga o‘xshab, ildizdan boshlanadi va u keyingi shoxlarda tarmoqlanadi hamda daraxtga o‘xshash tuzilmani yaratadi. Bunda qarorlar yoki testlar berilgan ma’lumotlar to‘plamining xususiyatlari asosida amalga oshiriladi. Qaror daraxti berilgan shartlar asosida muammo/qarorning barcha mumkin bo‘lgan yechimlarini olish uchun **grafik tasvirni** hosil qiladi. Qaror daraxti shart beradi va javobga (Ha/Yo‘q) asoslanib, daraxtni shoxlarga ajratadi.

Quyidagi grafikda qarorlar daraxtining umumiy tuzilishi tasvirlangan.



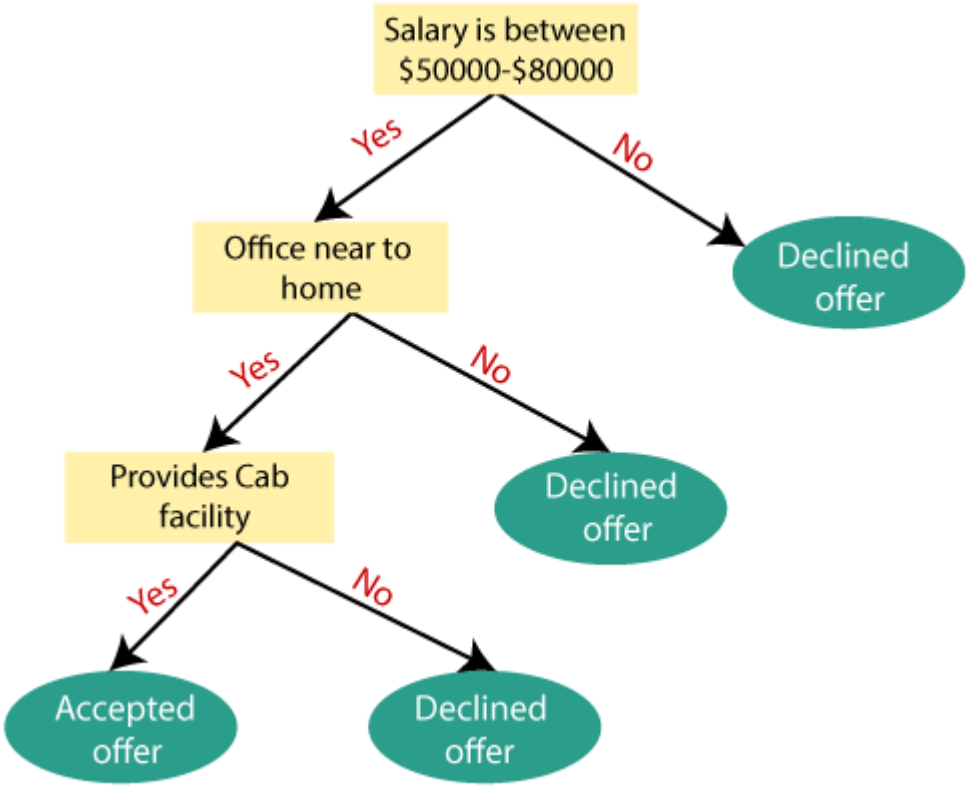
Qarorlar daraxti raqamli ma’lumotlar bilan bir qatorda kategorik ma'lumotlarni (HA/YO‘Q) o‘z ichiga olishi mumkin.

Qarorlar daraxtidan foydalanish sababi

Mashinaviy o‘qitishda turli xil algoritmlar mavjud, shuning uchun berilgan ma’lumotlar to‘plami va muammo uchun eng yaxshi algoritmni tanlash mashinaviy o‘qitish modelini yaratishda eng asosiy maqsad hisoblanadi. Quyida qaror daraxtidan foydalanishning ikkita asosiy sababi bor:

- Qaror daraxtlari odatda qaror qabul qilishda insonning fikrlash qobiliyatini taqlid qiladi, shuning uchun uni tushunish oson;
- Qaror daraxti ortidagi mantiqni osongina tushunish mumkin, chunki u daraxtga o‘xshash tuzilmani ko‘rsatadi.

Qaror daraxtini tushunish uchun quyidagi misol keltiriladi.
Misol: Ish taklifiga ega bo‘lgan va taklifni qabul qilish yoki qabul qilmaslik to‘g‘risida qaror qabul qilmoqchi bo‘lgan nomzod masalasi qaraladi. Shunday qilib, bu muammoni hal qilish uchun qaror daraxti algoritmini qo‘llash mumkin. Ushbu masalada ish haqi, ofisning yaqinligi va sharoitlar asosida daraxt shoxlarga ajraladi. Quyidagi grafikda muammoming hal qilinish jarayoni tasvirlangan.



Yuqoridagi grafikdan shuni anglash mumkinki qarorlar daraxti masalani yechishdagi barcha holatlar bo‘yicha tizimlashtirilgan yechimlar to‘plamini nazorat qilish orqali yagona qarorga olib keladi.

Qaror daraxtining afzalliklari

Qaror daraxtining afzalliklari quyidagilardan iborat:

- Tushunish juda oson, chunki u haqiqiy hayotda har qanday qaror qabul qilishdagi inson amal qiladigan jarayonga amal qiladi;
- Qaror bilan bog‘liq muammolarni hal qilish uchun juda foydali bo‘lishi mumkin;
- Bu muammoning barcha mumkin bo‘lgan natijalari haqida o‘ylashga yordam beradi;
- Boshqa algoritmlarga qaraganda ma’lumotlarni tozalash talablari kamroq bo‘ladi.

Qaror daraxtining kamchiliklari

Qaror daraxtining kamchiliklari quyidagilardan iborat:

- Qaror daraxti juda ko‘p qatlamlarni o‘z ichiga oladi, bu esa uni murakkablashtiradi;

- Tasodifiy o‘rmon algoritmi yordamida hal qilinishi mumkin bo‘lgan ortiqcha muammo paydo bo‘lishi mumkin;
- Ko‘proq sinf muammolari uchun qaror daraxtining hisoblash murakkabligi oshishi mumkin.

Qaror daraxti algoritmini Python dasturlash tilida amalga oshirilishi

Python yordamida qarorlar daraxtini amalga oshirish uchun mijozlarning mashina sotib olishini tavsiya qilish tizimi muammosi bo‘yicha masalani hal qilish qarab o‘tiladi. Buning uchun “**car_data.csv**” ma’lumotlar to‘plamidan foydalaniladi. Agar sizda ma’lumotlar to‘plami mavjud bo‘lmasa ochiq ma’lumotlar to‘plamidan foydalanish mumkin.

Ushbu ma'lumotlar to‘plamida yillik maoshlarini hisobga olgan holda avtomobil sotib olmoqchi bo‘lgan 1000 ta mijoz haqida ma’lumotlar mavjud.

Ma’lumotlar to‘plami quyidagi ustunlardan iborat:

- User IDFoydalanuvchi identifikatori
- Gender – Jinsi
- Age – yosh;
- Annual Salary –Yillik ish haqqi
- Purchase Decision – sotib olish qarori (Yo‘q = 0; Ha = 1)

Ushbu muammo ikki toifali klassifikator (binary classification) yordamida hal qilinadi.

Qaror daraxti algoritmini amalga oshirish bosqichlari

Qaror daraxti algoritmini amalga oshirish bosqichlari quyidagilardan iborat:

- Ma’lumotlarni oldindan qayta ishlash bosqichi, ma’lumotlar to‘plamini tayyorlas;
- O‘quv to‘plamiga qarorlar daraxti algoritmini o‘rnatish;
- Sinov natijasini bashorat qilish;
- Natijaning to‘g‘riligini tekshirish (Confusion matritsasini yaratish);
- Test to‘plamining natijasini vizualizatsiya qilish.

1-qadam. Qaror daraxti algoritmini amalga oshirish dastlab kerakli kutubxonalar chaqirib olinadi.

```
import numpy as np
import pandas as pd
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import
RandomForestClassifier
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import
train_test_split
from sklearn.metrics import
classification_report
from sklearn.metrics import confusion_matrix
from sklearn import tree
from matplotlib import pyplot as plt
from sklearn.model_selection import
cross_val_predict
from sklearn.metrics import
classification_report
import seaborn as sns
import matplotlib.pyplot as plt
```

2-qadam. Ma’lumotlar to‘plami chaqiriladi va ma’lumotlar ekranga chop qilinadi:

```
df=pd.read_csv("car_data.csv")
df.head()
```

	User ID	Gender	Age	AnnualSalary	Purchased
17	372	1	35	53000	0
54	428	0	49	43500	1
5	846	0	47	33500	1
60	917	0	50	109500	1
21	134	0	49	39000	1

3-qadam. Tanlangan ma’lumotlar to‘plamida matnli ma’lumotlar ham bor, Encoder funksiyasi yordamida bu matnli ma’lumotlar raqamli ma’lumotlarga o‘tkaziladi.

```
encoder=LabelEncoder()
```

```
df['Gender']=encoder.fit_transform(df['Gender'].values)
df.sample(5)
```

	User ID	Gender	Age	AnnualSalary	Purchased
730	189	1	44	54500	0
116	375	1	59	143000	1
896	319	1	49	75500	1
385	845	0	55	92500	1
882	451	1	31	108500	1

‘**Purchased’** ustunida mijozga tavsiya etilgan natijasi mashina sotib olishi yoki ololmasligi bo‘yicha ma’lumotlar mavjud. Masalani maqsadi, shu mijozning yillik oyligiga qarab mashina sotib olishi mumkin yoki mumkin emasligi haqida.

4-qadam. *X* o‘zgaruvchiga ‘**Purchased**’ ustunidan boshqa barcha ustundagi ma’lumotlarni, *Y* o‘zgaruvchiga esa ‘**Purchased**’ ustunidagi qiymatlar ta’minlanadi.

```
x = df.drop('Drug‘, axis=1).values
x[0:5]
```

Natija.

```
array([[ 385,    1,   35, 20000],
       [ 681,    1,   40, 43500],
       [ 353,    1,   49, 74000]])
y = df['Purchased'].values
```

5-qadam. Ma’lumotlarni test va o‘quv to‘plamlariga ajratiladi va **‘y_predict’** o‘zgaruvchisiga modelning bashorat natijasi ta’minlanadi.

```
x_train,x_test,y_train,y_test=train_test_split(x,y, test_size=0.4, random_state=20)

tree_model=DecisionTreeClassifier()

tree_model.fit(x_train,y_train)
```

```
DecisionTreeClassifier()

y_predict=tree_model.predict(x_test)
```

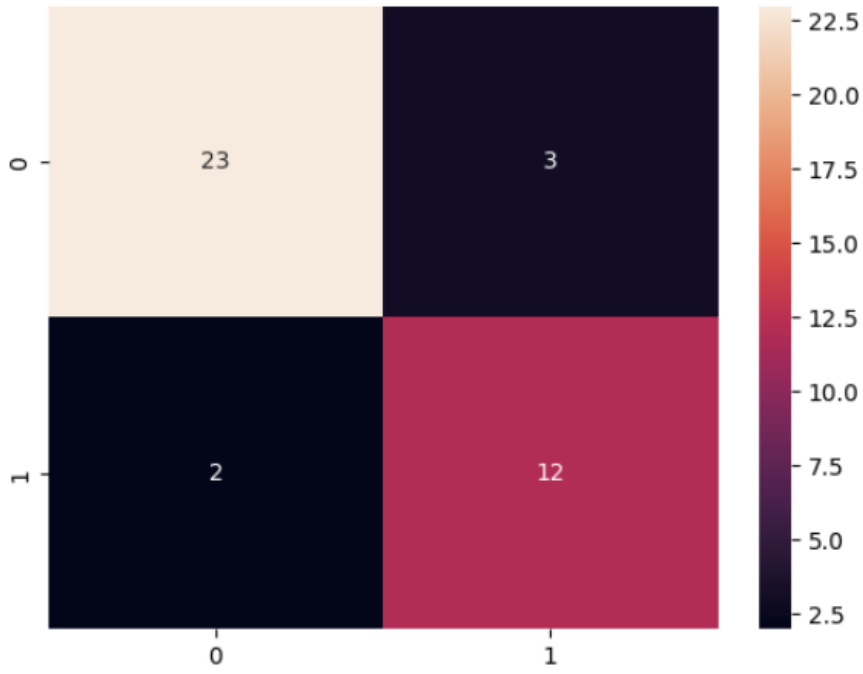
6-qadam. Bashorat natijasi olingandan so‘ng, modelni Precision, Recall, F-1 Score yordamida baholab olinadi.

```
print(classification_report(y_test, y_predict))
```

	precision	recall	f1-score	support
0	0.92	0.88	0.90	26
1	0.80	0.86	0.83	14
accuracy			0.88	40
macro avg	0.86	0.87	0.86	40
weighted avg	0.88	0.88	0.88	40

7-qadam. Confusion matritsasini yaratish. Endi model uchun klassifikatorning aniqlilik ko‘rsatkichini aniqlash maqsadida confusion matritsasi yaratiladi.

