Python: Malumot turlari, operations

Malumot turlari: data types:

(integer)
Int => Sonoq sonlar manfiy va musbat butun sonlar

Float => qolduqli sonlar

(string)
Str => (object) => " " ' ' orasidagi so'zlar

Boolean => True, False.

Operation.

Math (+, -, *.

Bolish /, //, %.) ** => daraja

/ => to'liq bo'linmani ko'rsatadi. 4/3 = 1.333

// => bo'lib faqat butun qismini chiqarib beradi.

4/3 = 1.

% => sonlarni o'zaro bo'lib ortib qolgan qoldiq qismini chiqarib beradi.

4/3 = (1) => ortib qolgan qismi.

** => daraja (pow) formulasidan ham foydalanse bo'ladi

4^3 => 64.

Logic operation. (and ; or)

and => 2ta toman ham to'g'ri bo'lishi kk.
or => 1ta yoki 2ta toman ham to'g'ri bo'lishi kerak.

Comparative operations.

>, <, (==) (!=).

Indexing.

So'zdagi harflarning joylashuvini ko'rsatib beradi.

Hello
0 1 2 3 4
-5 -4 -3 -2 -1

Pythonda data setni yuklash va data set bn tanishuv

pip install pandas ①

import pandas as pd ②

(data =) (pd =) df = pd. read. csv (your file name) ③

VCS ochilib yangi file ochiladi va unda pandas yuklab olinadi keyin file bn bir joyda data set saqlanadi, colon ketma-ket terishi kk. pandasda óqishi uchun

Agar file qoʻyilishida muammo bólsa:

yangi VSC da yangi file ochilib uni desktopga saqlanib qilinadi. Sóngra data setni ózini usha filega óttaziladi. keyin VSC ichida qayta usha file ochiladi. keyin qayta ③ bosqich takrorlanadi.

File import bólgandan keyin:

*) df info () ⇒ data setdagi ma'lumotlarni kórish uchun

*) df head () ⇒ data setdagi boshidagi 5ta ~~[illegible]~~ni ustun ma'lumotlarni chiqarib beradi.

*) df.isnull().sum() => tushib qoldirilgan ma'lumotlar qaysi qatorda, nechtadanligi haqidagi ma'lumotlarni olish uchun.
(integer) shaklda chiqarib beradi

*) df.isnull().any() => tushib qoldirilgan ma'lumotlarni (boolean) shaklida chiqarib beradi.

*) df.tail(5) => data setning oxiridan 5 ta qator malumotlarni ko'rsatadi.

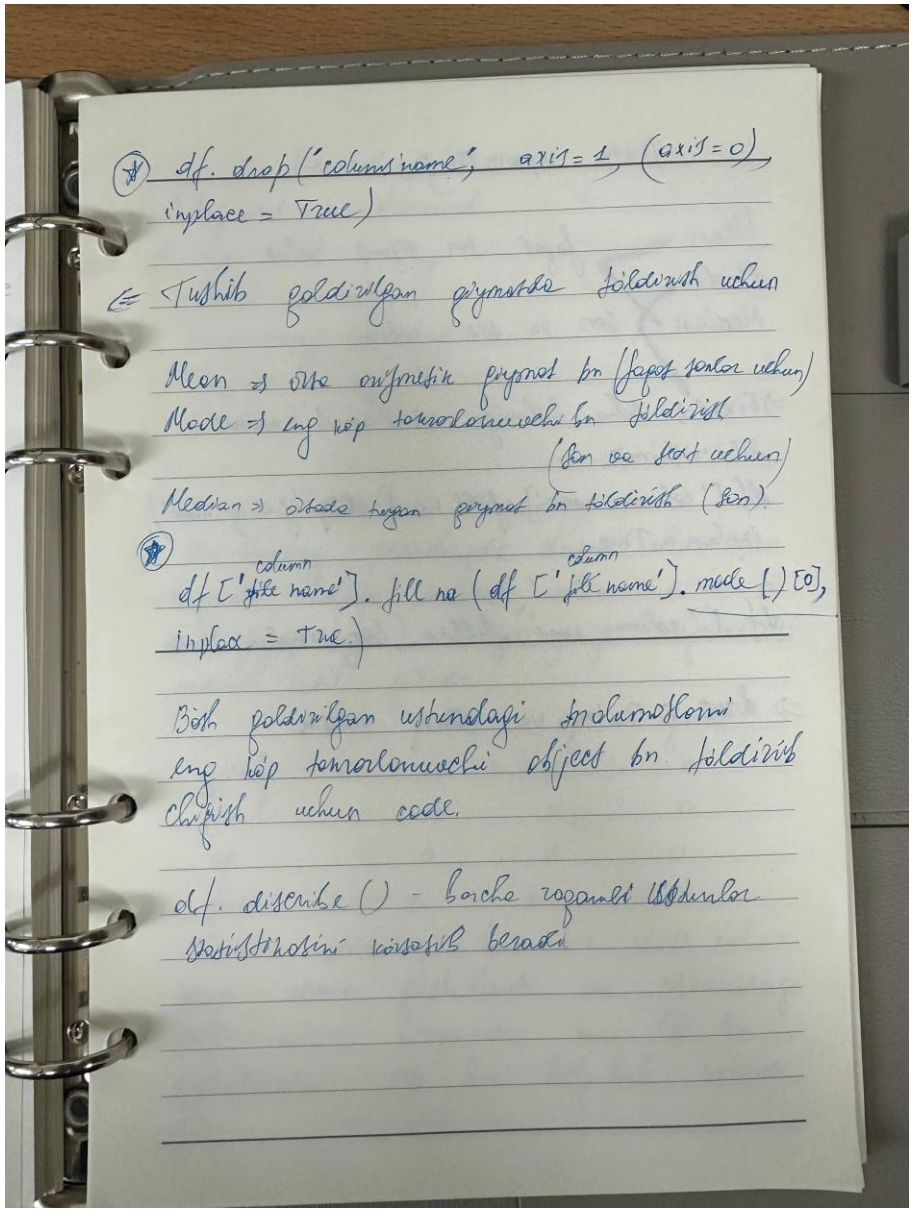
*) df.nunique() har bir ustundagi klasslar sonini ko'rsatib beradi

=> Data preprocessing.

Bu bosqichda tushib qoldirilgan malumotlar to'ldiriladi yoki keraksiz deb hisoblanganlari tushib qoldiriladi.

