O‘ZBEKISTON RESPUBLIKASI RAQAMLI TEXNOLOGIYALAR VAZIRLIGI
MUHAMMAD AL-XORAZMIY NOMIDAGI TOSHKENT AXBOROT TEXNOLOGIYALARI UNIVERSITETI SAMARQAND FILIALI

KOMPYUTER INJINIRINGI FAKULTETI
MA'LUMOTLAR TAHLILI FANIDAN

1-Mustaqil ish

Python kutubxonalari (Pandas, NumPy) yordamida ma’lumotlar bilan ishlash

Guruh: 22-06 talabasi.

Bajardi: AbdisodiqovDiyorbek

Qabul qildi: Ismailov I. T.

Samarqand-2025

Reja:

1. NumPy kutubxonasi va uning imkoniyatlari.
2. Pandas kutubxonasi va ma'lumotlar tahlili.
3. NumPy va Pandas kutubxonalarining amaliy qo‘llanilishi.

NumPy kutubxonasi va uning imkoniyatlari

Hozirgi kunda ma'lumotlar tahlili va ilmiy hisoblashlar sohasida Python dasturlash tili eng mashhur va keng qo‘llaniladigan vositalardan biriga aylandi. Python tilining muvaffaqiyati uning soddaligi, o‘qish va tushunish qulayligi hamda kuchli kutubxonalar ekotizimiga ega ekanligida. Ma'lumotlar tahlili va ilmiy hisoblashlar uchun Python tilida ishlab chiqilgan ko‘plab kutubxonalar mavjud bo‘lib, ularning orasida NumPy va Pandas kutubxonalari alohida o‘rin tutadi.

NumPy kutubxonasi Python tilida ilmiy hisoblashlar uchun asosiy paket hisoblanadi. U ko‘p o‘lchovli massivlar va matritsalarni samarali boshqarish, matematik va mantiqiy operatsiyalarni tez bajarish imkonini beradi. NumPy kutubxonasi C va Fortran tillarida yozilgan bo‘lib, bu unga yuqori ishlash tezligini ta'minlaydi. Kutubxona 1995 yilda Numeric nomi bilan yaratilgan va 2006 yilda NumPy nomi bilan qayta ishlab chiqilgan. Bugungi kunda NumPy ilmiy Python ekotizimining asosi hisoblanadi va ko‘plab boshqa kutubxonalar, jumladan Pandas, SciPy, Matplotlib va boshqalar NumPy massivlaridan foydalanadi.

Pandas kutubxonasi esa strukturali ma'lumotlar bilan ishlash uchun yuqori darajadagi vosita hisoblanadi. U ma'lumotlarni jadval ko‘rinishida saqlash va qayta ishlash, ma'lumotlar tahlili, tozalash va transformatsiya qilish kabi vazifalarni bajarish uchun mo‘ljallangan. Pandas kutubxonasi 2008 yilda AQSh moliyaviy firmasi ishchisi Wes McKinney tomonidan yaratilgan va tez orada ma'lumotlar tahlili sohasida standart vositaga aylangan. Pandas kutubxonasi ichida NumPy kutubxonasidan foydalanadi va uning yuqori tezligidan bahramand bo‘ladi. Pandas kutubxonasi DataFrame va Series kabi ob'ektlarni taqdim etadi, ular ma'lumotlar bilan ishlashni juda qulay va samarali qiladi.

NumPy va Pandas kutubxonalarining asosiy imkoniyatlari, ularning amaliy qollanilishi va ma'lumotlar tahlilida tutgan o‘rni haqida batafsil

ma'lumot berishdan iborat.Ma'lumotlar tahlili zamonaviy dunyoda juda muhim rol o‘ynaydi. Korxonalar, ilmiy tadqiqot institutlari, davlat muassasalari va boshqa tashkilotlar har kuni katta hajmdagi ma'lumotlarni to‘playdi va tahlil qiladi. Bu ma'lumotlardan to‘g‘ri xulosalar chiqarish, naqshlarni aniqlash va bashoratlar qilish uchun samarali vositalar zarur. NumPy va Pandas kutubxonalari aynan shunday vositalar hisoblanadi. Ular ma'lumotlar bilan ishlashni tezlashtiradi, xatolar ehtimolini kamaytiradi va murakkab tahlillarni osonlashtiradi.

NumPy (Numerical Python) kutubxonasi Python tilida ilmiy hisoblashlar uchun asosiy kutubxona hisoblanadi. Uning markaziy elementi ndarray (N-dimensional array) ob'ekti bo‘lib, bu bir xil turdagi ma'lumotlarni saqlash uchun mo‘ljallangan ko‘p o‘lchovli massivdir. NumPy massivlari Python ro‘yxatlariga qaraganda ancha tezroq ishlaydi, chunki ular xotirada ketma-ket joylashgan va C tilida yozilgan optimallashtirilgan funksiyalar yordamida qayta ishlanadi. NumPy kutubxonasi matematik operatsiyalar, tasodifiy sonlar generatsiyasi, chiziqli algebra, Fourier transformatsiyasi va boshqa ko‘plab matematik amallarni bajarish uchun keng imkoniyatlar taqdim etadi.