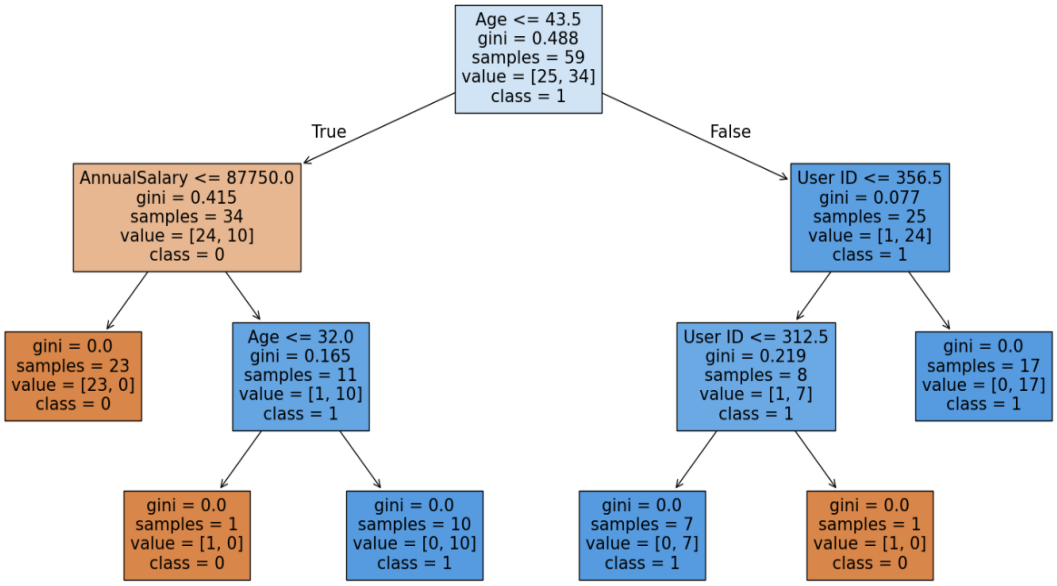
```
sns.heatmap(confusion_matrix(y_test, y_predict),
annot= True)
plt.show()
```



8-qadam. Qaror darxtining algoritmini quyidagi dastur kodi orqali grafik ko‘rinishi chop qilinadi.

```
cols=df.drop('Drug‘, axis=1).columns
classes=df['Drug‘].unique()
plt.figure(figsize=(30,20))
tree.plot_tree(tree_model,feature_names=cols,
class_names=classes, filled=True)
plt.show()
```

Qaror daraxtida o‘quv to‘plamining natijasini tasavvur qilish maqsadida yuqoridagi dastur kodi orqali qarorlar daraxti tasniflagichi uchun grafik chiziladi.



Endi yuqoridagi ma’lumotlar to‘plamiga mos ixtiyoriy bitta yozuvli ma’lumotlar x_test o‘rniga o‘qitilsa mashina sotib olishi mumkin yoki mumkin emasligini aniqlash mumkin bo‘ladi.

Nazariy savollar

1. Qaror daraxti (Decision Tree) algoritmi va uning imkoniyatlarini tushuntiring?
2. Qarorlar daraxtidan foydalanish sababini tushuntiring?
3. Qaror daraxtining afzalliklarini tushuntiring?
4. Qaror daraxtining kamchiliklarini tushuntiring?

IV-BOB. MASHINAVIY O‘QITISHDA KLASTERIZATSIYA ALGORITMLARI VA SUN’IY NEYRON TARMOQLARI

4.1. Mashinaviy o‘qitishda klasterlash texnologiyasi va algoritmlari

Reja:

Nazoratsiz o‘qitish;
Klasterlash;
Klasterlash usullarining turlari;
Guruhlash yordamida klasterlash;
Zichlikka asoslangan klasterlash;
Tarqatish modeliga asoslangan klasterlash;
Ierarxik klasterlash;
Noaniq klasterlash;
Klasterlash algoritmlari;
Klasterlash texnologiyasini sohalarda qo‘llanilishi.

Nazoratsiz o‘qitish

Klasterlash jarayoni tartiblanmagan ma’lumotlarni ma’lum bir guruhlarga ajratishga xizmat qiladi. Regressiya masalasida ma’lumotlarga tayanib keyingi holatlarni nazoratli o‘qitish usullari orqali aniqlash imkoni mavjud. Nazoratsiz o‘qitish jarayonida ma’lumotlarga aniq ko‘rsatmasiz holda masala uchun qaror qabul qilish mumkin bo‘ladi.

Nazoratsiz o‘qitish - tasniflanmagan va etiketlanmagan ma’lumotlardan foydalangan holda algoritmga ushbu ma’lumotlarga ko‘rsatmalarsiz harakat qilish imkonini beradigan mashinaviy o‘qitish usuli hisoblanadi. Bu yerda mashinaning vazifasi saralanmagan ma’lumotlarni o‘xshashliklari, naqshlari va farqlari bo‘yicha ma’lumotlarni oldindan tayyorlamasdan guruhlashdan iborat.

Nazoratsiz o‘qitish nazorat ostidagi o‘qitishdan farqli o‘laroq, hech qanday o‘qituvchi ya’ni o‘quv tanlanma taqdim etilmaydi. Shuning uchun mashinaga yorliqsiz ma’lumotlardagi yashirin tuzilmani o‘zi topishi belgilangan.

Nazoratsiz o‘qitish algoritmi ikki toifaga bo‘linadi:

- **Klasterlash;**
- **Assotsiatsiya.**

Klasterlash yoki **klaster tahlili** - bu yorliqsiz ma’lumotlar to‘plamini guruhlaydigan mashinaviy o‘qitish usuli hisoblanadi. Bunda ma'lumotlar nuqtalarini o‘xshash ma'lumotlar nuqtalaridan tashkil topgan turli klasterlarga guruhlash jarayoni bajariladi. Natijada mumkin bo‘lgan o‘xshashliklarga ega bo‘lgan ob'ektlar boshqa guruh bilan kamroq yoki umuman o‘xshashliklarga ega bo‘lmagan guruhda qoladi deb ta'riflanishi mumkin.

Klasterlash

Klasterlash jarayoni nazoratsiz o‘qitish usuli hisoblanadi, shuning uchun algoritmga hech qanday nazorat berilmaydi va u etiketlanmagan ma’lumotlar to‘plami bilan shug‘ullanadi.

Ushbu klasterlash texnologiyasi qo‘llanilgandan so‘ng, har bir klaster yoki guruh klaster-identifikatori bilan ta’minlanadi. Mashinaviy o‘qitish tizimi katta va murakkab ma’lumotlar to‘plamlarini qayta ishlashni soddalashtirish uchun ushbu identifikatordan foydalanishi mumkin.

Klasterlash usuli odatda **statistik ma’lumotlarni tahlil qilish uchun ishlatiladi.**

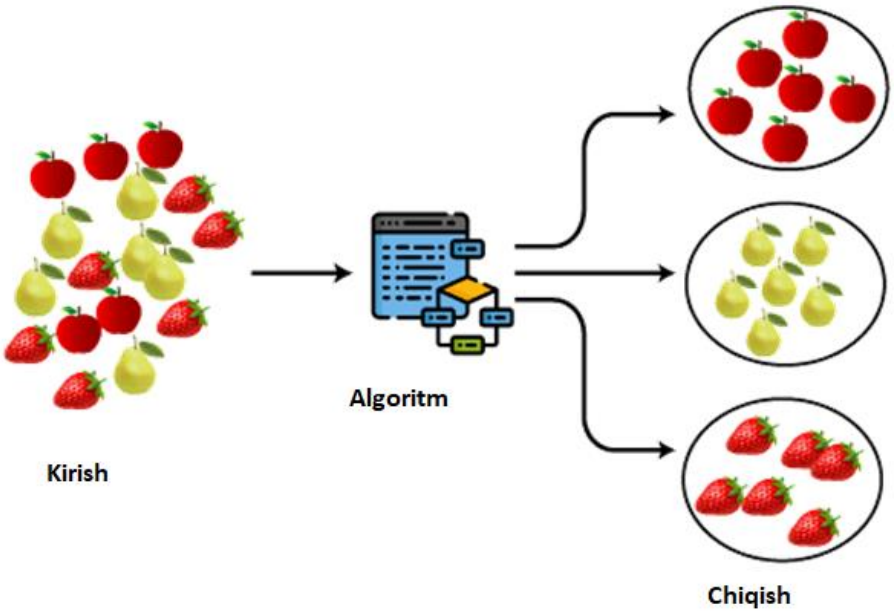
Klasterlash tasniflash algoritmiga o‘xshaydi, ammo uning farqi foydalanayotgan ma’lumotlar to‘plamining turi boshqa ekanligi bilan izohlanadi. Tasniflashda etiketli ma’lumotlar to‘plami bilan ishlanadi, klasterlashda esa yorliqsiz ma’lumotlar to‘plami bilan ishlanadi.

Klasterlash texnologiyasi turli vazifalarni bajarishda keng qo‘llanilishi mumkin:

- Bozor segmentatsiyasida;
- Statistik ma’lumotlarni tahlil qilishda;
- Ijtimoiy tarmoqlarni tahlil qilishda;
- Tasvir segmentatsiyasida;
- Anomaliyalarni aniqlash va boshqalar.

Quyidagi grafik orqali klasterlash algoritmining ishlashini tushunish mumkin. Turli xil mevalar to‘plami berilgan bo‘lsa, ularni

klasterlash o‘xshash xususiyatlarga ega bo‘lgan bir nechta guruhlarga ajratish mumkin.



Klasterlash algoritmi yorliqsiz ma’lumotlar ichidan o‘xshashliklarni hisobga olgan holda guruhlarga ajratishni amalga oshiradi.

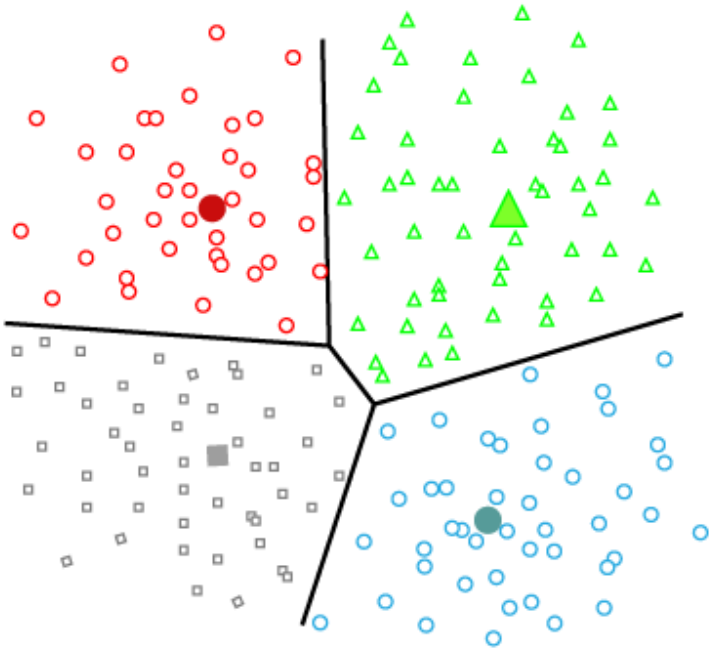
Klasterlash usullarining turlari

Klasterlash usullari keng miqyosda **qattiq klasterlash** (ma’lumotlar nuqtasi faqat bitta guruhga tegishli) va **yumshoq klasterlash** (ma’lumotlar nuqtalari boshqa guruhga ham tegishli bo‘lishi mumkin) ga bo‘linadi. Ammo klasterlashning boshqa turli xil yondashuvlari ham mavjud. Mashinaviy o‘qitishda qo‘llaniladigan asosiy klasterlash usullari quyida keltirib o‘tilgan:

- Guruhlash yordamida klasterlash;
- Zichlikka asoslangan klasterlash;
- Tarqatish modeliga asoslangan klasterlash;
- Ierarxik klasterlash;
- Noaniq klasterlash;

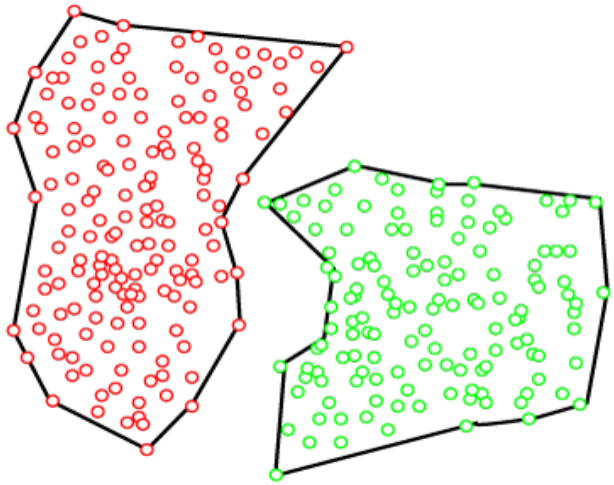
Guruhlash yordamida klasterlash

Ma’lumotlarni ierarxik bo‘lmagan guruhlarga ajratadigan klasterlashning turi **centroid(klasterlar markazi)ga asoslangan usul** sifatida ham tanilgan. Klasterlarni bo‘lishning eng keng tarqalgan misoli K-Means Clustering algoritmi hisoblanadi. Bunda ma’lumotlar quyidagi rasm ko‘rinishidagi guruhlarga ajratiladi.



Zichlikka asoslangan klasterlash

Zichlikka asoslangan klasterlash usuli yuqori zichlikga ega hududlarni klasterlarga bog‘laydi. Ushbu algoritm turi ma'lumotlar to‘plamidagi turli klasterlarni aniqlash orqali amalga oshiradi. Ma'lumotlar maydonidagi zich joylar bir-biridan siyrakroq joylarga bo‘linadi. Agar ma’lumotlar to‘plami turli xil zichlik va yuqori o‘lchamlarga ega bo‘lsa, ushbu algoritmlar ma’lumotlar nuqtalarini klasterlashda qiyinchiliklarga duch kelishi mumkin. Bunda ma’lumotlar quyidagi rasm ko‘rinishidagi klasterlanadi.