*) Keraksiz deb topilgan ustunni tushib qoldirish.

df.drop('columns name')



df.drop('column name', axis=1, (axis=0), inplace = True)

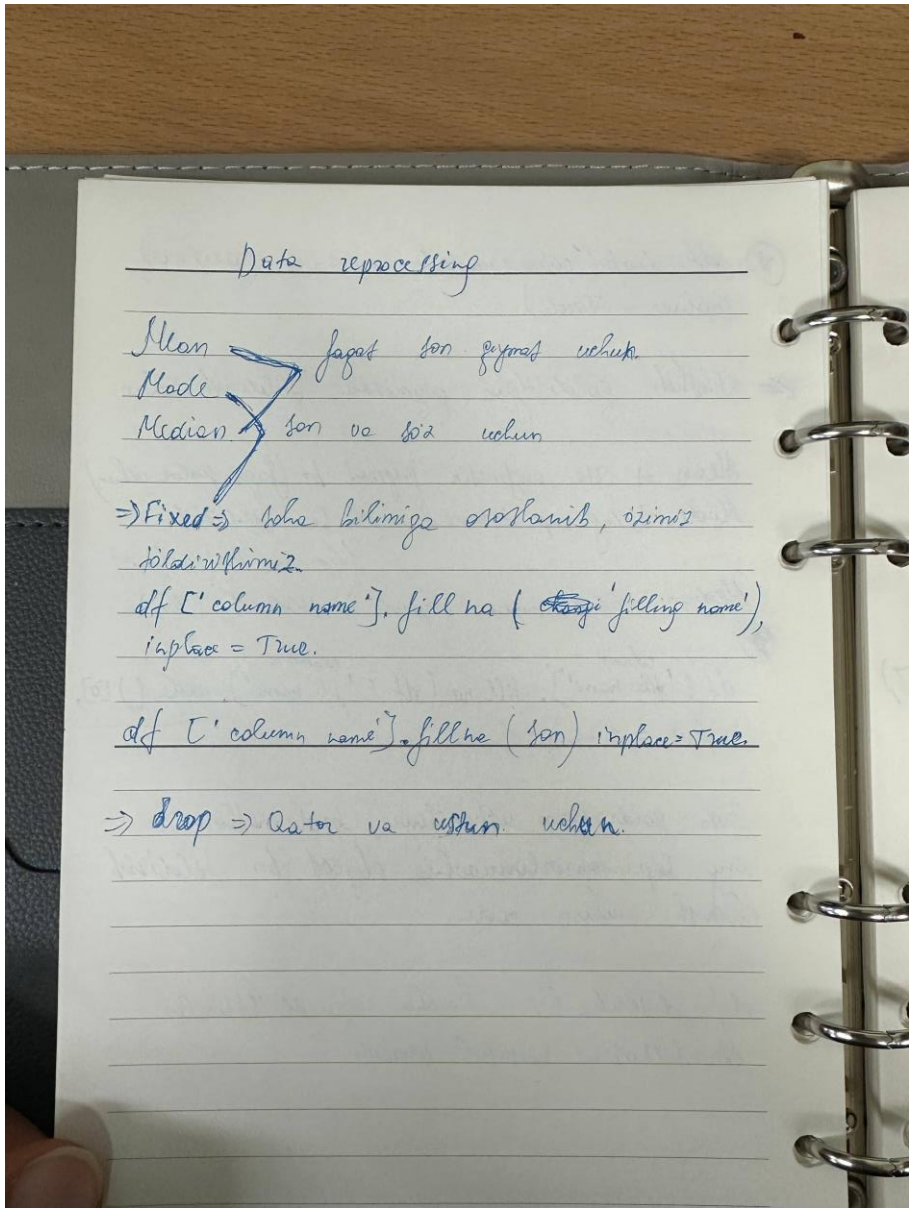
⇐ Tushib qoldirilgan qiymatlar jadvalni uchun

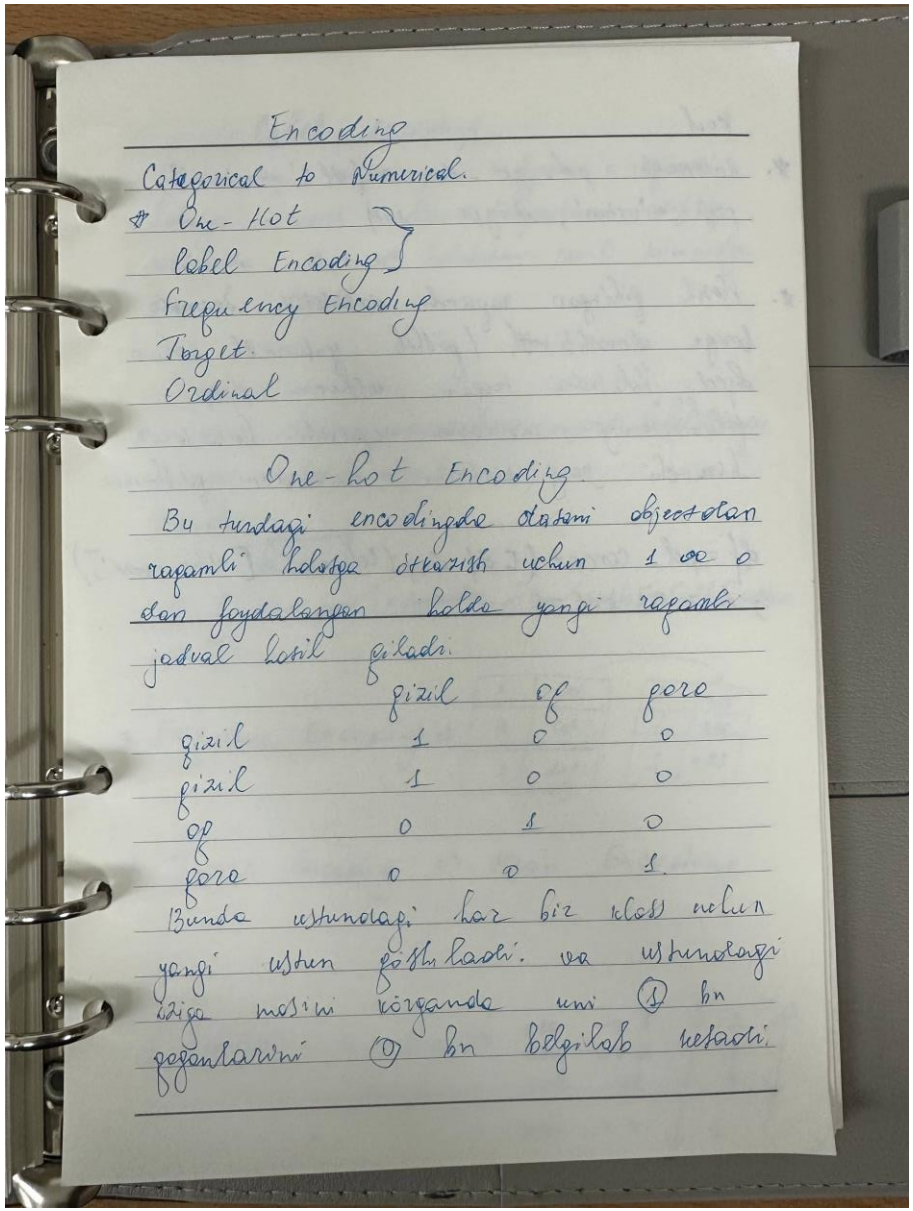
Mean ⇒ o'rta arifmetik qiymat bn (faqat sonlar uchun)
Mode ⇒ eng ko'p takrorlanuvchi bn jadvalni (ham va text uchun)
Median ⇒ o'rtada turgan qiymat bn ishlaydi (son)

df['column name'].fillna(df['column name'].mode()[0], inplace = True)