NumPy massivlari ikki asosiy xususiyatga ega: shape (shakl) va dtype (ma'lumot turi). Shape massivning o‘lchamlari haqida ma'lumot beradi, masalan, (3, 4) shaklidagi massiv 3 qator va 4 ustundan iborat ikki o‘lchovli massivni bildiradi. Dtype esa massivda saqlanadigan ma'lumotlar turini ko‘rsatadi, masalan, int32, float64, complex128 va boshqalar. NumPy massivlarida faqat bir xil turdagi ma'lumotlar saqlanishi kerak, bu esa xotiradan samarali foydalanish va tez hisoblashlarni ta'minlaydi. NumPy kutubxonasi vectorization (vektorlashtirish) konsepsiyasini qo‘llab-quvvatlaydi, ya'ni bir vaqtning o‘zida butun massiv ustida operatsiyalar bajarish mumkin.

1-rasm.Eng mashhur Python kutubxonalari.

NumPy kutubxonasini o‘rnatish juda oddiy va pip paket menejeri yordamida amalga oshiriladi. Odatda 'pip install numpy' buyrug‘i yordamida kutubxona o‘rnatiladi. o‘rnatilgandan so‘ng, NumPy kutubxonasini dasturda import qilish kerak: 'import numpy as np'. Bu yerda 'np' - bu NumPy uchun keng tarqalgan qisqartma bo‘lib, barcha dasturchilar orasida standart hisoblanadi. NumPy kutubxonasi doimiy ravishda yangilanib turadi va yangi versiyalarda yangi funksiyalar, ishlash tezligini oshirish va xatolarni tuzatish kiritiladi. Hozirgi kunda NumPy versiya 1.26 gacha rivojlangan va millionlab dasturchilar tomonidan foydalanilmoqda.

NumPy massivlarini yaratishning ko‘plab usullari mavjud. Eng oddiy usul - bu Python ro‘yxatidan massiv yaratish: np.array([1, 2, 3, 4, 5]). Bundan tashqari, maxsus funksiyalar yordamida ham massivlar yaratish mumkin. Masalan, np.zeros() funksiyasi nollardan iborat massiv, np.ones() funksiyasi birlardan iborat massiv, np.arange() funksiyasi ketma-ketlik, np.linspace() funksiyasi chiziqli oraliqda teng taqsimlangan sonlar massivini yaratadi. Tasodifiy sonlardan iborat massivlar uchun np.random moduli ishlatiladi, masalan, np.random.rand() yoki np.random.randn() funksiyalari. Identifikatsiya matritsasini yaratish uchun np.eye() funksiyasi qo‘llaniladi.

NumPy massivlari ustida turli matematik operatsiyalarni bajarish mumkin. Element bo‘yicha operatsiyalar juda tez bajariladi: qo‘shish (+), ayirish (-), ko‘paytirish (*), bo‘lish (/), daraja (** yoki np.power()), ildiz (np.sqrt()), logarifm (np.log()), trigonometrik funksiyalar (np.sin(), np.cos(), np.tan()) va boshqalar. Massivlar ustida statistik operatsiyalar ham mavjud: o‘rtacha qiymat (np.mean()), standart chetlanish (np.std()), maksimal va minimal qiymatlar (np.max(), np.min()), yig‘indi (np.sum()), ko‘paytma (np.prod()) va boshqalar. Matritsa operatsiyalari uchun np.dot() yoki @ operatori (matritsa ko‘paytirish), np.transpose() yoki .T atributi (transponirlash), np.linalg.inv() (teskari matritsani topish), np.linalg.det() (determinantni hisoblash) kabi funksiyalar mavjud.

NumPy massivlarini indekslash va kesish Python ro‘yxatlariga o‘xshash, lekin ko‘proq imkoniyatlarga ega. Bir o‘lchovli massivda elementga murojaat qilish uchun arr[0] yoki arr[-1] kabi indekslar ishlatiladi. Ko‘p o‘lchovli massivlarda vergul bilan ajratilgan indekslar ishlatiladi: arr[0, 1] yoki arr[1:3, 0:2]. Boolean indekslash ham qo‘llaniladi, masalan, arr[arr > 5] barcha 5 dan katta elementlarni qaytaradi. Fancy indexing (murakkab indekslash) yordamida massivga indekslar ro‘yxati yoki massivi berish mumkin: arr[[0, 2, 4]]. NumPy massivlarini reshape() funksiyasi yordamida qayta shakllantirish, flatten() yoki ravel() funksiyalari yordamida bir o‘lchovli massivga aylantirish, concatenate() yoki stack() funksiyalari yordamida birlashtirib qo‘yish mumkin.