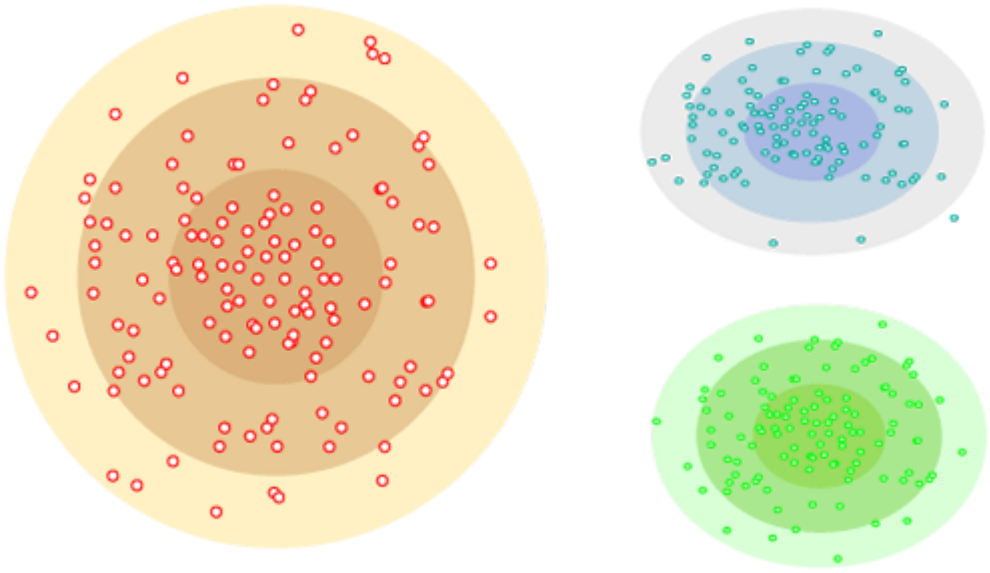


Tarqatish modeliga asoslangan klasterlash

Tarqatish modeliga asoslangan klasterlash usulida ma’lumotlar, ma’lumotlar to‘plamining ma’lum bir taqsimotiga tegishli bo‘lish

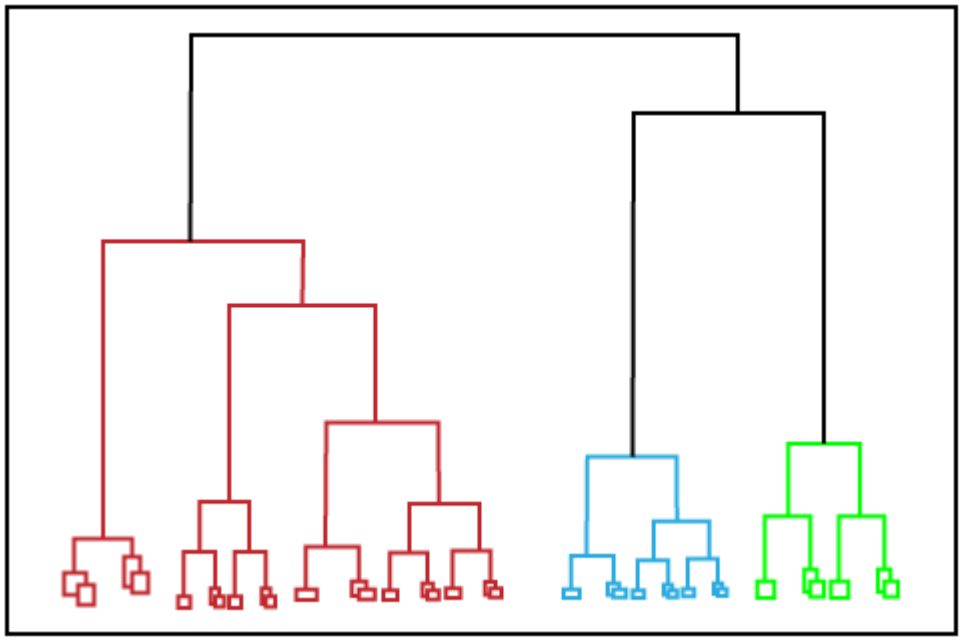
ehtimoli asosida bo‘linadi. Guruhlash ba’zi taqsimotlarni umumiy **Gauss taqsimoti** asosida qabul qilish orqali amalga oshiriladi.

Ushbu turga misol Gauss aralashmasi modellaridan foydalanadigan **Kutish-Maksimizatsiya klasterlash algoritmi** hisoblanadi. Bunda ma’lumotlar quyidagi rasm ko‘rinishidagi klasterlanadi.



Ierarxik klasterlash

Ma’lumotlarni qayta ishlashda ierarxik klasterlashdan bo‘lingan klasterlarga muqobil sifatida foydalanish mumkin, chunki yaratiladigan klasterlar sonini oldindan belgilash talabi yo‘q. Ushbu texnologiyada ma’lumotlar to‘plami daraxtga o‘xshash tuzilmani yaratish uchun klasterlarga bo‘linadi, bu dendrogramma deb ham ataladi . Kuzatishlar yoki har qanday miqdordagi klasterlar daraxtni to‘g‘ri darajada kesish orqali tanlanishi mumkin. Ushbu usulning eng keng tarqalgan misoli **Aglomerativ ierarxik algoritm** hisoblanadi. Bunda ma’lumotlar quyidagi rasm ko‘rinishidagi klasterlanadi.



Noaniq klasterlash

Noaniq klasterlash - bu ma'lumotlar ob’ekti bir nechta guruh yoki klasterga tegishli bo‘lishi mumkin bo‘lgan usulning bir turi hisoblanadi. Har bir ma’lumotlar to‘plamida klasterga a’zolik darajasiga bog‘liq bo‘lgan a’zolik koeffitsientlari to‘plami mavjud. Bu turdagi klasterlashning misoli **Fuzzy C-means algoritmi** hisoblanadi.

Klasterlash algoritmlari

Klasterlash algoritmlarini yuqorida tavsiflangan modellari asosida ajratish mumkin. Klasterlash algoritmlarining har xil turlari ishlab chiqilgan, ammo ulardan faqat bir nechtasi keng tarqalgan. Klasterlash algoritmi biz foydalanadigan ma’lumotlar turiga asoslanadi. Masalan, ba’zi algoritmlar berilgan ma’lumotlar to‘plamidagi klasterlar sonini taxmin qilishlari kerak, ba’zilari esa ma’lumotlar to‘plamini kuzatish orasidagi minimal masofani topish uchun talab qilinadi.

Mashinaviy o‘qitishda hozirda keng qo‘llaniladigan mashhur klasterlash algoritmlaridan quyidagilarni keltirish mumkin:

- K-means algoritmi;
- Mean-shift algoritmi;
- DBSCAN algoritmi;
- Aglomerativ ierarxik algoritm;
- Affinity Propagation algoritm.

Klasterlash texnologiyasini sohalarda qo‘llanilishi

Quyida mashinaviy o‘qitishda klasterlash texnologiyasining ba’zi keng tarqalgan ilovalari keltirilgan:

Saraton hujayralarini aniqlashda: Klasterlash algoritmlari saraton hujayralarini aniqlash uchun keng qo‘llaniladi. U saraton va saraton bo‘lmagan ma'lumotlar to‘plamini turli guruhlarga ajratadi.

Qidiruv mexanizmlarida: Qidiruv mexanizmlari klasterlash texnologiyasi ustida ham ishlaydi. Qidiruv natijasi qidiruv so‘roviga eng yaqin ob’ekt asosida paydo bo‘ladi. Bu o‘xshash ma’lumotlar ob’ektlarini boshqa o‘xshash bo‘lmagan ob’ektlardan uzoqda joylashgan bir guruhda guruhlash orqali amalga oshiradi. So‘rovning aniq natijasi ishlatiladigan klasterlash algoritmining sifatiga bog‘liq bo‘ladi.

Mijozlarni segmentatsiyalash: Mijozlarni segmentatsiyalashda u bozor tadqiqotida iste'molchilarni tanlovi va afzalliklariga qarab segmentlash uchun ishlatiladi.

Biologiyada: Biologiya oqimida tasvirni aniqlash texnologiyasidan foydalangan holda o‘simliklar va hayvonlarning turli turlarini tasniflash uchun ham foydalaniladi.

Nazariy savollar

1. Nazoratsiz o‘qitish va uning vazifasini tushuntiring?
2. Klasterlash va uning vazifasini tushuntiring?
3. Klasterlash usullarining turlari va ularning vazifasini tushuntiring?
4. Guruhlash yordamida klasterlash jarayonini tushuntiring?
5. Zichlikka asoslangan klasterlash jarayonini tushuntiring?
6. Tarqatish modeliga asoslangan klasterlash jarayonini tushuntiring?
7. Ierarxik klasterlash jarayonini tushuntiring?
8. Noaniq klasterlash jarayonini tushuntiring?
9. Klasterlash algoritmlari va ularni vazifasini tushuntiring?
10. Klasterlash texnologiyasini sohalarda qo‘llanilishini tushuntiring?

4.2. K-means klasterlash algoritmi

Reja:

K-Means algoritmi;

K-Means algoritmining bosqichlari;

K-means klasterlashda "K klasterlar soni" qiymatini tanlash;

Python dasturlash tilida K-means klasterlash algoritmini amalga oshirish.

K-means klasterlash (K-Means Clustering) - bu nazoratsiz o‘qitish algoritmi bo‘lib, u mashinaviy o‘qitish yoki ma’lumotlar tahlili fanida klasterlash muammolarini hal qilish uchun ishlatiladi. Ushbu mavzuda K-means klasterlash algoritmi nima ekanligini, algoritm qanday ishlashini, shuningdek Python dasturlash tilida k-means klasterini amalga oshirish tushuntirib o‘tilgan.

K-Means algoritmi

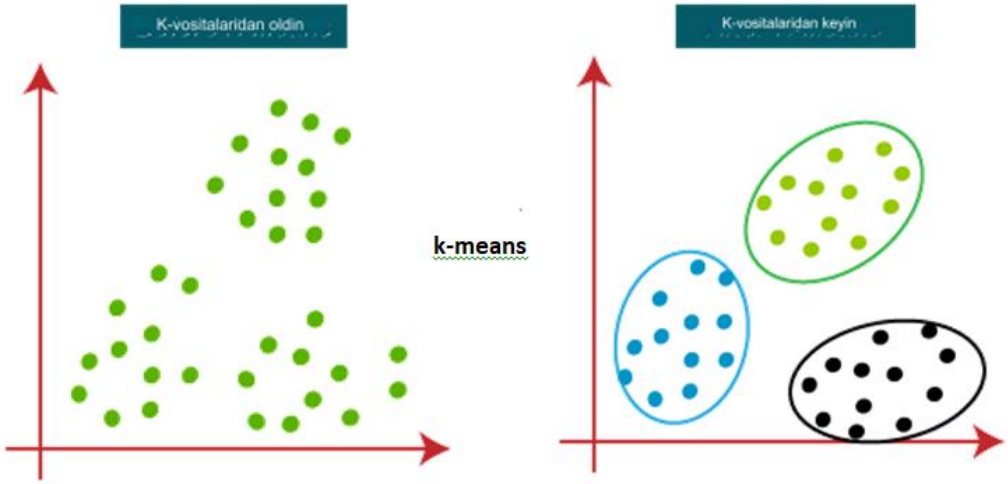
K-means klasterlash (K-Means Clustering) - bu nazoratsiz o‘qitish algoritmi bo‘lib, yorliqsiz ma’lumotlar to‘plamini turli klasterlarga guruhlaydi. Quyida K jarayonda yaratilishi kerak bo‘lgan oldindan belgilangan klasterlar sonini belgilaydi, bunda K=2 bo‘lsa, ikkita klaster bo‘ladi, K=3 uchun esa uchta klaster bo‘ladi va hokazo.

Bu iterativ algoritm bo‘lib, etiketlanmagan ma’lumotlar to‘plamini har bir ma’lumot to‘plami uchun o‘xshash xususiyatlarga ega bo‘lgan faqat bitta guruhga tegishli bo‘lgan turli xil klasterlarga ajratadi. Bu bizga ma’lumotlarni turli guruhlarga birlashtirish imkonini beradi va hech qanday o‘qitishga ehtiyoj sezmasdan, yorliqsiz ma’lumotlar to‘plamidagi guruhlar toifalarini mustaqil ravishda topish imkonini beradi. Bu centroidga asoslangan algoritm bo‘lib, har bir klaster markaz bilan bog‘langan bo‘ladi. **Ushbu algoritmning asosiy maqsadi** ma’lumotlar nuqtasi va ularning tegishli klasterlari orasidagi masofalar yig‘indisini **minimallashtirishdan** iborat. Algoritm yorliqsiz ma’lumotlar to‘plamini kirish sifatida oladi, ma’lumotlar to‘plamini k-sonli klasterlarga ajratadi va eng yaxshi klasterlarni topmaguncha jarayonni takrorlaydi. Ushbu algoritmda k ning qiymati oldindan belgilanishi kerak.

K-means klasterlash algoritmi asosan **ikkita vazifani** bajaradi:

- Iterativ jarayon orqali K markaz nuqtalari yoki markazlar uchun eng yaxshi qiymatni aniqlaydi;
- Har bir ma’lumot nuqtasini eng yaqin k-markaziga tayinlaydi. Muayyan k-markazga yaqin bo‘lgan ma’lumotlar nuqtalariga klaster yaratadi.

Demak, har bir klasterda ba’zi umumiyliklarga ega bo‘lgan ma’lumotlar nuqtalari mavjud va u boshqa klasterlardan uzoqda bo‘ladi. Quyidagi grafikda K-means klasterlash algoritmining ishlashi tushuntirilgan:



K-Means algoritmining bosqichlari

K-Means algoritmining ishlashi quyidagi bosqichlarda amalga oshiriladi.

1-qadam: Klasterlar sonini aniqlash uchun K raqami tanlanadi.

2-qadam: Tasodifiy K nuqtalari yoki markazlari tanlanadi. (Kirish ma’lumotlar to‘plamidan boshqa bo‘lishi mumkin).

3-qadam: Har bir ma’lumot nuqtasi oldindan belgilangan K klasterlarni tashkil etadigan eng yaqin markazga tayinlanadi.

4-qadam: Dispersiya hisoblanadi va har bir klasterning yangi centroidiga joylashtiriladi.

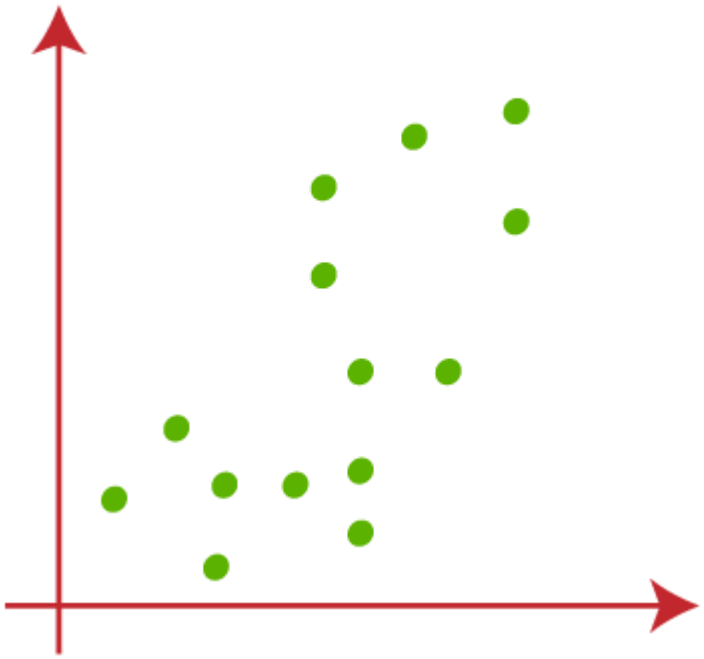
5-qadam: Uchinchi qadam takrorlanadi, ya’ni har bir ma’lumot nuqtasini har bir klasterning yangi eng yaqin klaster markaziga qayta tayinlaydi.