Bitta qoldirilgan ustundagi barcha nomalumlarni eng ko'p takrorlanuvchi object bn jadvalni chiqish uchun code.

df.describe() - barcha raqamli ustunlar statistikasini ko'rsatib beradi


Data reprocessing
Mean
Mode
Median
jagat son qiymat uchun
son va so'z uchun
=> Fixed => bitta bilimga asoslanib, o'zimiz
to'ldiramiz.
df['column name'].fillna('filling name'),
inplace = True.
df['column name'].fillna(son) inplace=True
=> drop => Qator va ustun uchun.



Encoding

Categorical to Numerical.

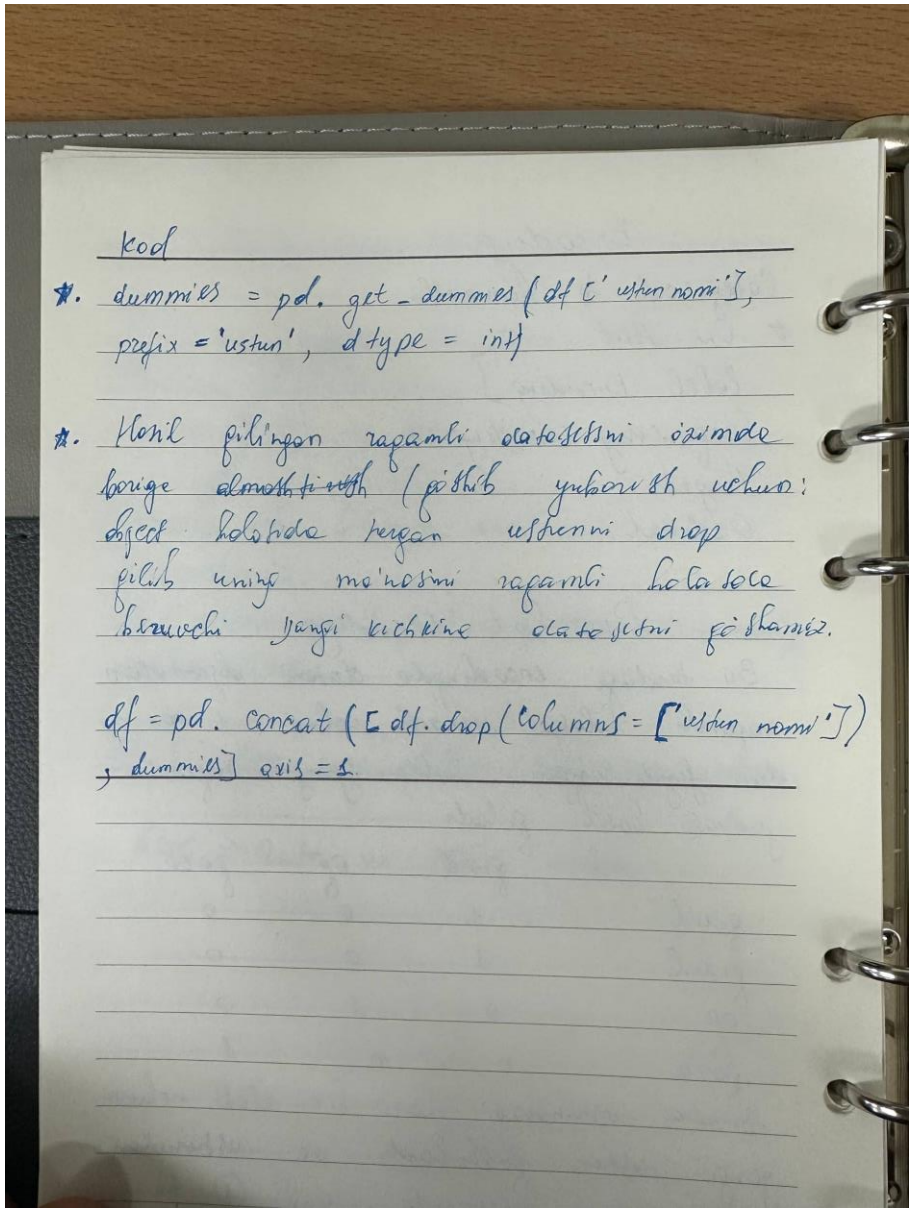
* One-Hot
  Label Encoding
  Frequency Encoding
  Target
  Ordinal

One-hot Encoding.

Bu turdagi encodingda datani objectdan raqamli holatga o'tkazish uchun 1 va 0 dan foydalangan holda yangi raqamli jadval hosil qiladi.

	gizil	of	gora
gizil	1	0	0
gizil	1	0	0
of	0	1	0
gora	0	0	1

Bunda ustunlagi har bir klass uchun yangi ustun qo'shiladi. va ustunlagi kiriga mosini ko'rganda uni (1) bn qolganlarini (0) bn belgilab ketadi.