NumPy kutubxonasi broadcasting (yoyish) mexanizmini qo‘llab-quvvatlaydi, bu turli shakldagi massivlar ustida arifmetik operatsiyalarni avtomatik ravishda bajarish imkonini beradi. Masalan, (3, 1) shakldagi massivni (3, 4) shakldagi massivga qo‘shganda, NumPy avtomatik ravishda kichikroq massivni kengaytiradi va operatsiyani bajaradi. Bu xususiyat kodni soddalashtirib, tsikllardan qochish imkonini beradi. Broadcasting qoidalari aniq: massivlar oxirgi o‘lchamlaridan boshlab taqqoslanadi va har bir o‘lcham yoki

teng bo‘lishi yoki ulardan biri 1 ga teng bo‘lishi kerak. Aks holda, xatolik yuzaga keladi.

Universal funksiyalar (ufuncs) NumPy kutubxonasining kuchli tomonlaridan biridir. Ufuncs massivlar ustida element bo‘yicha operatsiyalarni juda tez bajaradigan funksiyalardir. Ular vektorlashtirilgan operatsiyalarni qo‘llab-quvvatlaydi va Python tsikllarga qaraganda ancha tezroq ishlaydi. NumPy da 60 dan ortiq ufuncs mavjud bo‘lib, ularning aksariyati matematik operatsiyalar uchun mo‘ljallangan: np.add(), np.subtract(), np.multiply(), np.divide(), np.sin(), np.cos(), np.exp(), np.log() va boshqalar. Bundan tashqari, ufuncs taqqoslash operatsiyalarini ham qo‘llab-quvvatlaydi: np.greater(), np.less(), np.equal() va hokazo. Ufuncs bir nechta kirish massivlarini qabul qilishi va bir yoki bir nechta chiqish massivlarini qaytarishi mumkin.

NumPy kutubxonasida chiziqli algebra uchun numpy.linalg moduli mavjud bo‘lib, u matritsalarni yechish, xususiy qiymatlarni topish, QR va SVD dekompozitsiyasini bajarish kabi murakkab operatsiyalarni amalga oshiradi. np.linalg.solve() funksiyasi chiziqli tenglamalar tizimini yechadi, np.linalg.eig() xususiy qiymatlar va vektorlarni topadi, np.linalg.svd() singular value decomposition ni amalga oshiradi, np.linalg.qr() QR dekompozitsiyasini bajaradi. Bu funksiyalar ilmiy hisoblashlar, mashinali o‘rganish va signal qayta ishlash kabi sohalarda keng qo‘llaniladi. NumPy ning chiziqli algebra moduli LAPACK kutubxonasi asosida qurilgan bo‘lib, bu unga yuqori tezlik va ishonchlilikni ta'minlaydi.

```
import numpy as np
ballar = np.random.randint(50, 101, size=(5,4))
print("Tasodifiy ballar matritsasi:\n", ballar)

print("o‘lchami:", ballar.shape)
print("o‘lchamlar soni:", ballar.ndim)
print("Element turi:", ballar.dtype)
```

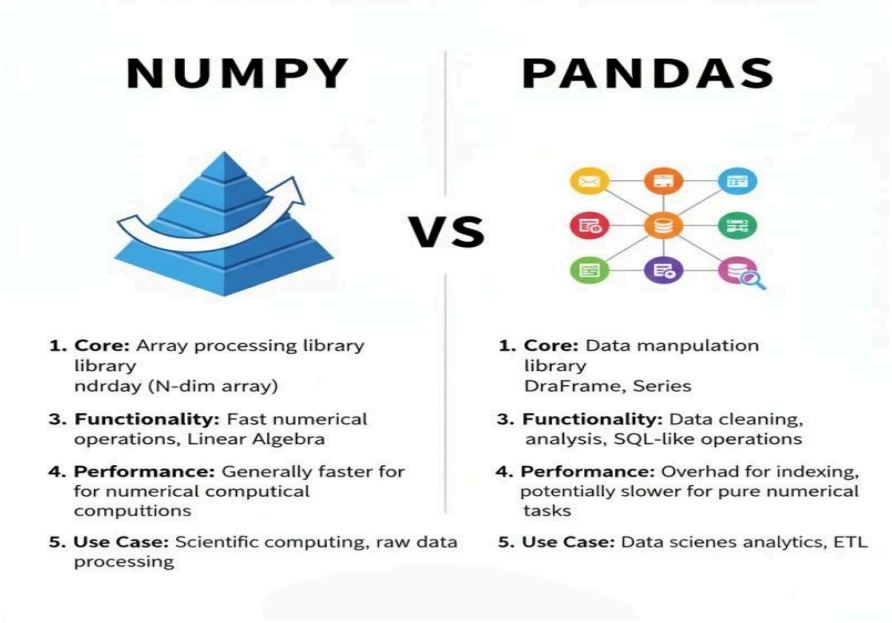
```
    print("Fanlar bo‘yicha o‘rtacha:", np.mean(ballar, axis=0))
    print("Talabalar bo‘yicha jami:", np.sum(ballar, axis=1))
    print("Eng kichik ball:", np.min(ballar))
    print("Eng katta ball:", np.max(ballar))
    print("Standart og‘ish:", np.std(ballar))
    print("Kvadrat ildiz:", np.sqrt(ballar))

    matritsa2 = np.array([[1,2,3,4],[5,6,7,8],[2,4,6,8],[1,3,5,7],[0,2,4,6]])
    print("Matritsa ko‘paytmasi:\n", np.dot(ballar, matritsa2.T))
    print("Transpozitsiya:\n", ballar.T)

    print("1D ko‘rinish:", ballar.flatten())
    print("Yangi shakl:", ballar.reshape(2,10))
```