6-qadam: Agar biron-bir qayta tayinlash sodir bo‘lsa, 4-bosqichga o‘tiladi, aks holda tamomga o‘tiladi.

7-qadam : Model tayyor, tamom.

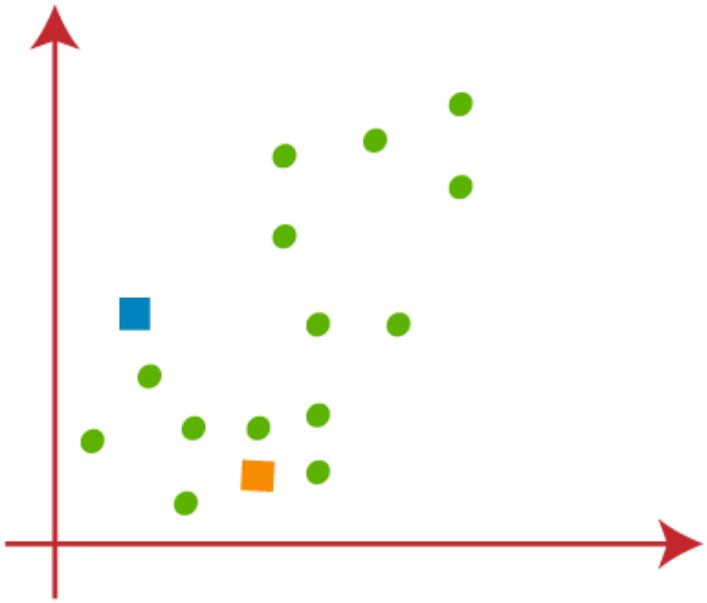
Quyidagi vizual syujetlarni ko‘rib chiqish orqali yuqoridagi bosqichlarni tushunish mumkin:

Faraz qilaylik, ikkita M1 va M2 o‘zgaruvchilar bor. Bu ikki o‘zgaruvchining **x** va **y** o‘qining tarqalish grafigi quyidagi grafikda keltirilgan:

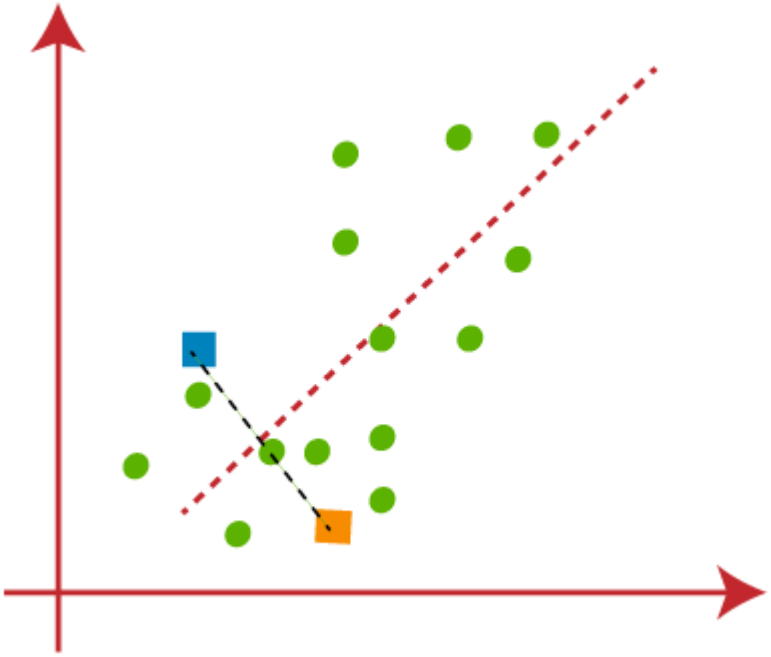


Ma’lumotlar to‘plamini aniqlash va ularni turli klasterlarga joylashtirish uchun klasterlarning k sonini, ya’ni K=2 ga teng deb qabul qilinsa. Bu shuni anglatadiki, bu erda ushbu ma’lumotlar to‘plamini ikki xil klasterga guruhlashga harakat qilinadi.

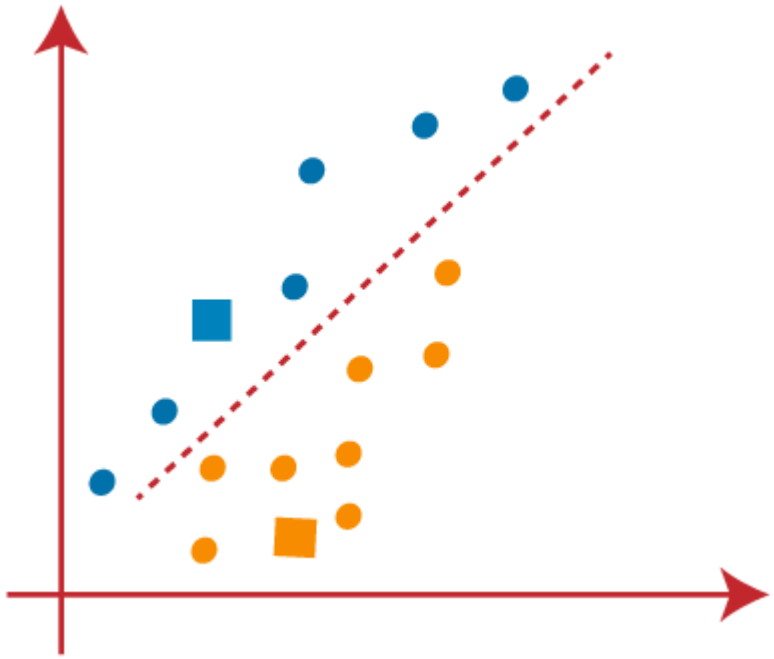
Klasterni shakllantirish uchun tasodifiy k nuqta yoki markazni tanlash kerak. Bu nuqtalar ma’lumotlar to‘plamidagi nuqtalar yoki boshqa nuqtalar bo‘lishi mumkin. Shunday qilib, biz quyida keltirilgan ikkita nuqtani k nuqta sifatida tanlaymiz, ular bizning ma’lumotlar to‘plamimizning bir qismi emas va u quyidagi rasmda ifodalangan.



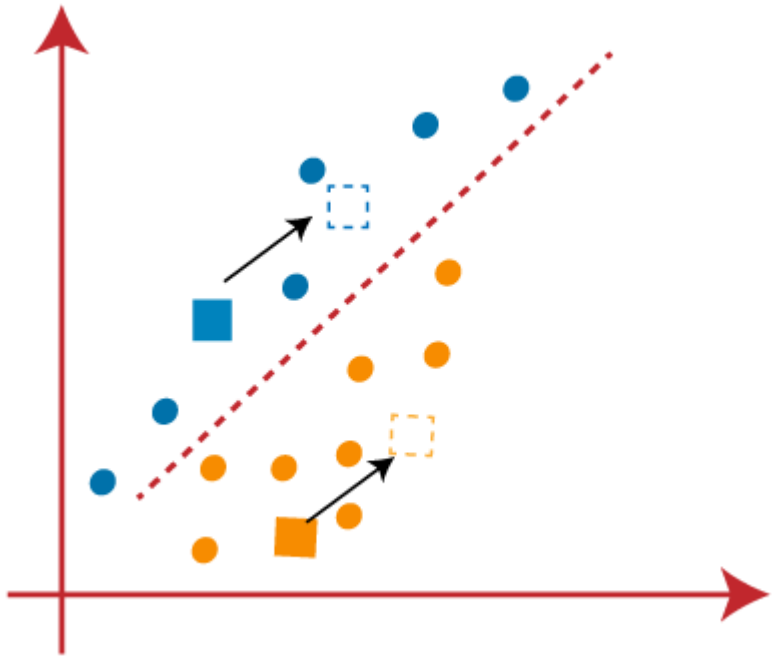
Endi biz tarqalish grafigining har bir ma’lumot nuqtasini, unga eng yaqin K-nuqtasi yoki klaster markaziga tayinlaymiz. Ikki nuqta orasidagi masofani hisoblanadi. Shunday qilib, biz ikkala markaz o‘rtasida mediana chizig‘ini o‘tkazamiz.



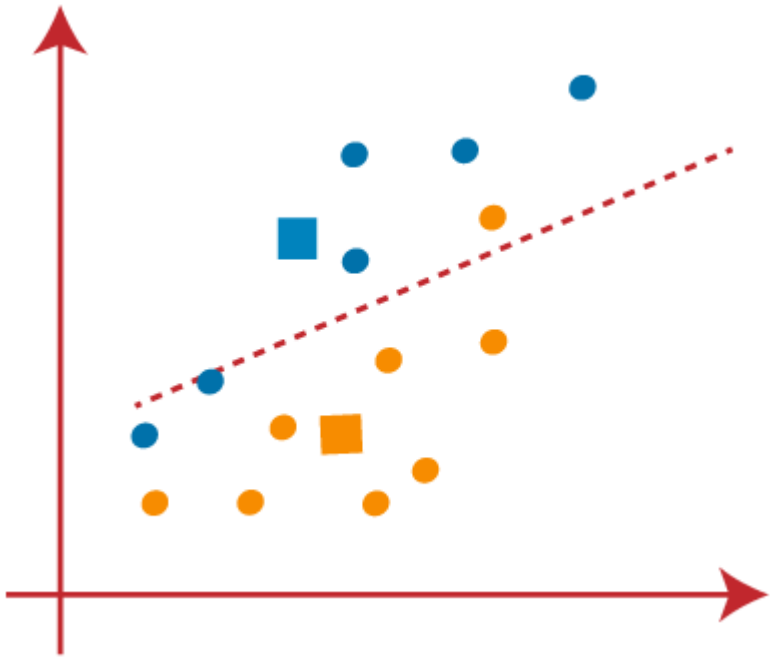
Yuqoridagi rasmdan ko‘rinib turibdiki, chiziqning chap tomonidagi nuqtalar K1 yoki ko‘k markazga yaqin, chiziqning o‘ng tomonidagi nuqtalar esa sariq markazga yaqindir. Bu jarayonni aniq ko‘rish uchun ularni ko‘k va sariq rangga bo‘yab ko‘rsatilgan.



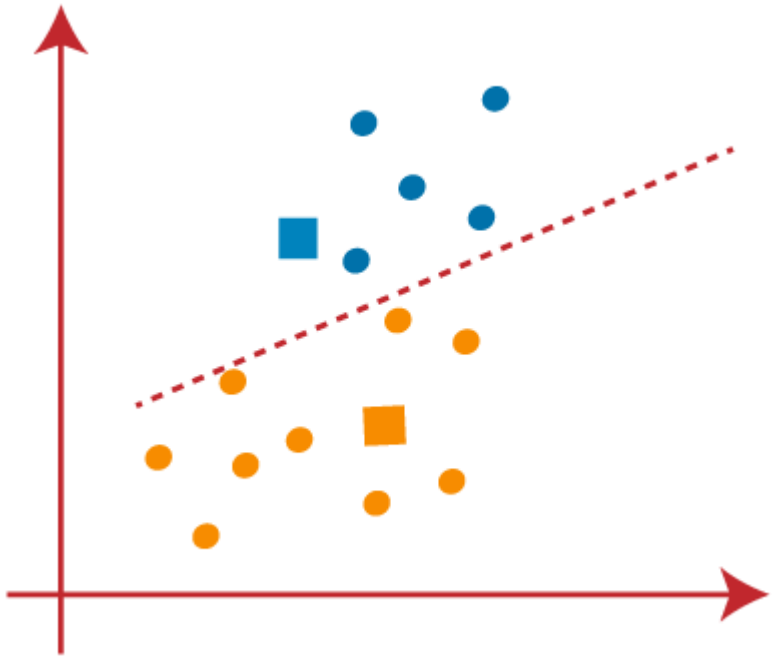
Biz eng yaqin klasterni topishimiz kerak bo‘lgani uchun, biz yangi klaster markazini tanlash orqali jarayonni takrorlaymiz. Yangi markazlarni tanlash uchun biz ushbu markazlarning og‘irlik markazini hisoblaymiz va quyidagi yangi markazlarni topamiz.