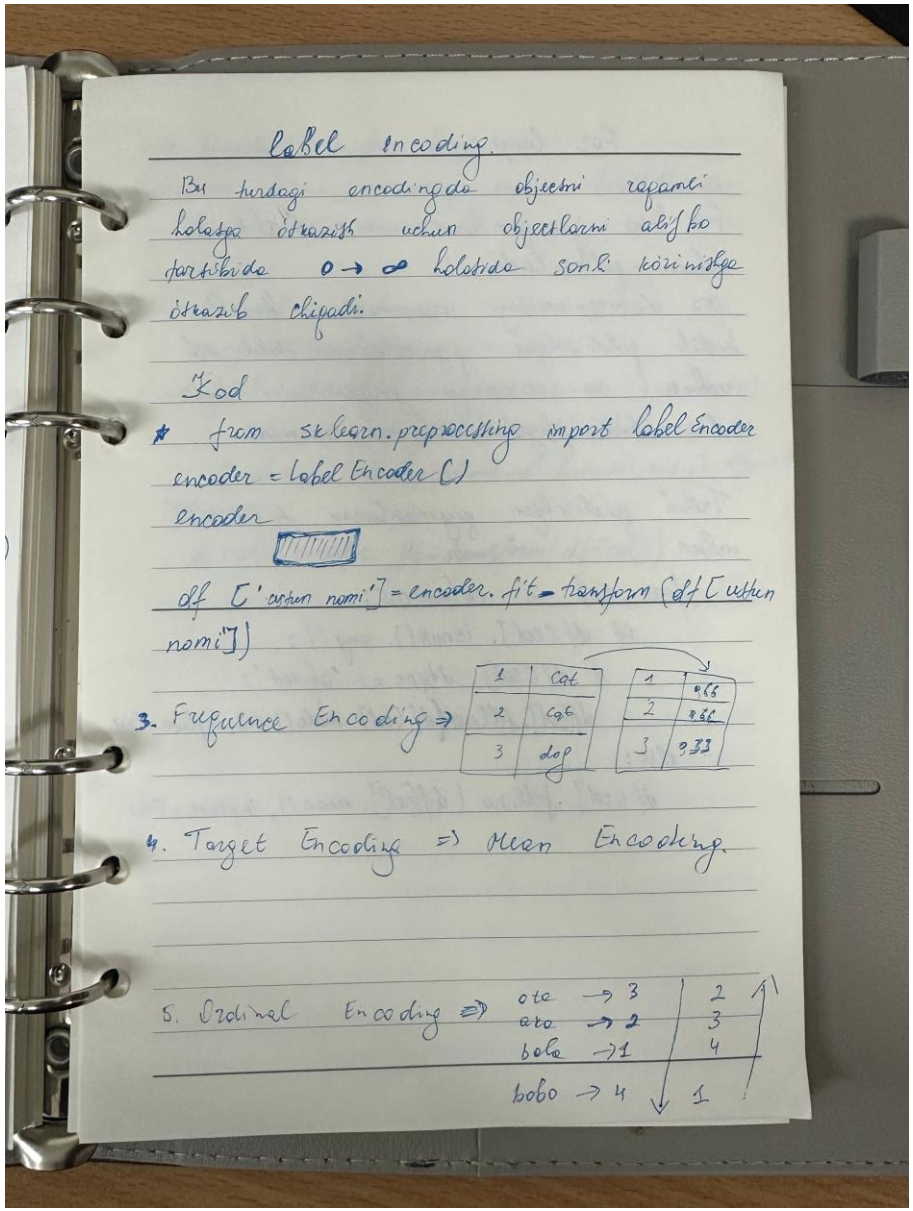


kod

*. dummies = pd. get_dummies(df['ustun nomi'], prefix='ustun', dtype=int)

*. Hosil bo'lingan raqamli ustunlarni o'zimiz berilgan ~~almashtirish~~ (pastda yuqorida uchun) object holatida turgan ustunni drop qilib uning mo'nosini raqamli ko'rinishdagi yangi ustunlarni qo'shamiz.

df = pd. concat([df. drop(columns=['ustun nomi']), dummies], axis=1.



Label Encoding.

Bu turdagi encodingda obyektni raqamli holatga o'tkazish uchun obyektlarni alifbo tartibida 0 → ∞ holatida sonli ko'rinishga o'tkazib chiqadi.

Kod

* from sklearn.preprocessing import LabelEncoder
encoder = LabelEncoder()
encoder

df['ustun nomi'] = encoder.fit_transform(df['ustun nomi'])