NumPy funksiyalari vazifalari: Ushbu kutubxonada ishlatilgan funksiyalar sonli ma’lumotlar va matritsa ustida turli amallarni bajarish uchun mo‘ljallangan. **np.array()** funksiyasi oddiy Python ro‘yxatlaridan ko‘p o‘lchamli massiv yaratadi va u bilan tezkor hisob-kitoblarni bajarish imkonini beradi. np.random.randint() funksiyasi orqali tasodifiy sonlar generatsiya qilinadi, bu esa test ma’lumotlarini yaratishda foydali. Massiv tuzilishi haqida ma’lumot olish uchun shape, ndim va dtype atributlari ishlatiladi; ular massiv o‘lchami, o‘lchamlar soni va element turi haqida ma’lumot beradi. Matematik va statistik hisoblar uchun **mean(), sum()**, **min()**, **max()**, **std()**, **sqrt()** kabi funksiyalar qo‘llaniladi — ular o‘rtacha qiymat, yig‘indi, eng kichik va eng katta qiymat, standart og‘ish va kvadrat ildizni hisoblash imkonini beradi. Matritsa amallari uchun esa **dot()** matritsa ko‘paytmasini, T transpozitsiyani bajaradi, **flatten()** va **reshape()** funksiyalari orqali massiv shaklini o‘zgartirish mumkin. Shu bilan NumPy kutubxonasi sonli ma’lumotlarni tez va aniq tahlil qilish, ilmiy hisob-kitoblarni bajarish va matritsa operatsiyalarini amalga oshirish uchun asosiy vosita hisoblanadi.

Pandas kutubxonasi va ma'lumotlar tahlili

Pandas kutubxonasi ikki asosiy ma'lumot tuzilmasini taqdim etadi: Series va DataFrame. Series - bu bir o‘lchovli indekslangan massiv bo‘lib, har qanday turdagi ma'lumotlarni saqlashi mumkin (butun sonlar, suzuvchi nuqtali sonlar, satrlar, Python ob'ektlari va hokazo). Series ob'ekti ikki asosiy komponentdan iborat: qiymatlar massivi va indekslar massivi. Indekslar avtomatik ravishda 0 dan boshlanadigan butun sonlar bo‘lishi mumkin yoki foydalanuvchi tomonidan belgilanishi mumkin. Series ob'ektlari ustida NumPy massivlariga o‘xshash operatsiyalarni bajarish mumkin, lekin qo‘shimcha imkoniyatlar ham mavjud, masalan, nomlangan indekslar yordamida elementlarga murojaat qilish.

DataFrame - bu ikki o‘lchovli, o‘zgaruvchan o‘lchamli, potensial turli xil turdagi ustunli jadval ma'lumot tuzilmasidir. DataFrame ni SQL jadvali yoki Excel elektron jadvali sifatida tasavvur qilish mumkin. Har bir ustun Series ob'ekti hisoblanadi va barcha ustunlar umumiy indeksni bo‘lishadi. DataFrame yaratishning ko‘plab usullari mavjud: lug‘atdan, NumPy massividan, boshqa DataFrame dan, CSV yoki Excel faylidan va boshqalar. DataFrame ob'ekti ma'lumotlarni ko‘rish, filtrlash, guruhlash, birlashtrish, pivot qilish va boshqa ko‘plab operatsiyalarni bajarish imkonini beradi.



2-rasm.NumPy va Pandas asosiy farqlar.