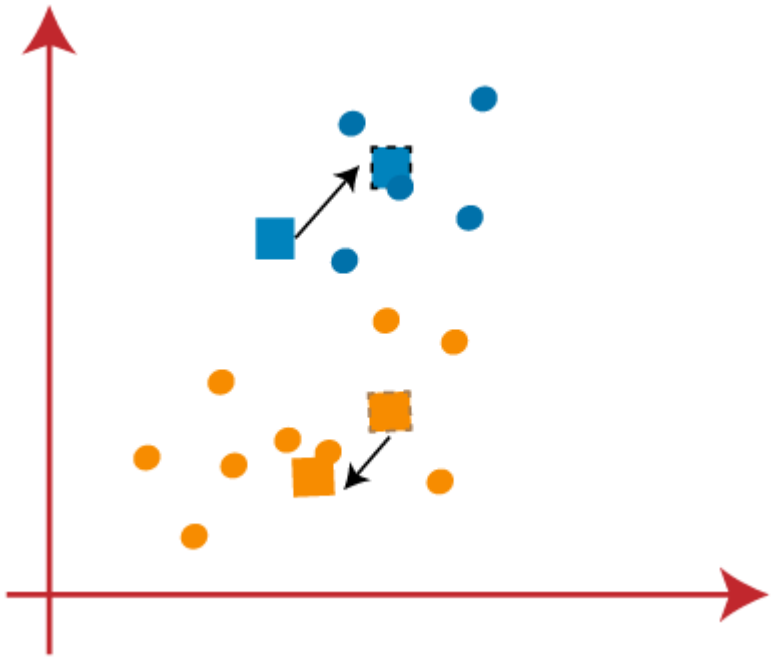
Keyinchalik, har bir ma’lumot nuqtasini yangi markazga qayta tayinlanadi. Buning uchun biz median chiziqni topishning bir xil jarayonini takrorlaymiz. Median quyidagi rasmga o‘xshash ko‘rinishda bo‘ladi:



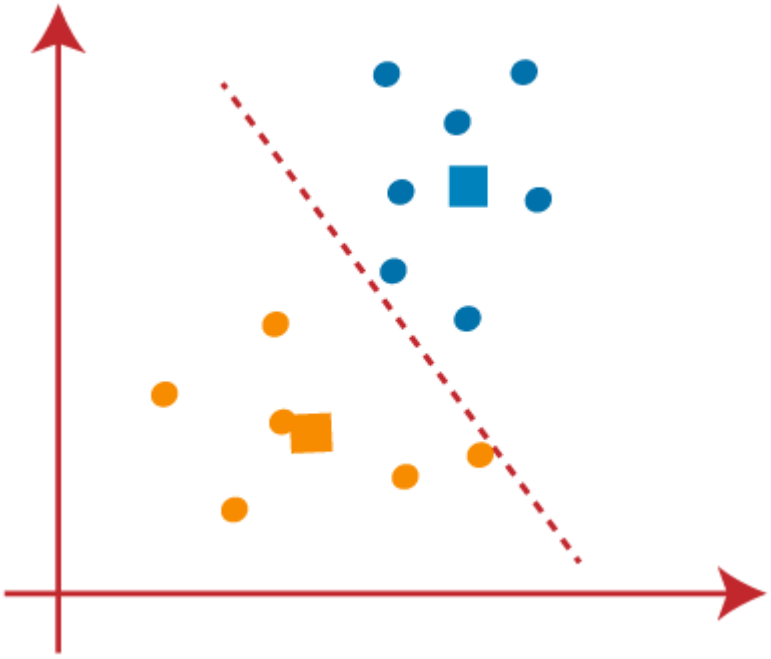
Yuqoridagi rasmdan biz bitta sariq nuqta chiziqning chap tomonida va ikkita ko‘k nuqta chiziqqa to‘g‘ri kelishini ko‘rishimiz mumkin. Shunday qilib, bu uch nuqta yangi markazlarga beriladi.



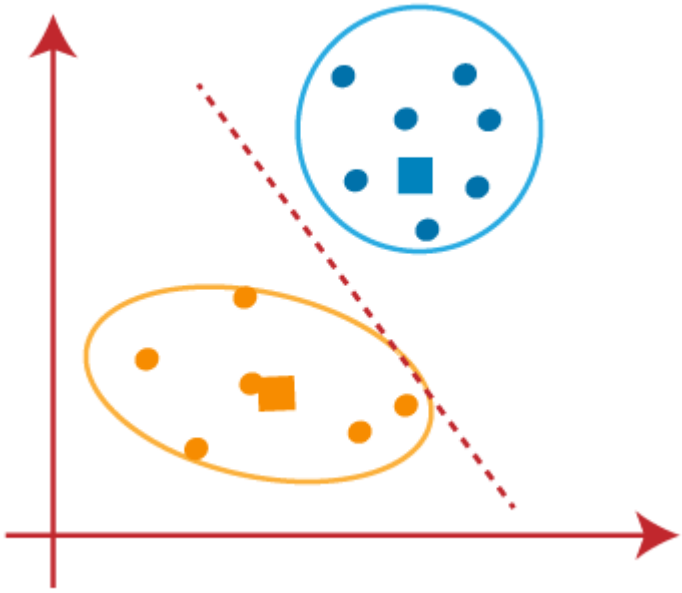
Qayta tayinlash sodir bo‘lganligi sababli, biz yana 4-bosqichga o‘tamiz, ya’ni yangi markazlar yoki K nuqtalarini topamiz. Biz markazning og‘irlik markazini topish orqali jarayonni takrorlaymiz, shuning uchun yangi markazlar quyidagi rasmda ko‘rsatilgandek bo‘ladi:



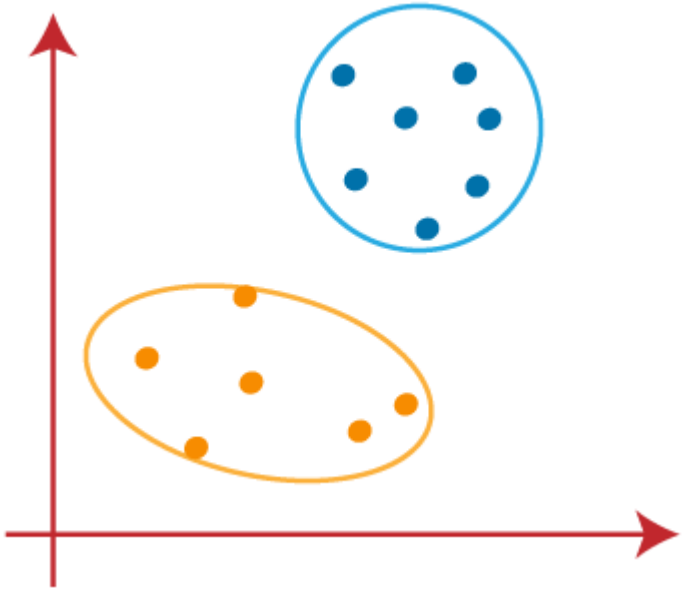
Yangi klaster markazlarini olganimizdek, yana median chiziq chiziladi va ma’lumotlar nuqtalari qayta tayinlanadi. Shunday qilib, rasm quyidagicha bo‘ladi.



Yuqoridagi rasmda ko‘rishimiz mumkin, chiziqning har ikki tomonida o‘xshash bo‘lmagan ma’lumotlar nuqtalari mavjud emas, bu bizning modelimiz shakllanganligini anglatadi. Quyidagi rasmga e’tibor bering.



Bizning model tayyor bo‘lgani uchun, biz endi taxmin qilingan klaster markazini olib tashlashimiz mumkin va ikkita oxirgi klaster quyidagi rasmda ko‘rsatilganidek bo‘ladi.



K-means klasterlashda “K klasterlar soni” qiymatini tanlash

K-means klasterlash algoritmining ishlashi u tashkil etuvchi yuqori samarali klasterlarga bog‘liq bo‘ladi. Ammo klasterlarning optimal sonini tanlash katta vazifa hisoblanadi. Klasterlarning optimal sonini topishning turli xil usullari mavjud, ammo bu yerda biz klasterlar sonini yoki K qiymatini topishning eng mos usulinitopish tushuntiriladi.

Tirsak usuli

Tirsak usuli klasterlarning optimal sonini topishning eng mashhur usullaridan biridir. Bu usul WCSS(Within Cluster Sum of Squares) qiymati tushunchasidan foydalanadi. **WCSS** klaster ichidagi jami o‘zgarishlarni belgilaydigan **kvadratlar yig‘indisi ichida** degan ma’noni anglatadi. WCSS qiymatini hisoblash formulasi (2 ta klaster uchun) quyida keltirilgan.

$$\cdot WCSS = \sum P_{i\ in\ Cluster1} distance(P_i C_1)^2 + \sum P_{i\ in\ Cluster2} distance(P_i C_2)^2$$

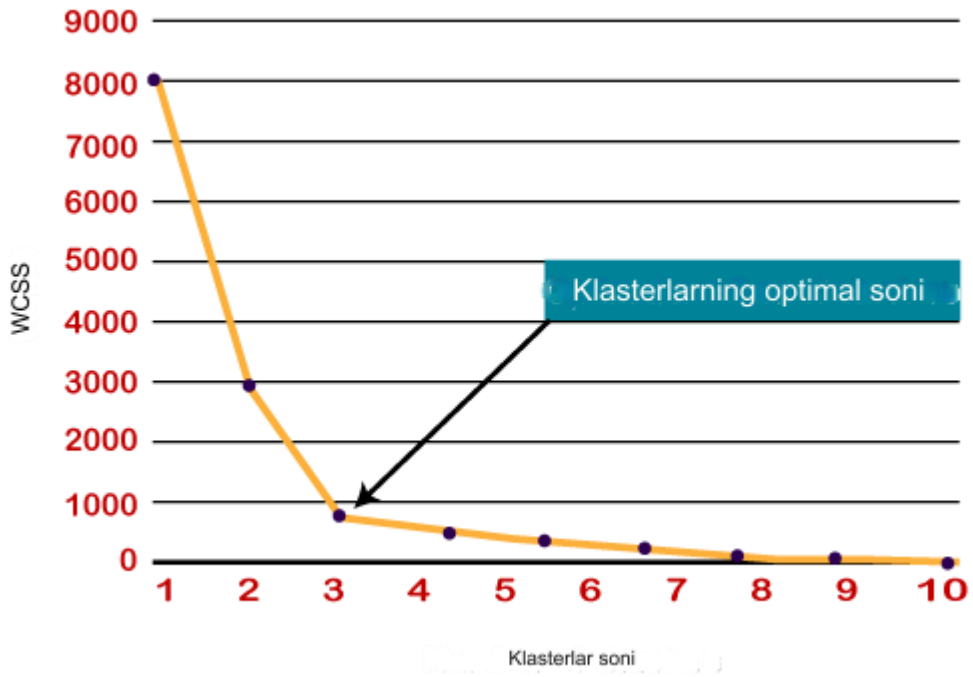
Yuqoridagi WCSS formulasidagi $\sum P_{i\ in\ Cluster1} distance(P_i C_1)^2$ bu klaster ichidagi har bir ma’lumot nuqtasi va uning klaster markazi orasidagi masofalar kvadratining yig‘indisi va qolgan hadlar uchun bir xil hisoblanadi.

Ma’lumotlar nuqtalari va markaz orasidagi masofani o‘lchash uchun Evklid masofasi yoki Manxetten masofasi kabi har qanday usuldan foydalanish mumkin.

Klasterlarning optimal qiymatini topish uchun **tirsak usuli** quyidagi bosqichlarda bajaradi:

- Turli K qiymatlari (1-10 oralig‘ida) uchun berilgan ma’lumotlar to‘plamida K-vositalarni klasterlashni amalga oshiradi;
- K ning har bir qiymati uchun WCSS qiymatini hisoblab chiqadi;
- Hisoblangan WCSS qiymatlari va K klasterlar soni o‘rtasida egri chiziq chizadi;
- Keskin egilish nuqtasi yoki sohaning bir nuqtasi qo‘lga o‘xshaydi, keyin bu nuqta K ning eng yaxshi qiymati deb hisoblanadi.

Grafikda tirsakka o‘xshash o‘tkir egilish ko‘rsatilganligi sababli, u tirsak usuli sifatida tanilgan. Tirsak usuli uchun grafik quyidagi rasmga o‘xshash ifodalanadi:



Biz berilgan ma’lumotlar nuqtalariga teng klasterlar sonini tanlashimiz mumkin. Agar biz ma'lumotlar nuqtalariga teng bo‘lgan klasterlar sonini tanlasak, WCSS qiymati nolga aylanadi va bu chizmaning so‘nggi nuqtasi bo‘ladi.

Python dasturlash tilida K-means klasterlash algoritmini amalga oshirish

Yuqoridagi bo‘limda K-means algoritmi haqida tushunchalar berildi, endi uni Python yordamida qanday amalga oshirish mumkinligi keltirib o‘tiladi.

Ushbu amaliyotda biz tasodifiy nuqtalar yaratamiz va bu nuqtalarni klasterlaymiz. Maqsad: klasterlash algoritmi qanday ishlashini ko‘rish:

Python dasturlash tilida K-means klasterlash algoritmini amalga oshirish uchun dastlab, birinchi navbatda, ma’lumotlarni oldindan qayta ishlashning bir qismi bo‘lgan model uchun quyidagi kutubxonalarni import qilish kerak bo‘ladi.

```
import random
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from sklearn.datasets import make_blobs
%matplotlib inline
```

Tasodifiy nuqtalar klasterini yaratish uchun **make_blobs** funksiyasidan foydalaniladi. Bu funksiya quyidagi parametrlarni qabul qiladi.

n_samples - nuqtalar soni.

centers - klasterlar markazi (sentroid) koordinatalari.

cluster_std - markazdan standart og‘ish.

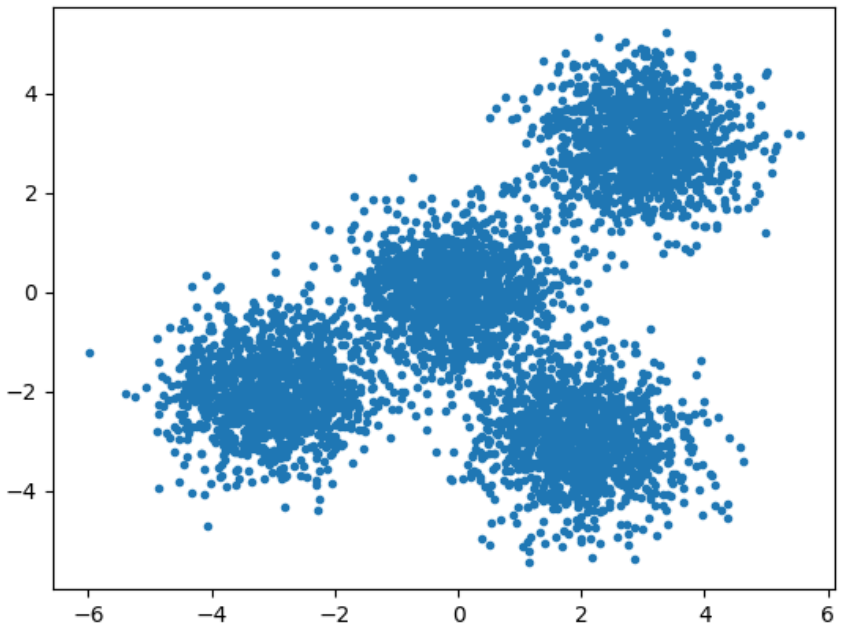
make_blobs funksiyasi nuqtalarning x va y koordinatalarini qaytaradi.