3. Frequence Encoding ⇒

1	cat
2	cat
3	dog

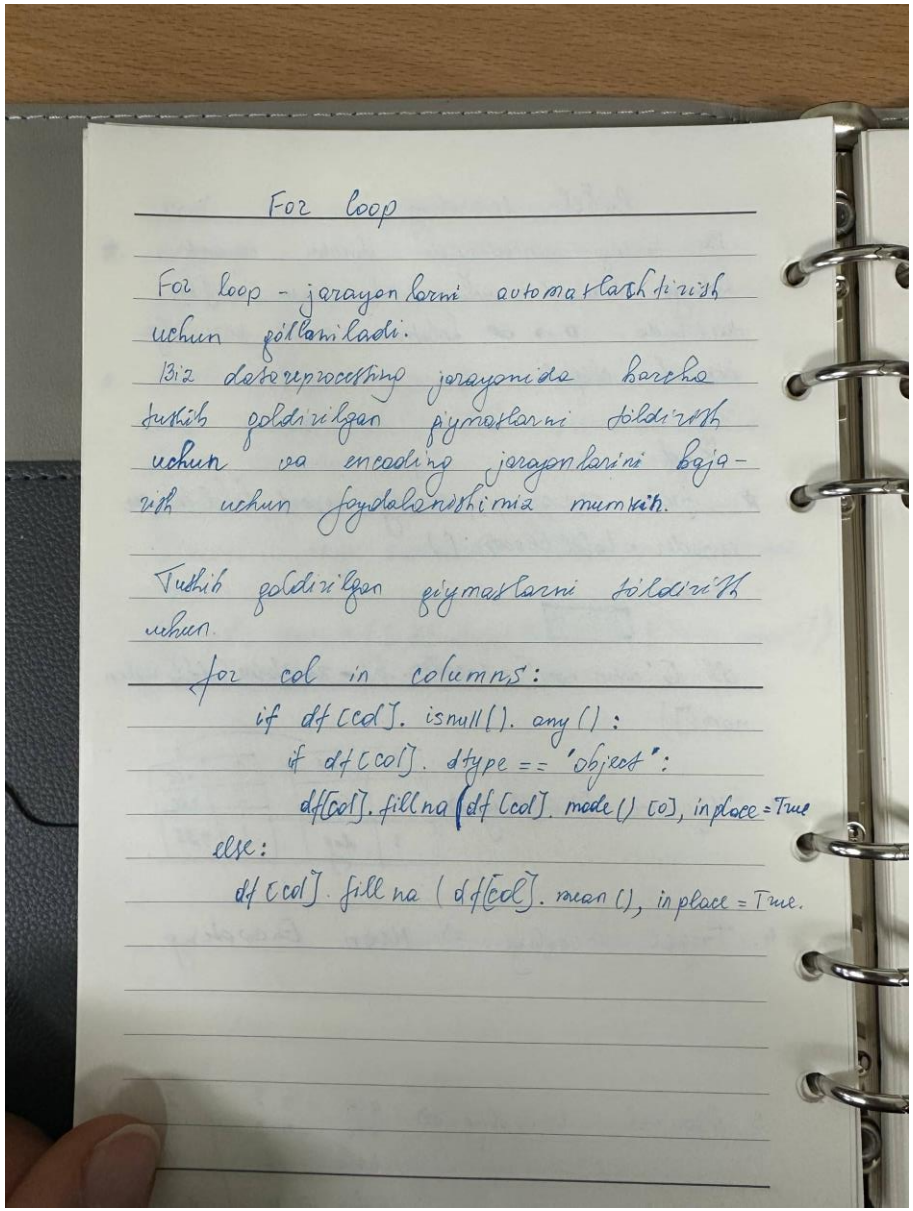
1	0.66
2	0.66
3	0.33

4. Target Encoding ⇒ Mean Encoding.

5. Ordinal Encoding ⇒

ota → 3
ota → 2
bola → 1
bobo → 4

2
3
4
1

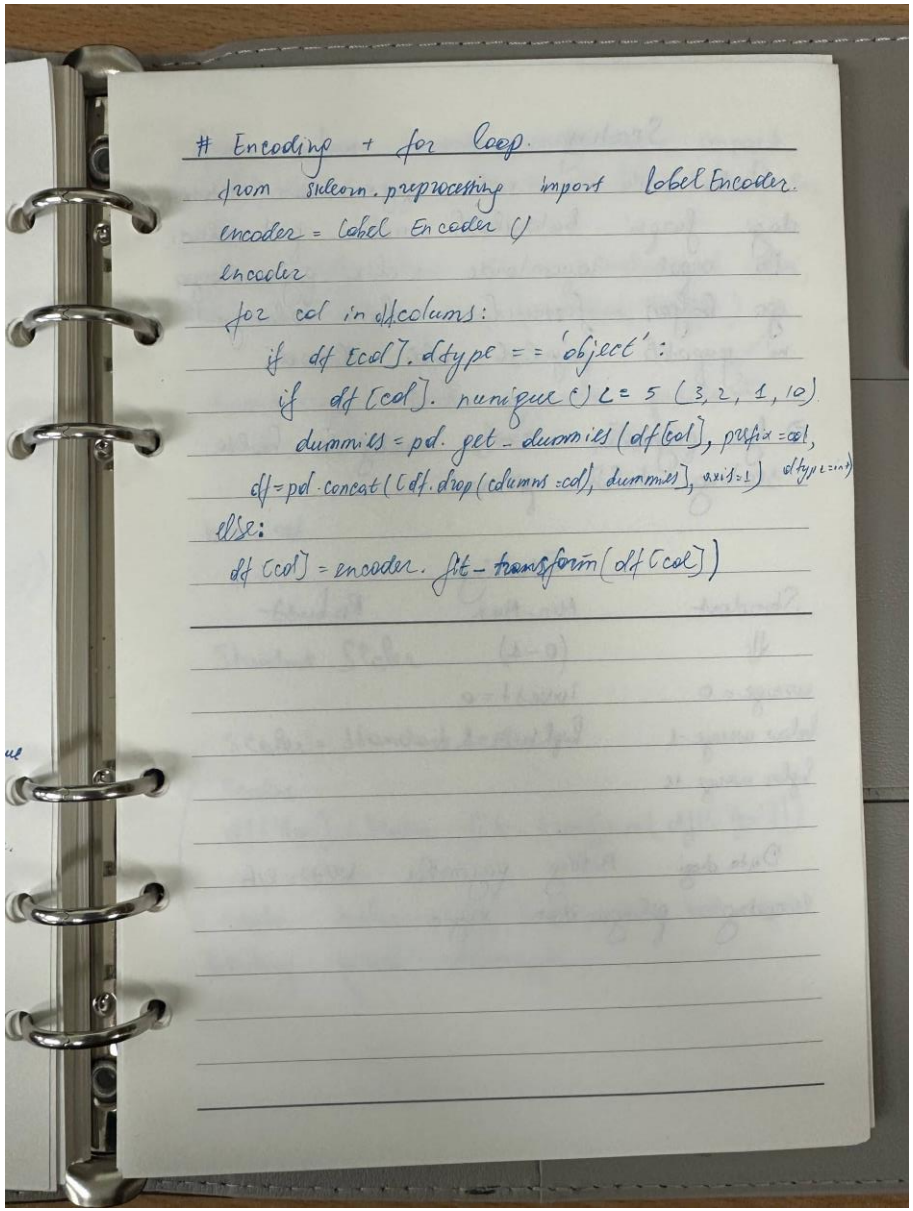


For loop

For loop - jarayonlarni avtomatlashtirish uchun qo'llaniladi.
Biz data preprocessing jarayonida barcha turlib qoldirilgan qiymatlarni to'ldirish uchun va encoding jarayonlarini bajarish uchun foydalanishimiz mumkin.