Pandas kutubxonasini o‘rnatish ham pip yordamida amalga oshiriladi: 'pip install pandas'. Import qilish uchun standart qisqartma ishlatiladi: 'import pandas as pd'. Pandas kutubxonasi NumPy ga bog‘liq bo‘lib, ichki tuzilmalarda NumPy massivlaridan foydalanadi. Pandas kutubxonasi ham doimiy ravishda yangilanib turadi va yangi versiyalarda ishlash tezligi oshiriladi, yangi funksiyalar qo‘shiladi va foydalanish qulayligi yaxshilanadi. Hozirgi kunda Pandas versiya 2.2 gacha rivojlangan va ma'lumotlar tahlili sohasida eng mashhur Python kutubxonasi hisoblanadi. Pandas kutubxonasi katta hajmdagi ma'lumotlar bilan ishlash, ma'lumotlarni tozalash va tayyorlash, statistik tahlil va vizualizatsiya uchun juda qulay vosita hisoblanadi.

Pandas kutubxonasi turli formatlardagi fayllardan ma'lumotlarni yuklash va saqlash uchun keng imkoniyatlar taqdim etadi. Eng ko‘p ishlatiladigan format CSV (Comma-Separated Values) hisoblanadi. CSV faylni yuklash uchun pd.read_csv() funksiyasi ishlatiladi. Bu funksiya ko‘plab parametrlarga ega: separator, header, names, index_col, usecols, dtype, parse_dates va boshqalar. Excel fayllarni yuklash uchun pd.read_excel() funksiyasi mavjud bo‘lib, u .xls va .xlsx formatlarini qo‘llab-quvvatlaydi. JSON formatdagi ma'lumotlarni yuklash uchun pd.read_json() funksiyasi ishlatiladi. SQL ma'lumotlar bazasidan ma'lumotlarni yuklash uchun pd.read_sql() yoki pd.read_sql_query() funksiyalari mavjud.

Ma'lumotlarni saqlash ham xuddi shunday oson. DataFrame ob'ektini CSV faylga saqlash uchun to_csv() metodi, Excel faylga saqlash uchun to_excel() metodi, JSON formatda saqlash uchun to_json() metodi ishlatiladi. Bundan tashqari, Pandas HDF5, Parquet, Feather kabi samarali formatlarni ham qo‘llab-quvvatlaydi. HDF5 format katta hajmdagi ma'lumotlarni saqlash uchun juda mos keladi va tez o‘qish-yozish tezligini ta'minlaydi. Parquet format ustunli saqlash formatida bo‘lib, siqilgan holda saqlash va tez o‘qish imkonini beradi. Feather format Pandas va Apache Arrow o‘rtasida ma'lumotlarni almashish uchun mo‘ljallangan va juda tez ishlaydi.

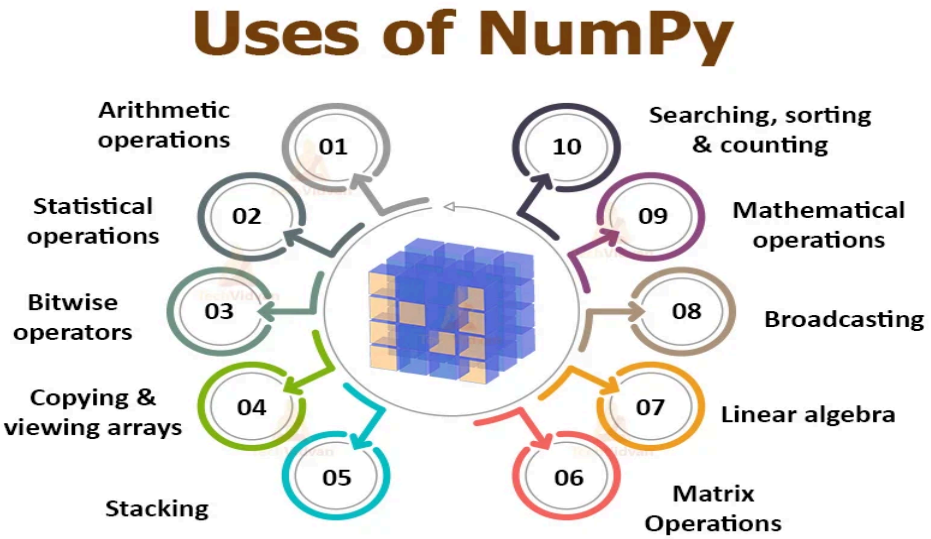
Ma'lumotlarni yuklashda xatolar yuz berishi mumkin: noto‘g‘ri fayl yo‘li, noto‘g‘ri format, noto‘g‘ri kodlash (encoding) va boshqalar. Pandas bu xatolarni boshqarish uchun turli parametrlar taqdim etadi. Masalan, encoding parametri fayl kodlashini belgilaydi (utf-8, latin-1, cp1251 va boshqalar), error_bad_lines parametri noto‘g‘ri qatorlarni o‘tkazib yuborish imkonini beradi, na_values parametri qaysi qiymatlar NULL deb hisoblanishini belgilaydi. Katta fayllarni yuklashda xotira muammolari yuzaga kelishi mumkin, buning uchun chunksize parametridan foydalanib, faylni qismlar bo‘yicha yuklash mumkin. Bu usul katta hajmdagi ma'lumotlarni qayta ishlashda juda foydali.