Tasodifiy nuqtalar klasterini yaratish uchun quyidagi dastur kodi tavsiya etiladi. Bunda dastlab klaster markazlari qo‘lda beriladi.

```
np.random.seed(0)
centroids=[[3,3],[-3,-2],[2,-3],[0,0]]

x,y=make_blobs(n_samples=5000,
centers=centroids, cluster_std=0.8)
```

Hosil qilingan tasodifiy nuqtalar klasterini grafik ko‘rinishi quyidagi dastur kodi orqali amalga oshiriladi.

```
plt.scatter(x[:,0],x[:,1], marker='.')
plt.show()
```



KMeans funksiyasi quyidagi parametrlarni qabul qiladi.

init - sentrodilarni tanlash usuli (k-means++ yoki random).

n_clusters - klastertlar soni.

n_init - algoritmni necha marta ishga tushirish (turli sentroidlar bilan qayta-qayta ishga tushirib, modelni qurishni boshlash uchun eng yaxshi sentroidlar tanlanadi).

```
k_means=KMeans(init="random",n_clusters=4,
n_init=15)
k_means.fit(x)
```

```
KMeans
KMeans(init='random', n_clusters=4, n_init=15)
```

Tasodifiy olingan nuqtalar klasterlandi, klaster raqamini ko‘rish uchun **.labels_** parametriga murojaat qilinadi.

```
k_means.labels_
```

```
array([1, 0, 0, ..., 3, 1, 1], dtype=int32)
```

Klaster markazlarini ko‘rich uchun **.cluster_centers_** parametriga murojaat qilinadi.

```
k_means.cluster_centers_
```

```
array([[ 1.99348887e+00, -3.01067931e+00],
       [-3.01979724e+00, -1.99489798e+00],
       [-3.05006801e-02,  1.12353254e-03],
       [ 2.97795387e+00,  2.99317691e+00]])
```

Algoritm topgan klaster markazlarini, dastlab avvaldan berilgan markazlar bilan solishtirish mumkin.

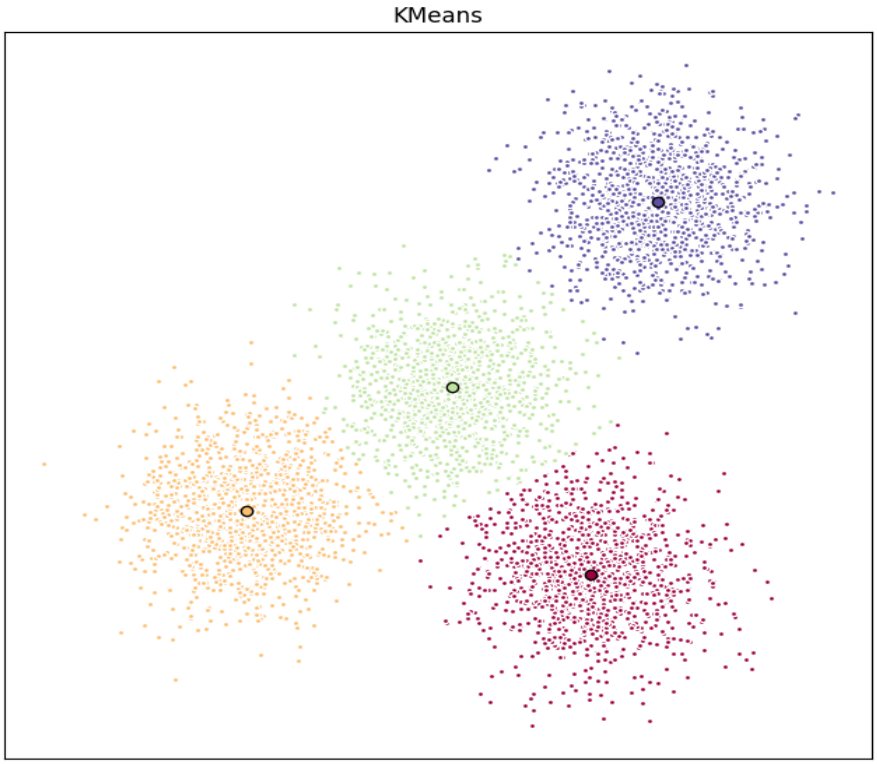
```
print(centroids)
```

```
[[3, 3], [-3, -2], [2, -3], [0, 0]]
```

Quyidagi dastur kodi orqali hosil qilingan klasterni grafik ko‘rinishi hosil qilinadi.

```
fig=plt.figure(figsize=(8,8))
```

```
# Har bir klaster uchun alohida rang.
colors=plt.cm.Spectral(np.linspace(0,1,
len(set(k_means.labels_))))
ax=fig.add_subplot(1,1,1)
for k, col in zip(range(len([[3,3],[-3,-2],[2,-
3],[0,0]])),colors):
  my_members=(k_means.labels_==k)
  cluster_center=k_means.cluster_centers_[k]
  ax.plot(x[my_members,0], x[my_members,1], 'w',
markerfacecolor=col,
  marker='.')
  ax.plot(cluster_center[0], cluster_center[1],
'o‘, markerfacecolor=col,
  markeredgecolor='k', markersize=6)
ax.set_title('Kmeans')
ax.set_xticks(())
ax.set_yticks(())
plt.show()
```



Yuqorida keltirilgan K-means algoritmi asosida ma’lumotlarni klasterlash imkoni yaratiladi. K-means algoritmi etiketlanmagan ma’lumotlar to‘plamini har bir ma’lumot to‘plami uchun o‘xshash

xususiyatlarga ega bo‘lgan faqat bitta guruhga tegishli bo‘lgan turli xil klasterlarga ajratish imkonini beradi.

Nazariy savollar

1. K-Means algoritmi va uning vazifasini tushuntiring?
2. K-Means algoritmini qo‘llanish sohalarini tushuntiring?
3. K-Means algoritmining bosqichlarini tushuntiring?
4. K-means klasterlashda “K klasterlar soni” qiymatini tanlashni tushuntiring?
5. Python dasturlash tilida K-means klasterlash algoritmini amalga oshirish bosqichlarini tushuntirib bering?
6. KMeans funksiyasini berilishi va uning vazifasini tushuntiring?
7. Tasodifiy nuqtalar klasterini yaratish funksiyasini berilishi va uning vazifasini tushuntiring?

4.3. Sun’iy neyron tarmoqlarning arxitekturasi va ishlash prinsplari

Reja;

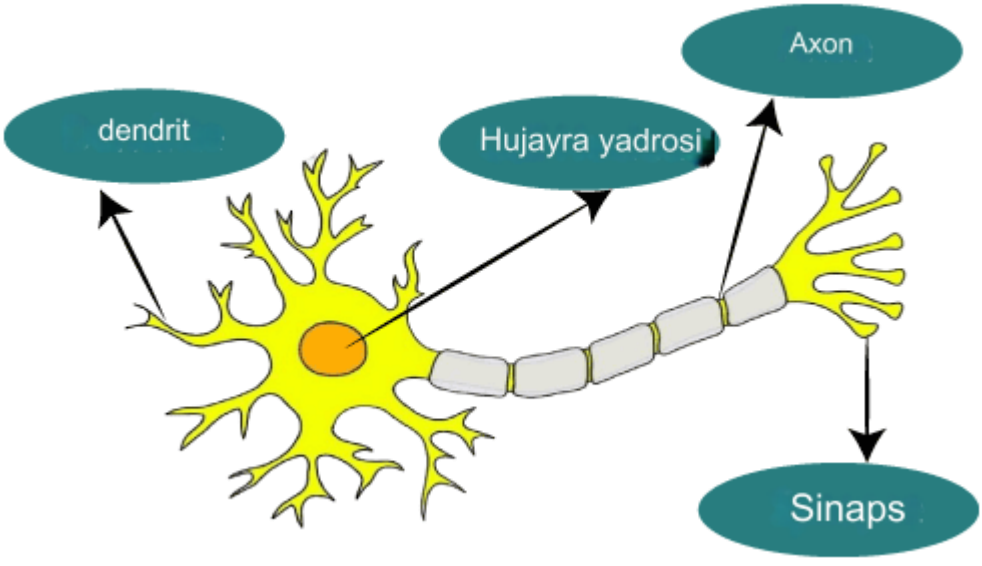
Sun’iy neyron tarmoqlari;

Sun’iy neyron tarmog‘ining arxitekturasi;

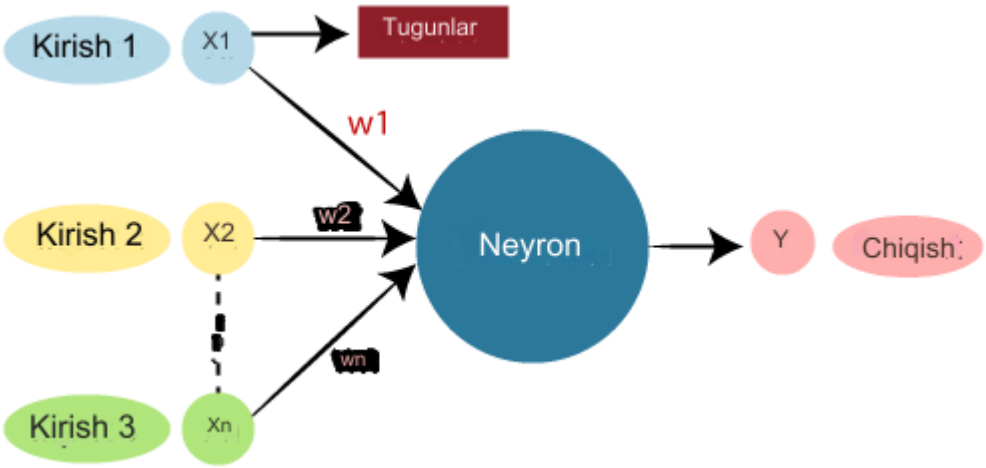
Neyron tarmoqlar asosida regression model ishlab chiqish.

Sun’iy neyron tarmoqlari

Sun’iy neyron tarmoqlari (SNN) bosh miya funksiyasiga asoslangan algoritmlar bo‘lib, murakkab naqshlarni modellashtirish va muammolarni yechimini prognoz qilish uchun ishlatiladi. Sun’iy neyron tarmoq (SNN) - bu inson miyasining biologik neyron tarmoqlari tushunchasidan kelib chiqqan chuqur o‘qitish usuli hisoblanadi va u inson miyasining ishlashini takrorlashga urinish natijasida ishlab chiqilgan. Sun’iy neyron tarmoqning ishlashi garchi ular bir xil bo‘lmasa ham biologik neyron tarmoqlarnikiga juda o‘xshaydi. Sun’iy neyron tarmoq algoritmi faqat raqamli va strukturalangan ma’lumotlarni qabul qiladi. Quyidagi rasmda biologik neyron tarmoqning odatiy grafik ko‘rinishi tasvirlangan.



Yuqoridagi rasmda keltirilgan neyronning ishlash funksiyasiga o‘xshatgan holda quyidagi rasm ko‘rinishidagi sun’iy neyron tarmoq ishlab chiqilgan.



Dendrit - boshqa neyronlarning aksonlaridan kimyoviy (yoki elektr) sinapslar orqali ma’lumot oladigan va uni elektr signali orqali neyron tanasiga (perikarion) uzatuvchi neyronning tarmoqlangan kengaytmasi.

Sinaps - nerv hujayralari (neyronlar)ning o‘zaro va ijrochi organlar hujayralari bilan tutashgan joyi.

Akson - bu nerv (asab hujayrasining uzun silindrsimon jarayoni), u nerv impulslarining hujayra tanasidan (soma) innervatsiya qilingan organlarga va boshqa nerv hujayralariga o‘tishni amalga oshiradi.

Sun’iy neyron tarmoqlarida dendritlar kirishlarni, hujayra yadrosi tugunlarni, sinaps og‘irliklarni va akson chiqishni ifodalaydi.

Biologik neyron tarmoq va sun’iy neyron tarmoq o‘rtasidagi o‘xshashlik quyidagi jadvalda keltirilgan.

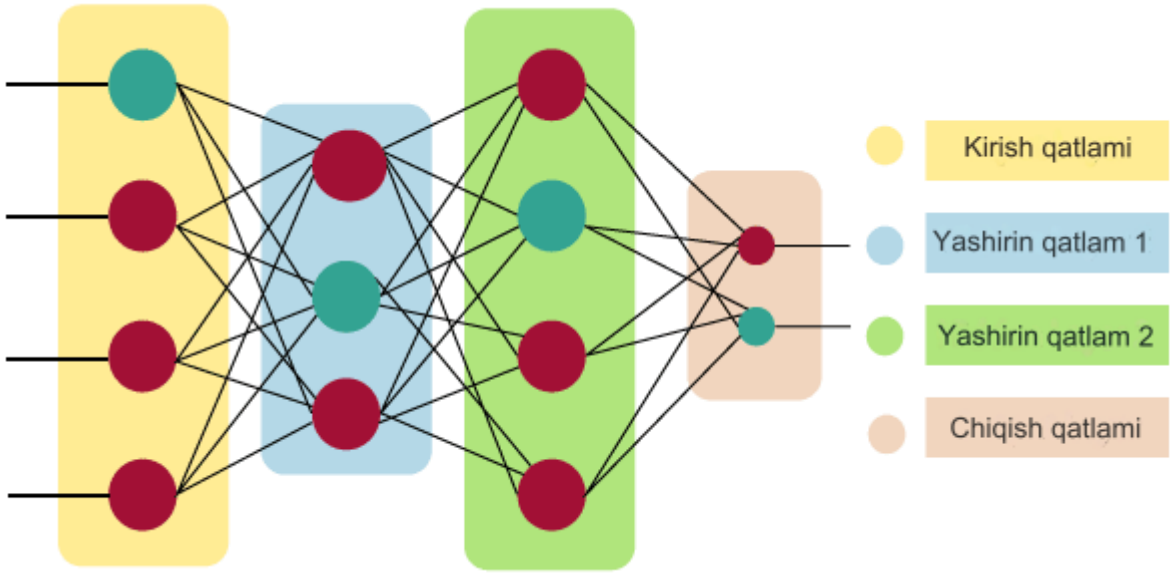
Biologik neyron tarmog‘i	Sun’iy neyron tarmog‘i
Dendritlar	Kirishlar
Hujayra yadrosi	Tugunlar
Sinaps	Og‘irliklar
Axon	Chiqish

Sun’iy intellekt sohasidagi **sun’iy neyron tarmog‘i** biologik neyron tarmog‘iga taqlid qilishga harakat qiladi, bunda kompyuterlar jarayonlarni tushunish va insonga o‘xshash tarzda qaror qabul qilish imkoniyatiga ega bo‘lishi tushuniladi. Inson miyasida 1000 milliardga yaqin neyronlar mavjud. Inson miyasida ma’lumotlar taqsimlanadigan

tarzda saqlanadi va biz kerak bo‘lganda bu ma’lumotlarning bir nechta qismini parallel ravishda xotiramizdan chiqarib olishimiz mumkin. Aytish mumkinki, inson miyasi aql bovar qilmaydigan darajada ajoyib parallel protsessorlardan iborat.