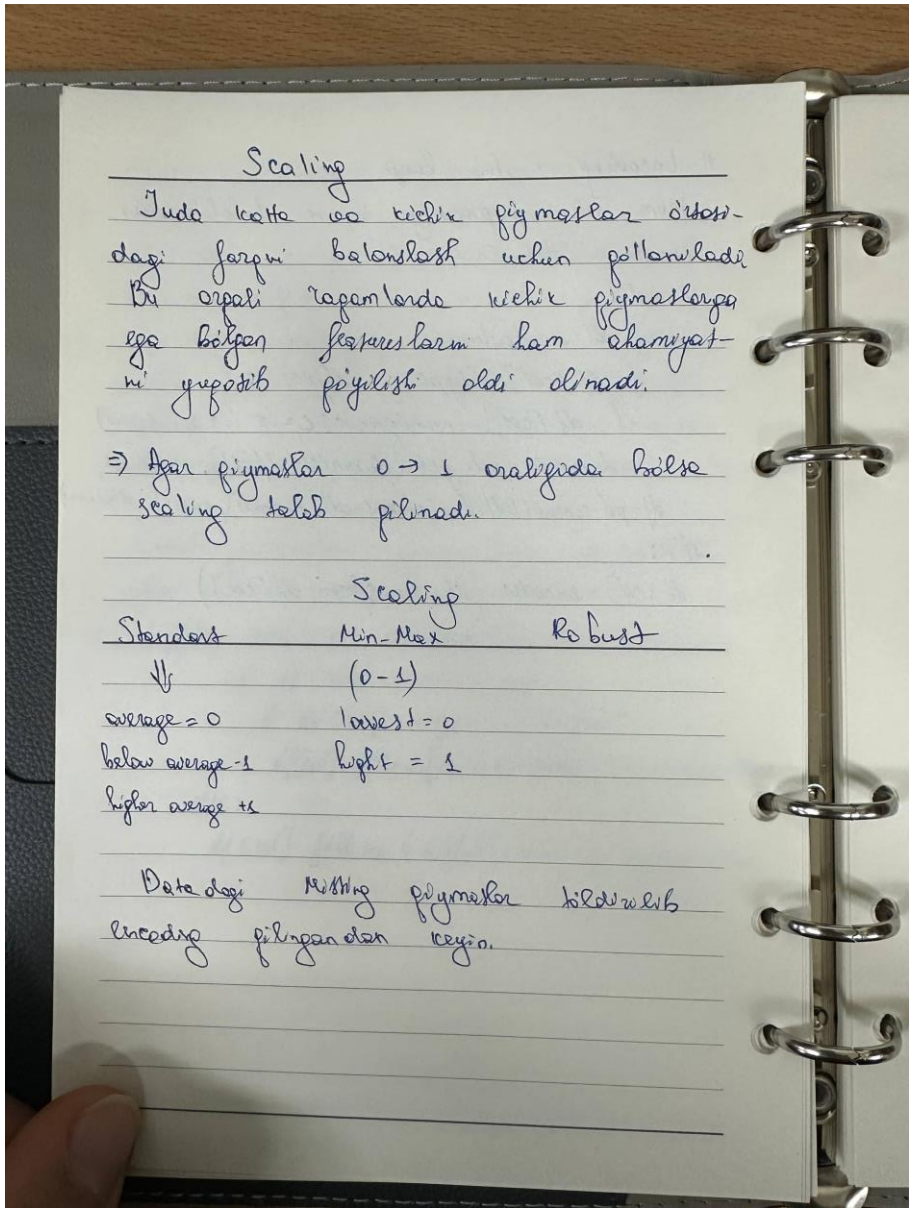
Turlib qoldirilgan qiymatlarni to'ldirish uchun

```
for col in columns:
    if df[col].isnull().any():
        if df[col].dtype == 'object':
            df[col].fillna(df[col].mode()[0], inplace=True
        else:
            df[col].fillna(df[col].mean(), inplace=True.
```



Encoding + for loop.

```
from sklearn.preprocessing import LabelEncoder.
encoder = LabelEncoder()
encoder
for col in df.columns:
    if df[col].dtype == 'object':
        if df[col].nunique() <= 5 (3, 2, 1, 10)
            dummies = pd.get_dummies(df[col], prefix=col,
    df = pd.concat([df.drop(columns=col), dummies], axis=1) dtype=int)
else:
    df[col] = encoder.fit_transform(df[col])
```



Scaling

Juda katta va kichik qiymatlar o'rtasidagi farqni balanslash uchun qo'llaniladi. Bu orqali raqamlarda kichik qiymatlarga ega bo'lgan featureslarni ham ahamiyatini yo'qotib qo'yilishi oldi olinadi.

⇒ Agar qiymatlar 0 → 1 oralig'ida bo'lsa scaling talab qilinadi.

Scaling

Standart	Min-Max	Robust
⇓	(0-1)	
average = 0	lowest = 0	
below average -1	high = 1	
higher average +1		

Data dagi missing qiymatlar to'ldirilib encoding qilingandan keyin.

from sklearn. ~~import~~ preprocessing import
Standart Scaler, Min.Max Scaler, Robust Scaler

⇓

num_col = df. select_dtypes (include = ['int64', 'float64']. colums. drop (output, Target)

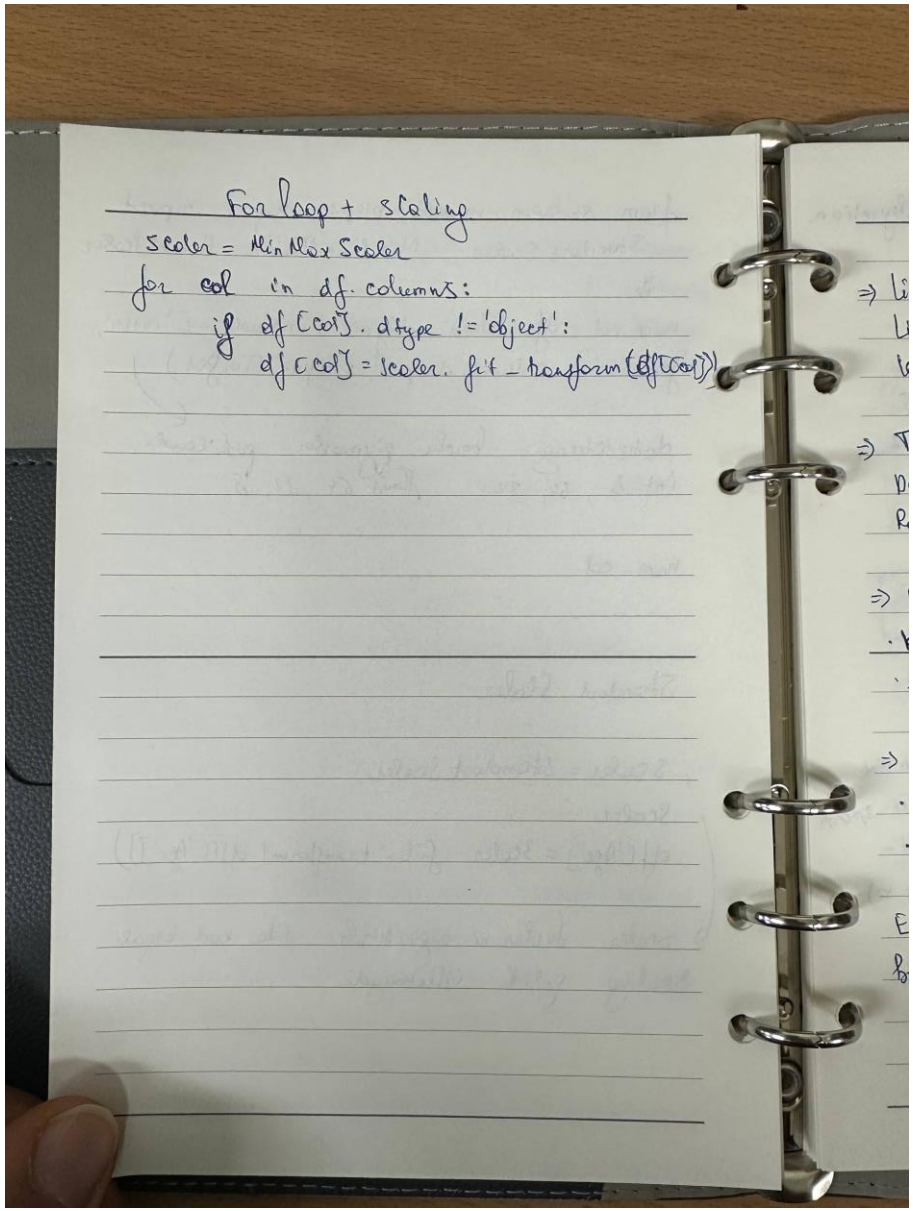
datasetdagi barcha qiymatlar qo'shiladi.
int 8, 64, 32   float 64, 32, 8.