Real ma'lumotlar odatda noperfekt bo‘ladi: yetishmayotgan qiymatlar, takroriy qatorlar, noto‘g‘ri formatlar, chetlanmalar (outliers) va boshqa muammolar mavjud bo‘ladi. Pandas kutubxonasi bu muammolarni hal qilish uchun ko‘plab vositalar taqdim etadi. Yetishmayotgan qiymatlarni aniqlash uchun isnull() yoki isna() metodlari ishlatiladi, ular Boolean DataFrame qaytaradi. Yetishmayotgan qiymatlarni to‘ldirish uchun fillna() metodi ishlatiladi, bu metod qiymat, metod (forward fill, backward fill) yoki interpolatsiya usulini qabul qiladi. Yetishmayotgan qiymatlarni o‘chirish uchun dropna() metodi mavjud bo‘lib, u qatorlar yoki ustunlarni o‘chirishi mumkin.

Takroriy qatorlarni topish va o‘chirish ham muhim vazifa hisoblanadi. duplicated() metodi takroriy qatorlarni topadi va Boolean Series qaytaradi. drop_duplicates() metodi takroriy qatorlarni o‘chiradi va yangilangan DataFrame qaytaradi. Ma'lumot turlarini o‘zgartirish uchun astype() metodi ishlatiladi, bu metod ustun yoki butun DataFrame ning ma'lumot turini o‘zgartiradi. Satrlarni raqamlarga, sanalarga yoki boshqa turlarga aylantirish uchun pd.to_numeric(), pd.to_datetime(), pd.to_timedelta() funksiyalari mavjud. Bu funksiyalar xatolarni boshqarish imkonini beradi: e'tibor bermaslik, NaT (Not a Time) yoki xato chiqarish.

Ma'lumotlarni transformatsiya qilish ham tahlilning muhim qismi hisoblanadi. apply() metodi har bir element, qator yoki ustun uchun funksiyani

qo‘llash imkonini beradi. map() metodi Series uchun ishlatiladi va har bir qiymatni boshqa qiymatga moslashtiradi. replace() metodi ma'lum qiymatlarni boshqa qiymatlar bilan almashtiradi. Satrlar bilan ishlash uchun str aksesori mavjud bo‘lib, u satrlar ustida turli operatsiyalarni bajarish imkonini beradi: lower(), upper(), strip(), split(), contains() va boshqalar. Sana va vaqt bilan ishlash uchun dt aksesori mavjud bo‘lib, u yil, oy, kun, soat, daqiqa va boshqa komponentlarni ajratib olish imkonini beradi.



3-rasm. NumPy asosiy foydalanish funksiyalari.

NumPy va Pandas kutubxonalarining amaliy qo‘llanilishi

Pandas kutubxonasi ma'lumotlarni tahlil qilish uchun kuchli vositalar to‘plamini taqdim etadi. describe() metodi DataFrame ning asosiy statistik xususiyatlarini ko‘rsatadi: count (qiymatlar soni), mean (o‘rtacha qiymat), std (standart chetlanish), min (minimal qiymat), 25%, 50%, 75% percentillar va max (maksimal qiymat). value_counts() metodi Series ning har bir noyob qiymatining takrorlanish sonini ko‘rsatadi. groupby() metodi ma'lumotlarni guruhlash va har bir guruhda agregatsiya operatsiyalarini bajarish imkonini beradi. Masalan, ma'lumotlarni kategoriya bo‘yicha guruhlash va har bir kategoriyada o‘rtacha qiymatni hisoblash mumkin.

Pivot jadvallar ma'lumotlarni qayta tuzish va tahlil qilish uchun juda qulay vosita hisoblanadi. pivot_table() metodi ma'lumotlarni ko‘p o‘lchovli jadval shaklida tuzadi, bu yerda qatorlar, ustunlar va qiymatlar turli ustunlardan olinadi. Agregatsiya funksiyalari (sum, mean, count, min, max va boshqalar) qo‘llanilishi mumkin. pivot_table() metodi Excel ning pivot jadvallariga o‘xshash, lekin ko‘proq imkoniyatlarga ega. crosstab() funksiyasi ikki yoki undan ortiq ustunlar orasidagi chastota jadvalini yaratadi, bu kategorik ma'lumotlarni tahlil qilishda juda foydali.