Sun’iy neyron tarmog‘ining arxitekturasi

Sun’iy neyron tarmog‘i arxitekturasi tushunchasini tushunish uchun neyron tarmoq nimadan iborat ekanligini tushunish kerak bo‘ladi. Sun’iy neyron tarmog‘i asosan uchta qatlamdan iborat bo‘ladi.

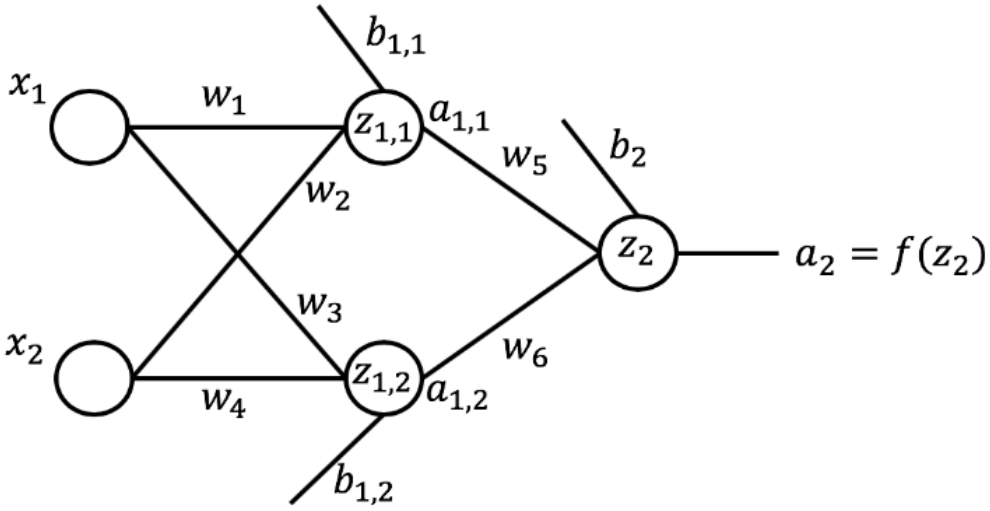


Kirish qatlami. Neyron tarmoqning birinchi qatlami, bu yerda kirish ma’lumotlari tarmoqqa uzatiladi. Kirish qatlamidagi har bir neyron kirish ma’lumotlarining xususiyatini ifodalaydi.

Yashirin qatlamlar. Hisoblashlar amalga oshiriladigan kirish va chiqish qatlamlari orasidagi oraliq qatlamlar. Har bir yashirin qatlam bir nechta neyronlardan iborat bo‘lib, har bir qatlamdagi yashirin qatlamlar va neyronlar soni muammoning murakkabligiga qarab o‘zgarishi mumkin.

Chiqish qatlami. Tarmoqning chiqishini ishlab chiqaradigan neyron tarmoqning oxirgi qatlami. Chiqish qatlamidagi neyronlar soni muammoning tabiatiga bog‘liq (masalan, regressiya, tasniflash).

Quyida yuqoridagi arxitekturani tushunish uchun oddiy 2 ta kirish neyronli, ikkita yashirin qatlamli va bitta chiqish neyronli arxitektura ko‘rib chiqiladi.



Bunda yakuniy chiqish neyroni quyidagi foymulalar orali hisoblanadi.

$$z_{11} = x_1 w_1 + x_2 w_2 + b_{11}$$

$$z_{12} = x_1 w_3 + x_2 w_4 + b_{12}$$

$$a_{11} = f(z_{11}) = \frac{1}{1+e^{-z_{11}}}$$

$$a_{12} = f(z_{12}) = \frac{1}{1+e^{-z_{12}}}$$

$$z_2 = a_{11} w_5 + a_{12} w_6 + b_2$$

$$a_2 = f(z_2) = \frac{1}{1+e^{-z_2}}$$

Bunda:

w - vazn(og‘irlik koeffitseyntlari)lar, neyronlar orasidagi aloqalar neyronlar o‘rtasidagi munosabatlarning kuchini ifodalaydi. Har bir vazn kirish signalining chiqish signaliga qanchalik ta’sir qilishini aniqlaydigan qiymat;

f - faollashtirish funktsiyasi, har bir neyron tarmoqqa nochiziqlilikni kiritish va unga murakkab naqshlarni o‘rganish imkonini berish uchun kirishlarning vaznli yig‘indisiga faollashtirish funksiyasi qo‘llaniladi. Umumiy faollashtirish funktsiyalariga

sigmasimon(sigmoida), tanh(tangensial), ReLU (Rektifikatsiyalangan chiziqli birlik) va softmax funksiyalari kiradi;

b - bias neyron tarmog‘ida faollashtirish funksiyasini gorizontal ravishda siljitish orqali ma’lumotlardagi murakkabroq naqsh va munosabatlarni qo‘lga kiritish imkonini beradi.

Shunday qilib, sun’iy neyron tarmoqning umumiy matematik formulasini quyidagicha ifodalash mumkin:

$$y = f(\sum_{i=1}^{n} w_i x_i + b)$$

Sun’iy neyron tarmoqlarning arxitekturasi, qatlamlar bog‘lanishi va bajarish uchun mo‘ljallangan vazifalariga qarab har xil turlarga bo‘linishi mumkin. Sun’iy neyron tarmog‘i kirish ma’lumotlarini bir marta tanlangan vazn qiymatlari bo‘yicha to‘liq o‘qitilishni amalga oshiradi va bu jarayon *n* marta iterativ tarzda amalga oshiradi. Har iteratsiyada vaznlar yangilanadi va eng yaxshi vazn aniqlangan holda model quriladi. Sun’iy neyron tarmoqlarning eng keng tarqalgan turlariga quyidagilar kiradi:

Feedforward neyron tarmoqlari (FNN-oldinga yo‘naltirilgan). FNN neyron tarmog‘ining eng oddiy turi bo‘lib, unda ma’lumotlar bir yo‘nalishda, kirish qatlamidan chiqish qatlamiga, hech qanday aylanishlarsiz o‘tadi. Ular regressiya va tasniflash kabi vazifalar uchun keng qo‘llaniladi.

Multilayer Perceptrons (MLP-ko‘p qatlamli perseptronlar). MLP - kirish qatlami, bir yoki bir nechta yashirin qatlamlar va chiqish qatlamini o‘z ichiga olgan Feedforward neyron tarmog‘ining bir turi. MLP ma’lumotlardagi murakkab chiziqli bo‘lmagan munosabatlarni o‘rganishga qodir va odatda turli xil mashinaviy o‘qitish vazifalarida qo‘llaniladi.

Convolution neyron tarmoqlari (CNN-konvolyutsiya). CNN tasvirlar kabi tuzilgan ma’lumotlarni qayta ishlash uchun maxsus mo‘ljallangan arxitektura hisoblanadi. CNN kirish ma’lumotlariga filtrlarni qo‘llaydigan konvolyutsion qatlamlardan iborat. CNN tasvirlarni

aniqlash, ob’ektlarni aniqlash va boshqa kompyuterli ko‘rish vazifalarida keng qo‘llaniladi.

Recurrent neyron tarmoqlari (RNN-takroriy). RNN ketma-ket tuzulishga ega ma’lumotlarni qayta ishlash uchun mo‘ljallangan, bu yerda kirish tartibi muhim ahamiyatga ega. Ularda ma’lumotlarning vaqt o‘tishi bilan saqlanib qolishiga imkon beruvchi aloqa xotiralari mavjud bo‘lib, ularni nutqni aniqlash, tilni modellashtirish va vaqt seriyalarini bashorat qilish kabi vazifalarni amalga oshirishda ishlatiladi

Long Short-Term Memory tarmoqlari (LSTM- Uzoq qisqa muddatli xotira). LSTM uzoq ketma-ketliklar bo‘yicha standart RNNlarni o‘rgatishda yuzaga keladigan yo‘qolib borayotgan gradient muammosini hal qiluvchi RNNning ixtisoslashgan turi hisoblanadi. Ular ketma-ket ma’lumotlardagi uzoq muddatli bog‘liqlikni yanada samarali o‘rganish imkonini beruvchi xotira hujayralari va mexanizmlariga ega.

Gated Recurrent Units (GRU- Darvozali takrorlanuvchi birliklar). GRU LSTMga o‘xshaydi va RNNda yo‘qolib borayotgan gradient muammosini hal qilish uchun mo‘ljallangan. Ular LSTM larga qaraganda kamroq parametrlarga ega va ba’zan oddiyroq vazifalar uchun yoki hisoblash resurslari cheklangan hollarda afzallik beradi.

Neyron tarmoqlar asosida regression modelni ishlab chiqish

Quyida Beton mahsulotlarni mustahkamligini hisoblovchi Feedforward neyron tarmog‘iga asoslangan modelni ishlab chiqish qarab o‘tiladi.

1-qadam. Kerakli kutubxonalar import qilinadi.

```
import pandas as pd
import numpy as np
import tensorflow as tf
from sklearn.model_selection import
train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import mean_absolute_error,
mean_squared_error, r2_score
from tensorflow import keras
from keras.models import Sequential
```

```
from keras.layers import Dense
```

Bunda:

import tensorflow as tf – TensorFlow kutubxonasini import qiladi. TensorFlow - bu Google tomonidan ishlab chiqilgan ochiq manbali mashinaviy o‘qitish tizimi. U mashinaviy o‘qitish modellarini, shu jumladan chuqur o‘qitish modellarini yaratish, o‘qitish va joylashtirish uchun vositalarni taqdim etadi;

from tensorflow import keras – Keras kutubxonasini import qiladi, u neyron tarmoqlarni qurish va o‘qitish uchun funksionallikni ta’minlaydi;

from keras.models import Sequential – Kerasdan Sequential sinfini import qiladi. U qatlamlarni bir-birining ustiga ketma-ket joylashtirish orqali neyron tarmoq modelini yaratish imkonini beradi. Sequential model odatda ma’lumotlar qatlamlar bo‘ylab ketma-ket oqadigan neyron tarmoqlari uchun ishlatiladi;

from keras.layers import Dense – Kerasdan neyron tarmoqdagi to‘liq bog‘langan qatlamni ifodalovchi Dense sinfini import qiladi. Kerasdagi har bir Dense qatlam neyronlar to‘plamidan iborat bo‘lib, har bir neyron oldingi qatlamdagi neyronlarga to‘liq bog‘langan bo‘ladi. Dense qatlam tarmoqqa kirish ma’lumotlaridan murakkab naqshlarni o‘rganish imkonini beruvchi faollashtirish funktsiyasidan so‘ng oddiy chiziqli transformatsiyani amalga oshiradi.

Masala aniq bo‘lishi uchun tayyor ma’lumotlar to‘plamini yuklab olish asosida amalga oshirilgan.

2-qadam. Kerakli ma’lumotlar to‘plami yuklab olinadi.

```
df = pd.read_csv('https://s3-api.us-geo.objectstorage.softlayer.net/cf-courses-data/CognitiveClass/DL0101EN/labs/data/concrete_data.csv')
df.head()
```

	Cement	Blast Furnace Slag	Fly Ash	Water	Superplasticizer	Coarse Aggregate	Fine Aggregate	Age	Strength
0	540.0	0.0	0.0	162.0	2.5	1040.0	676.0	28	79.99
1	540.0	0.0	0.0	162.0	2.5	1055.0	676.0	28	61.89
2	332.5	142.5	0.0	228.0	0.0	932.0	594.0	270	40.27
3	332.5	142.5	0.0	228.0	0.0	932.0	594.0	365	41.05
4	198.6	132.4	0.0	192.0	0.0	978.4	825.5	360	44.30

Bunda “Strength” Betonni mustahkamligini ifodalovchi xususiyat. Qolgan ustunlar “Strength” xususiyatini bashorat qilishda foydalaniladigan xususiyatlar hisoblandi.

3-qadam. Ma’lumotlar to‘plami o‘rganiladi.

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1030 entries, 0 to 1029
Data columns (total 9 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   Cement              1030 non-null   float64
 1   Blast Furnace Slag  1030 non-null   float64
 2   Fly Ash             1030 non-null   float64
 3   Water               1030 non-null   float64
 4   Superplasticizer    1030 non-null   float64
 5   Coarse Aggregate    1030 non-null   float64
 6   Fine Aggregate      1030 non-null   float64
 7   Age                 1030 non-null   int64  
 8   Strength            1030 non-null   float64
dtypes: float64(8), int64(1)
memory usage: 72.5 KB
```

Demak, qaralayotgan ma’lumotlar to‘plamida (Datasetda) NaN va satrli qiymatlar mavjud emas.

4-qadam. Ma’lumotlar to‘plami test va o‘quv to‘plamlarga ajratiladi.

```
train_set, test_set = train_test_split(df,
test_size=0.10, random_state=42)

x_train = train_set.drop('Strength',
axis=1).values
y_train = train_set['Strength'].values
x_test = test_set.drop('Strength',
axis=1).values
y_test = test_set['Strength'].values
```

5-qadam. Bog‘liq o‘zgaruvchilar normallashtiriladi.

```
scaler = StandardScaler()
x_train = scaler.fit_transform(x_train)
x_test = scaler.transform(x_test)
```

6-qadam. Feedforward neyron tarmog‘i arxitekturasi quriladi.

```
model = keras.Sequential([keras.layers.Dense(64,
activation='relu',
input_shape=(x_train.shape[1],)),
keras.layers.Dense(32, activation='relu'),
keras.layers.Dense(1)])
```

7-qadam. Keras/TensorFlow-da neyron tarmoq modeli kompilyatsiya qilinadi.