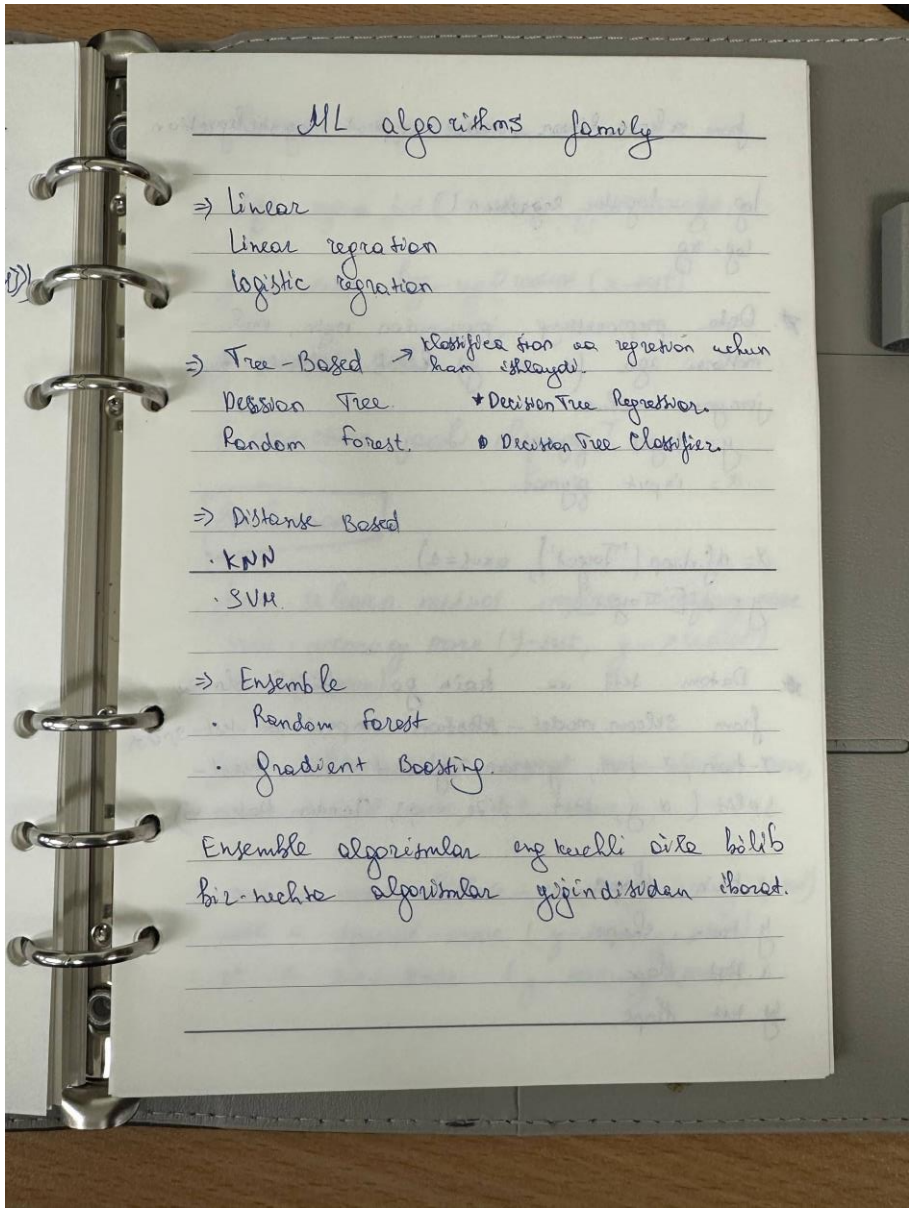
num_col.
......

Standart Scaler

Scaler = Standart Scaler
Scaler
df['Age'] = Scaler. fit_transform(df[['Age']])

→ Scaler turlarini o'zgartirib 1ta kod orqali Scaling qilish ishlamaydi.


For loop + scaling
scaler = MinMaxScaler
for col in df.columns:
if df[col].dtype != 'object':
df[col] = scaler.fit_transform(df[col])



ML algorithms family

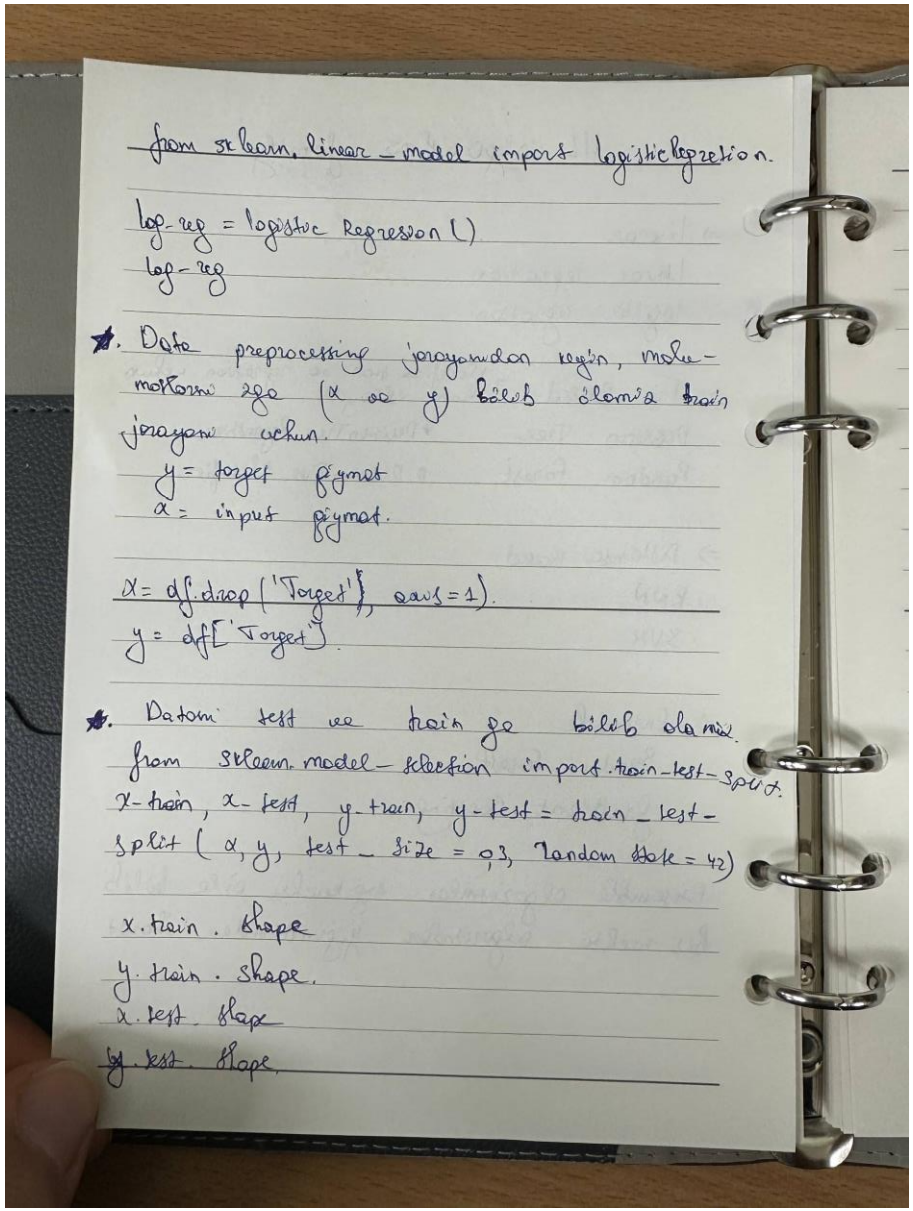
⇒ linear
linear regration
logistic regration