Ma'lumotlarni vizualizatsiya qilish uchun Pandas Matplotlib kutubxonasi bilan integratsiyalangan. DataFrame va Series ob'ektlarida plot() metodi mavjud bo‘lib, u turli xil grafiklarni yaratish imkonini beradi: chiziqli grafik (line), ustunli diagram (bar), gistogramma (hist), tarqalish diagrammasi (scatter), quti diagrammasi (box) va boshqalar. plot() metodiga kind parametri orqali grafik turini belgilash mumkin.

```
import pandas as pd
import numpy as np
data = {
    'Ism': ['Ali', 'Vali', 'Hasan', 'Sara', 'Nodir'],
    'Matematika': [80, 85, 78, 90, 82],
    'Informatika': [90, 88, 92, 85, 87],
    'Fizika': [70, 75, 80, 85, 78],
    'Kimyo‘: [88, 82, 85, 90, 80]
}

df = pd.DataFrame(data)

print(df.head())
print(df.info())
print(df.describe())

print("Fanlar bo‘yicha o‘rtacha:\n", df[['Matematika','Informatika','Fizika','Kimyo‘]].mean())
print("Talabalar bo‘yicha jami ball:\n", df[['Matematika','Informatika','Fizika','Kimyo‘]].sum(axis=1))
```

```
    print("Eng katta va eng kichik ball:\n", df[['Matematika','Informatika','Fizika','Kimyo‘]].max(), df[['Matematika','Informatika','Fizika','Kimyo‘]].min())

    print("Matematika bo‘yicha tartiblangan jadval:\n", df.sort_values(by='Matematika', ascending=False))
    print("Matematika 85 dan katta bo‘lgan talabalar:\n", df[df['Matematika']>85])

    print("Bo‘sh qiymatlar mavjudmi?\n", df.isnull())
    df.fillna(0, inplace=True)
```

Pandas funksiyalari vazifalari: Pandas kutubxonasida ishlatilgan funksiyalar jadval ko‘rinishidagi ma’lumotlarni yaratish, tahlil qilish va boshqarish uchun ishlatiladi. **DataFrame()** funksiyasi yordamida ma’lumotlar jadval shakliga keltiriladi, ustun va indekslar nomlanadi, bu esa ma’lumotlarni oson boshqarish imkonini beradi. Jadvalni dastlabki ko‘rinishda tekshirish uchun **head()** va **info()** funksiyalari ishlatiladi, ular qatorlar, ustunlar va ma’lumot turi haqida umumiy ma’lumot beradi, **describe()** esa statistik xulosa chiqaradi. Ma’lumotlarni tahlil qilish jarayonida **mean()**, **sum()**, **max()**, **min()** funksiyalari yordamida fanlar bo‘yicha o‘rtacha, talabalar bo‘yicha jami va eng katta/eng kichik balllar hisoblanadi. Ma’lumotlarni tartiblash va filtrlash uchun **sort_values()** va shartli tanlash **(df[df['Matematika']>85])** ishlatiladi. Bo‘sh qiymatlarni aniqlash va to‘ldirish uchun esa isnull() va fillna() funksiyalari qo‘llanadi. Shu tarzda Pandas kutubxonasi ma’lumotlar bilan ishlash, tahlil qilish va statistik xulosalar chiqarish uchun qulay va samarali vosita hisoblanadi.

NumPy kutubxonasi ilmiy hisoblashlarning asosi hisoblanadi. Ko‘plab ilmiy Python kutubxonalari NumPy massivlari ustida qurilgan: SciPy (ilmiy hisoblashlar), Matplotlib (vizualizatsiya), Scikit-learn (mashinali o‘rganish), TensorFlow va PyTorch (chuqur o‘rganish) va boshqalar. NumPy massivlari xotiradan samarali foydalanadi va tez ishlaydi, bu esa katta hajmdagi ma'lumotlar va murakkab hisoblashlar bilan ishlashda juda muhim. NumPy

Fourier transformatsiyasi, tasodifiy sonlar generatsiyasi, polinomial operatsiyalar, chiziqli algebra va boshqa ko‘plab ilmiy hisoblashlar uchun funksiyalar taqdim etadi.

Mashinali o‘rganishda Pandas va NumPy kutubxonalari asosiy rol o‘ynaydi. Ma'lumotlarni tayyorlash (data preprocessing) bosqichida Pandas yordamida ma'lumotlar yuklanadi, tozalanadi, transformatsiya qilinadi va tahlil qilinadi. NumPy massivlari esa mashinali o‘rganish algoritmlariga kirish ma'lumotlari sifatida beriladi. Scikit-learn kutubxonasi NumPy massivlari bilan ishlaydi va turli xil mashinali o‘rganish algoritmlarini taqdim etadi: klassifikatsiya, regressiya, klasterlash, o‘lchamlarni kamaytirish va boshqalar. Feature engineering (xususiyatlarni yaratish) jarayonida ham Pandas va NumPy kutubxonalaridan keng foydalaniladi.