```
model.compile(optimizer='adam',loss='mean_square
d_error', metrics=['mean_absolute_error'])
```

Bunda:

optimizator='adam'–argumenti o‘qitish davomida foydalaniladigan optimallashtiruvchini belgilaydi va har bir parametr uchun o‘qitish tezligini moslashtiradi, bu esa tezroq konvergentsiya bo‘lish va yaxshi ishlashga olib keladi;

loss='mean_squared_error'–argumenti o‘qitish davomida optimallashtiriladigan yo‘qotish funksiyasini belgilaydi. Bunday holda, yo‘qotish funktsiyasi sifatida MSE ishlatiladi.

Quyidagi kod qatori taqdim etilgan o‘quv ma’lumotlari (*x_train* va *y_train*) yordamida neyron tarmoq modelini o‘qitishni amalga oshiradi.

model.compile(optimizer='adam', loss='mean_squared_error', metrics=['mean_absolute_error'])

epochs=100 – argumenti butun o‘quv ma’lumotlar to‘plami bo‘yicha davrlar yoki iteratsiyalar sonini belgilaydi. Davr - bu butun o‘quv ma'lumotlar to‘plamidan bir marta o‘tish. Bunday holda, model 100 davr uchun o‘qitiladi, ya’ni u mashg‘ulot davomida o‘quv ma’lumotlar to‘plamini 100 marta ko‘rib chiqadi.

batch_size=32 – argumenti o‘qitish uchun ishlatiladigan to‘plam hajmini belgilaydi. U o‘qitish davomida har bir iteratsiyada (yoki partiyada) qayta ishlangan o‘quv misollari sonini aniqlaydi. Bunday holda, model vaznini yangilashdan oldin bir vaqtning o‘zida 32 ta o‘quv misolini qayta ishlaydi. Iteratsiyaning kichik o‘lchamlari model og‘irliklarining tez-tez yangilanishiga olib kelishi mumkin, lekin sekinroq bo‘lishi mumkin, kattaroq iteratsiyalar esa tezroq o‘qitishga olib kelishi mumkin.

verbose=2 – argumenti o‘qitish davomida batafsil ma’lumot rejimini boshqaradi. U o‘qitish davomida o‘quv jarayoni yangilanishlarini chop etish yoki chop etmaslikni aniqlaydi.

verbose=0: Jimlik rejim, bu yerda trening davomida hech qanday natija chop etilmaydi.

verbose=1: Harakatlanish satri rejimi, bunda har bir davrning borishini ko‘rsatuvchi satr ko‘rsatiladi.

verbose=2: Batafsil rejim, bu yerda mashg‘ulot jarayoni yangilanishlari har bir davrdan keyin chop etiladi.

Model test to‘plami orqali quyidagicha baholanadi.

```
y_predict = model.predict(x_test)
MAE = mean_absolute_error(y_test, y_predict)
RMSE = np.sqrt(mean_squared_error(y_test,
y_predict))
r2 = r2_score(y_test, y_predict)
print(f'MAE: {MAE}, RMSE: {RMSE}, R^2: {r2}')
```

```
Epoch 100/100
29/29 - 0s - loss: 33.9765 - mean_absolute_error: 4.4466 - 78ms/epoch - 3ms/step
4/4 [==============================] - 0s 11ms/step
MAE: 4.868157850098841, RMSE: 6.107450840638139, R^2: 0.8597329114545498
```

Demak yuqorida keltirilgan qoida va modellar asosida sun’iy neyron tarmoq arxitekturasi quriladi hamda qurilgan sun’iy neyron tarmoqqa berilgan to‘plamni o‘qitish asosida natijalar olinadi.

Nazariy savollar

1. Sun’iy neyron tarmoqlari va uning vazifasini tushuntiring?

2. Dendrit va uning vazifasini tushuntiring?
3. Sinaps va uning vazifasini tushuntiring?
4. Akson va uning vazifasini tushuntiring?
5. Sun'iy neyron tarmog'ining arxitekturasini tushuntiring?
6. Kirish qatlami va uning vazifasini tushuntiring?
7. Yashirin qatlamlar va uning vazifasini tushuntiring?
8. Chiqish qatlamlari va uning vazifasini tushuntiring?
9. Feedforward neyron tarmoqlari va uning vazifasini tushuntiring?
10. Multilayer Perceptrons neyron tarmoqlari va uning vazifasini tushuntiring?
11. Convolution neyron tarmoqlari va uning vazifasini tushuntiring?
12. Recurrent neyron tarmoqlari va uning vazifasini tushuntiring?
13. Long Short-Term Memory tarmoqlari va uning vazifasini tushuntiring?
14. Gated Recurrent Units neyron tarmoqlari va uning vazifasini tushuntiring?
15. Neyron tarmoqlar asosida regression model ishlab chiqish jarayonini tushntiring?
16. import tensorflow as tf va uning vazifasini tushuntiring?
17. from tensorflow import keras va uning vazifasini tushuntiring?
18. from keras.models import Sequential va uning vazifasini tushuntiring?
19. from keras.layers import Dense va uning vazifasini tushuntiring?
20. Epochs argumenti va uning vazifasini tushuntiring?
21. batch_size argumenti va uning vazifasini tushuntiring?
22. verbose argumenti va uning vazifasini tushuntiring?

Xulosa

Hozirgi vaqtda ma’lumotlar oqimining ko‘pligi tufayli ularni qisqa vaqt ichida qayta ishlash muommosi ham ortib bormoqda. Sun’iy intellekt algoritmlarini rivojlantirish, ma’lumotlarni statistik tahlil qilish, oldindan intellektual qayta ishlash kabi muammolarni hal etishning yechimlaridan biri hisoblanadi. Zamonaviy jamiyatda tobora o‘sib borayotgan axborot oqimi, axborot texnologiyalarining turli tumanligi, kompyuterda yechiladigan masalalarning murakkablashuvi, ushbu texnologiyalardan foydalanuvchining oldiga bir qator vazifalarni qo‘ydi. Bu vazifalarni bosqichma bosqich raqamlashtirish va sun’iy intellekt tizimlari orqali hal etish masalasini yechish uchun mazkur darslik ma’lum bir vosita bo‘lib xizmat qiladi.

Mazkur darslik sun’iy intellekt asoslari fanini o‘rganishga bag‘ishlangan bo‘lib, bunda sun’iy intellekt asoslari fani uchun sun’iy intellekt haqida dastlabki ma’lumotlar, sun’iy intellekt uchun python dasturlash kutubxonalari, mashinaviy o‘qitish, nazoratli va nazoratsiz o‘qitish, regressiya va klassifikatsiya algoritmlari va ularni qo‘llashning qoidalari batafsil keltirib o‘tilgan.

Mazkur darslikda sun’iy intellekt va ma’lumotlarga ishlov berish, mashinaviy o‘qitish, mashinaviy o‘qitishning regressiya algoritmlari, mashinaviy o‘qitishda klassifikatsiya algoritmlari, mashinaviy o‘qitishda klasterizatsiya algoritmlari keltirib o‘tilgan.

Bu darslik nafaqat pedagogik va muhandis yo‘nalishidagi talabalar, balki mustaqil o‘rganuvchilar uchun ham asosiy qo‘llanma hisoblanadi. Mazkur darslik dasturiy injiniring, sun’iy intellekt yo‘nalishlarining o‘quv rejasi hamda o‘quv dasturi asosida yaratildi. Har bir mavzu bo‘yicha ma’lumotlar nazariy, amaliy qismlardan iborat va mavzu oxirida nazariy savollar to‘plami keltirib o‘tilgan. Har bir ishlab chiqilgan algoritm python dasturlash tilida dasturlanib, natijalari olingan.

GLOSSARIY

Sun’iy intellekt - inson intellektiga taqlid qilishga qodir bo‘lgan mashinalar yaratishga qaratilgan fan va texnologiya sohasi.

Mashinaviy o‘qitish - sun’iy intellekt sohasi bo‘lib, katta hajmdagi ma’lumotlardan foydalangan holda inson ko‘magisiz ham o‘zi o‘rganish hamda o‘rgangan ma’lumotlari asosida bashorat qilish va qaror chiqarish imkoniyatini beradigan algoritmlar.

Teachable Machine - bu vebga asoslangan vosita bo‘lib, u mashinaviy o‘qitish modellarini tez, oson va hamma uchun ochiq qiladi.

Anaconda - python dasturlash tilining kompilyatori.

Jupyter Notebook - python dasturlash tilining komplyatori.

Google Colab - google kompaniyasiga tegishli python dasturlash tilining komplyatori.

NumPy - python dasturlash tili uchun kutubxona bo‘lib, katta, ko‘p o‘lchovli massivlar va matritsalar ustida bajariladigan amallarni qo‘llab-quvvatlaydi.

Sorting - tartiblashni amalga oshirishga mo‘ljallangan algoritm va funksiya hisoblanadi.

Pandas - bu ochiq manbali kutubxona bo‘lib, u asosan bir-biri bilan bog‘langan yoki etiketli ma'lumotlar bilan oson va intuitiv tarzda ishlash uchun yaratilgan.

DataFrame -2 o‘lchovli ma’lumotlar strukturasi bo‘lib, 2 o‘lchovli massiv yoki satr va ustunli jadval kabi ko‘rinishda bo‘ladi.

Korrelyatsiya - kerakli ustunlarning bir-biriga bog‘liqligini tekshirish uchun mo‘ljallangan funksiya.

HDF5 - ilmiy tadqiqodlar qilishda katta hajmdagi ma’lumotlarni saqlash uchun keng ishlatiladigan formatlardan biri hisoblanadi.

Vizuallashtirish - ma’lumotlarni grafik shaklda tasvirlanishi hisoblanadi.

Regressiya - kirish o‘zgaruvchisi va chiqish o‘zgaruvchisi o‘rtasida bog‘liqlik asosida hisoblanadigan matematik jarayon.

Nazoratli o‘qitish - berilgan ma’lumotlar tayanib qurilgan model asosida o‘qitish.

Klassifikatsiya - bu ma’lumotlar to‘plamini turli parametrlar asosida sinflarga bo‘lishda yordam beradigan funksiyani topish jarayoni hisoblanadi.

Nazoratsiz o‘qitish - tasniflanmagan va etiketlanmagan ma’lumotlardan foydalangan holda algoritmga ushbu ma’lumotlarga ko‘rsatmalarsiz harakat qilish imkonini beradigan mashinaviy o‘qitish usuli hisoblanadi.

Chiziqli regressiya - bu bashoratli tahlil uchun ishlatiladigan statistik regressiya usuli hisoblanadi.

O‘qitish to‘plami - bu to‘plam model yaratish uchun kerak bo‘ladi.

Test to‘plami - bu to‘plam model aniqligini tekshirish uchun kerak bo‘ladi.

Ko‘p chiziqli regressiya - bitta bog‘liq doimiy o‘zgaruvchi va bir nechta mustaqil o‘zgaruvchilar o‘rtasidagi chiziqli munosabatni modellashtiradigan muhim regressiya algoritmlaridan biridir.

Polinomial regressiya - bog‘liq va mustaqil o‘zgaruvchi o‘rtasidagi munosabatni *n*-darajali ko‘phad sifatida modellashtiruvchi regressiya algoritmi hisoblanadi.

K-NN - nazorat ostidagi mashinaviy o‘qitish algoritmi bo‘lib, u ham tasniflash, ham regressiya muammolarini hal qilish uchun ishlatilishi mumkin bo‘lgan algoritm hisoblanadi.

Qaror daraxti - bu tasniflash va regressiya muammolari uchun ishlatilishi mumkin bo‘lgan nazoratli o‘qitish usuli, lekin asosan tasniflash muammolarini hal qilish uchun yaxshi natijalarni ko‘rsatadi.

Klasterlash - bu yorliqsiz ma’lumotlar to‘plamini guruhlaydigan mashinaviy o‘qitish usuli hisoblanadi.

K-means - bu nazoratsiz o‘qitish algoritmi bo‘lib, yorliqsiz ma’lumotlar to‘plamini turli klasterlarga guruhlaydi.

FOYDALANILGAN ADABIYOTLAR RO‘YXATI

1. Wolfgang Ertel. “Introduction to Artificial Intelligence”. Germany-2017.

2. Joshua Eckroth. “Python Artificial Intelligence Projects for Beginners”.Birmingham 2018.

3. Philip C. Jackson, Jr. “Introduction to Artificial Intelligence”. United States-2019.

4. Shengquan Yu. “An Introduction to Artificial Intelligence in Education”. China-2021.

5. Itisha Gupta, Garima Nagpal. Artificial Intelligence and Expert Systems. New Delhi-2020

6. Fuwei Li, Lifeng Lai, Shuguang Cui. Machine Learning Algorithms. 2022

7. S. Komolov, Sh. Raxmatov. Sun’iy intellekt asoslari. Mashinaviy o‘qitish. Toshkent-2022.

8. A. Narzullaev, M. Abdullayev. “Data Science va Sun’iy Intellekt” onlayn kursi. 2021.

9. Nigmatov H. “Intellektual tizimlar”. Toshkent-2019.

MUALLIFLAR HAQIDA MA’LUMOT

Nazarov Fayzullo Maxmadiyarovich 1990 yil 18-avgust kuni Samarqand viloyati Urgut tumani Taroqli mahallasining ziyoli oilasida tavallud topgan. Nazarov Fayzullo Maxmadiyarovich texnika fanlari bo‘yicha falsafa doktori ilmiy darajaga va dotsent ilmiy unvoniga ega. Nazarov Fayzullo Maxmadiyarovich 2 ta darslik, 5 ta o‘quv qo‘llanma, 10 dan ortiq uslubiy qo‘llanma, 150 dan ortiq ilmiy maqola va tezislar hamda 20 ga yaqin dasturiy mahsulotlar uchun guvohnomalar muallifi hisoblanadi. Hozirgi vaqtda Nazarov Fayzullo Maxmadiyarovich Samarqand davlat universiteti intellektual tizimlar va kompyuter texnologiyalari fakulteti dekani lavozimida faoliyat olib bormoqda. fayzulla-samsu@mail.ru

Yarmatov Sherzodjon Shokir o‘g‘li 1996 yil 26-yanvar kuni Samarqand viloyati Ishtixon tumani Qoraqo‘yli mahallasining ziyoli oilasida tavallud topgan. Yarmatov Sherzodjon Shokir o‘g‘li 40 dan ortiq ilmiy maqola va tezislar hamda 5 ga yaqin dasturiy mahsulotlar uchun guvohnomalar muallifi hisoblanadi. Hozirgi vaqtda Yarmatov Sherzodjon Shokir o‘g‘li Samarqand davlat universiteti sun’iy intellekt va axborot tizimlari kafedrasining mudiri lavozimida faoliyat olib bormoqda. sherzod2601@gmail.com