⇒ Tree-Based → klassifica tion ve regretion uchun ham ishlaydi.
Dession Tree * DecitionTree Regretion.
Random forest. * DecitionTree Classifier.

⇒ Distanse Based
· KNN
· SVM

⇒ Ensemble
· Random forest
· Gradient Boosting.

Ensemble algoritmlar eng kuchli ikki bo'lib
bir-nechta algoritmlar yig'indisidan iborat.



from sklearn.linear_model import LogisticRegression.

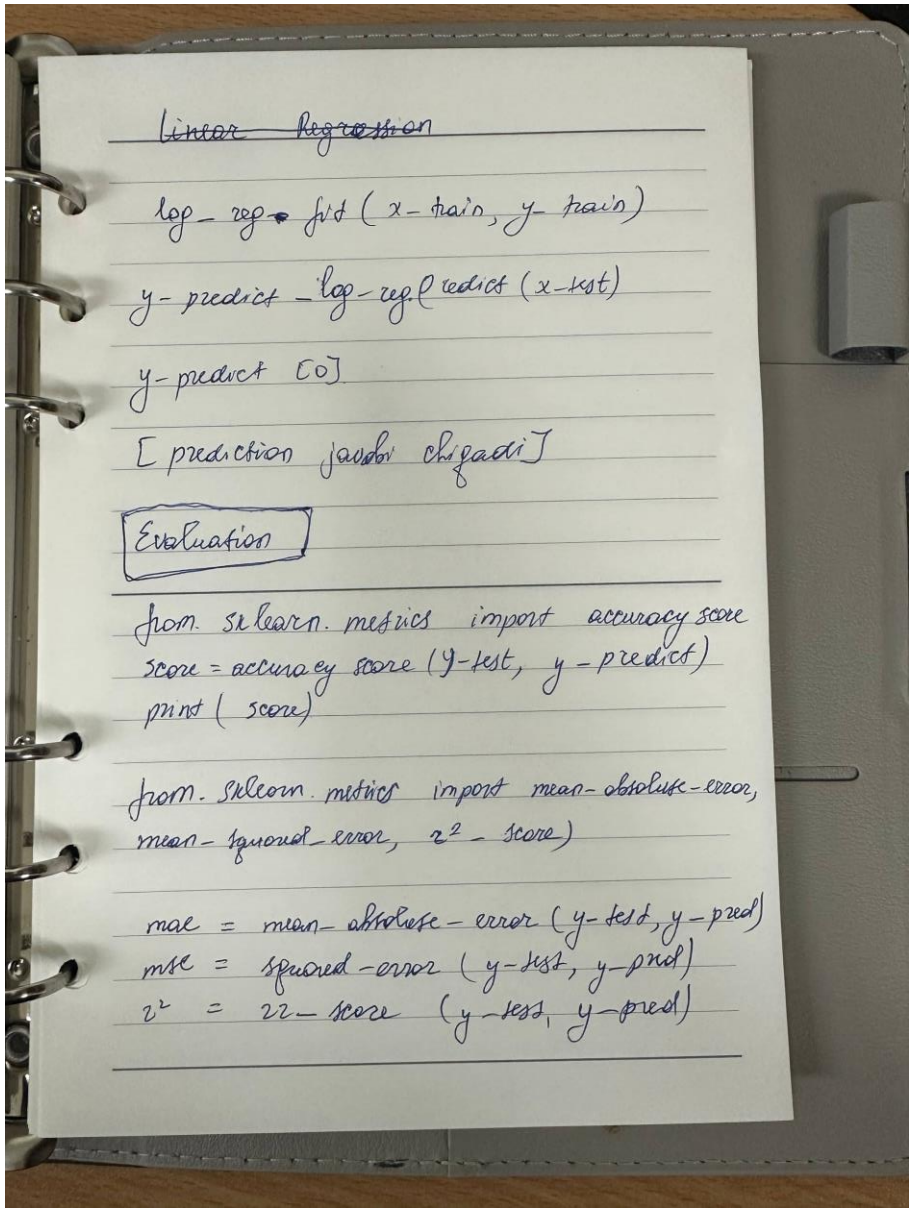
log_reg = LogisticRegression()
log_reg

* Data preprocessing jarayonidan keyin, modelni o'rgatish uchun (X va y) bo'lib olamiz train jarayoni uchun.
  y = target ustun
  X = input ustunlar.

X = df.drop('Target', axis=1).
y = df['Target'].

* Datani test va train ga bo'lib olamiz.
from sklearn.model_selection import train_test_split.
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

X_train.shape
y_train.shape.
X_test.shape
y_test.shape.



~~Linear Regression~~

log_reg.fit(x_train, y_train)

y_predict = log_reg.predict(x_test)

y_predict[0]

[prediction jaasbo chigadi]

Evaluation

from sklearn.metrics import accuracy_score
score = accuracy_score(y_test, y_predict)
print(score)

from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score

mae = mean_absolute_error(y_test, y_pred)
mse = squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)