Vaqt seriyalari tahlili (time series analysis) Pandas kutubxonasining kuchli tomonlaridan biridir. Pandas sana va vaqt bilan ishlash uchun maxsus ma'lumot turlari va funksiyalar taqdim etadi. DatetimeIndex yordamida vaqt bo‘yicha indekslangan ma'lumotlar yaratish mumkin. resample() metodi vaqt seriyalarini qayta namuna olish imkonini beradi, masalan, kunlik ma'lumotlarni haftalik yoki oylik ma'lumotlarga aylantirish. rolling() metodi sirg‘alib o‘rtacha (rolling average) va boshqa statistik ko‘rsatkichlarni hisoblash imkonini beradi. shift() metodi ma'lumotlarni vaqt bo‘yicha siljitadi, bu lag xususiyatlarini yaratishda foydali. Pandas vaqt zonalari bilan ham ishlaydi va turli vaqt zonalari o‘rtasida konvertatsiya qilish mumkin.

Katta hajmdagi ma'lumotlar bilan ishlashda xotira va tezlik muammolari yuzaga kelishi mumkin. Pandas kutubxonasi bu muammolarni hal qilish uchun bir nechta strategiyalarni taqdim etadi. Birinchidan, ma'lumot turlarini optimallashtirish: float64 o‘rniga float32 ishlatish, int64 o‘rniga int32 yoki int16 ishlatish xotirani sezilarli darajada kamaytiradi. category ma'lumot turi takrorlanuvchi satrlar uchun juda samarali. Ikkinchidan, fayllarni qismlar bo‘yicha yuklash: chunksize parametri yordamida katta faylni kichik qismlarga

bo‘lib yuklash va qayta ishlash mumkin. Bu usul xotiradan samarali foydalanishga yordam beradi.

Parallel ishlov berish ham ishlash tezligini oshirish uchun muhim. Pandas ba'zi operatsiyalarni parallel ravishda bajarishi mumkin, lekin to‘liq parallel qo‘llab-quvvatlash yo‘q. Buning uchun Dask kutubxonasidan foydalanish tavsiya etiladi. Dask Pandas API ga o‘xshash interfeys taqdim etadi, lekin katta hajmdagi ma'lumotlarni parallel ravishda qayta ishlaydi. Dask ma'lumotlarni qismlarga bo‘lib, har bir qismni alohida protsessorda qayta ishlaydi va natijalarni birlashtirib beradi. Bu usul katta hajmdagi ma'lumotlarni tezroq qayta ishlash imkonini beradi. Bundan tashqari, GPU dan foydalanish uchun cuDF kutubxonasi mavjud bo‘lib, u Pandas API ga o‘xshash, lekin NVIDIA GPU da ishlaydi va juda yuqori tezlikni ta'minlaydi.

Xotira iste'molini kamaytirish uchun zarur bo‘lmagan ustunlarni o‘chirish, faqat kerakli ustunlarni yuklash (usecols parametri), ma'lumotlarni siqilgan formatda saqlash (Parquet, HDF5) kabi usullar qo‘llaniladi. Indekslarni optimallashtirish ham muhim: kamroq foydalaniladigan indekslarni reset_index() yordamida o‘chirish, ko‘p indeksli (multi-index) strukturalarni ehtiyotkorlik bilan ishlatish tavsiya etiladi. Query operatsiyalarini optimallashtirish uchun query() metodidan foydalanish kerak, bu metod satrli ifodani qabul qiladi va tezroq ishlaydi. Masalan, df.query('age > 30 and salary < 50000') ifodasi df[(df['age'] > 30) & (df['salary'] < 50000)] ifodaga qaraganda tezroq ishlaydi.

Foydalanilgan adabiyotlar

1. Hastie, T., Tibshirani, R., Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer.
2. Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer.
3. Géron, A. (2019). *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*. o'Reilly Media.
4. Murphy, K. P. (2012). *Machine Learning: A Probabilistic Perspective*. MIT Press.
5. Russell, S., Norvig, P. (2020). *Artificial Intelligence: A Modern Approach* (4th ed.). Pearson.
6. Goodfellow, I., Bengio, Y., Courville, A. (2016). *Deep Learning*. MIT Press.
7. Sayood, K. (2017). *Introduction to Data Compression* (5th ed.). Morgan Kaufmann.
8. Schneier, B. (2015). *Applied Cryptography: Protocols, Algorithms, and Source Code in C*. Wiley.