حل مسئله 01

اضافه کردن کتابخانه pandas,numpy,matplotlib

In [21]:
```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
```

خواندن از فایل csv

In [3]:
```
df = pd.read_csv('./housePrice.csv')
```

مشاهده ۵ سطر اول دیتا فریم

In [4]:
```
df.head()
```

Out[4]:

	Area	Room	Parking	Warehouse	Elevator	Address	Price	Price(USD)
0	63	1	True	True	True	Shahran	1.850000e+09	61666.67
1	60	1	True	True	True	Shahran	1.850000e+09	61666.67
2	79	2	True	True	True	Pardis	5.500000e+08	18333.33
3	95	2	True	True	True	Shahrake Qods	9.025000e+08	30083.33
4	123	2	True	True	True	Shahrake Gharb	7.000000e+09	233333.33

بدست آوردن تعداد مقادیر یکتا ستون آدرس

In [5]:
```
len(df['Address'].unique())
```

Out[5]:
```
193
```

اضافه کردن sklearn استفاده از پکیج labelEncoder کتابخانه

In [6]:
```
from sklearn.preprocessing import LabelEncoder
```

تبدیل کردن داده های غیر عددی یا آبجکت به کد عددی

In [7]:
```
le = LabelEncoder()
```

برای یادگرفتن و فرمول ها و کارهایی که قرار است روی داده انجام شود تا یادگیری روی ستون آدرس متدfitانجام شود

In [8]:
```
le.fit(df["Address"])
```

Out[8]:
```
▾ LabelEncoder

LabelEncoder()
```

In [9]:
```
le.classes_
```

Out[9]:
```
array(['Abazar', 'Abbasabad', 'Absard', 'Abuzar', 'Afsarieh', 'Ahang',
       'Air force', 'Ajudaniye', 'Alborz Complex', 'Aliabad South',
       'Amir Bahador', 'Amirabad', 'Amirieh', 'Andisheh', 'Aqdasieh',
       'Araj', 'Argentina', 'Atabak', 'Azadshahr', 'Azarbaijan', 'Azari',
       'Baghestan', 'Bahar', 'Baqershahr', 'Beryanak', 'Boloorsazi',
       'Central Janatabad', 'Chahardangeh', 'Chardangeh', 'Chardivari',
       'Chidz', 'Damavand', 'Darabad', 'Darakeh', 'Darband', 'Daryan No',
       'Dehkade Olampic', 'Dezashib', 'Dolatabad', 'Dorous',
       'East Ferdows Boulevard', 'East Pars', 'Ekbatan', 'Ekhtiarieh',
       'Elahieh', 'Elm-o-Sanat', 'Enghelab', 'Eram', 'Eskandari',
       'Fallah', 'Farmanieh', 'Fatemi', 'Feiz Garden', 'Firoozkooh',
       'Firoozkooh Kuhsar', 'Gandhi', 'Garden of Saba', 'Gheitarieh',
       'Ghiyamdasht', 'Ghoba', 'Gholhak', 'Gisha', 'Golestan', 'Haft Tir',
       'Hakimiyeh', 'Hashemi', 'Hassan Abad', 'Hekmat', 'Heravi',
       'Heshmatieh', 'Hor Square', 'Islamshahr', 'Islamshahr Elahieh',
       'Javadiyeh', 'Jeyhoon', 'Jordan', 'Kahrizak', 'Kamranieh',
       'Karimkhan', 'Karoon', 'Kazemabad', 'Keshavarz Boulevard',
       'Khademabad Garden', 'Khavaran', 'Komeil', 'Koohsar', 'Kook',
       'Lavasan', 'Lavizan', 'Mahallati', 'Mahmoudieh', 'Majidieh',
       'Malard', 'Marzdaran', 'Mehrabad', 'Mehrabad River River',
       'Mehran', 'Mirdamad', 'Mirza Shirazi', 'Moniriyeh', 'Narmak',
       'Nasim Shahr', 'Nawab', 'Naziabad', 'Nezamabad', 'Niavaran',
       'North Program Organization', 'Northern Chitgar',
       'Northern Janatabad', 'Northern Suhrawardi', 'Northren Jamalzadeh',
       'Ostad Moein', 'Ozgol', 'Pakdasht', 'Pakdasht KhatunAbad',
       'Parand', 'Parastar', 'Pardis', 'Pasdaran',
       'Persian Gulf Martyrs Lake', 'Pirouzi', 'Pishva', 'Punak',
       'Qalandari', 'Qarchak', 'Qasr-od-Dasht', 'Qazvin Imamzadeh Hassan',
       'Railway', 'Ray', 'Ray - Montazeri', 'Ray - Pilgosh', 'Razi',
       'Republic', 'Robat Karim', 'Rudhen', 'Saadat Abad', 'SabaShahr',
       'Sabalan', 'Sadeghieh', 'Safadasht', 'Salehabad', 'Salsabil',
       'Sattarkhan', 'Seyed Khandan', 'Shadabad', 'Shahedshahr',
       'Shahr-e-Ziba', 'ShahrAra', 'Shahrake Apadana', 'Shahrake Azadi',
       'Shahrake Gharb', 'Shahrake Madaen', 'Shahrake Qods',
       'Shahrake Quds', 'Shahrake Shahid Bagheri', 'Shahrakeh Naft',
       'Shahran', 'Shahryar', 'Shams Abad', 'Shoosh', 'Si Metri Ji',
       'Sohanak', 'Southern Chitgar', 'Southern Janatabad',
       'Southern Program Organization', 'Southern Suhrawardi', 'Tajrish',
       'Tarasht', 'Taslihat', 'Tehran Now', 'Tehransar',
       'Telecommunication', 'Tenant', 'Thirteen November', 'Vahidieh',
       'Vahidiyeh', 'Valiasr', 'Vanak', 'Varamin - Beheshti', 'Velenjak',
       'Villa', 'Water Organization', 'Waterfall',
       'West Ferdows Boulevard', 'West Pars', 'Yaftabad', 'Yakhchiabad',
       'Yousef Abad', 'Zafar', 'Zaferanieh', 'Zargandeh', 'Zibadasht',
       nan], dtype=object)
```

یادگیری انجام شده را پس از تبدیل در همان ستون ذخیره می کند.

In [10]:
```
df['Address'] = le.transform(df['Address'])
```

مشاهده ۵ سطر اول دیتا فریم

In [11]:
```
df.head()
```

Out[11]:

	Area	Room	Parking	Warehouse	Elevator	Address	Price	Price(USD)
0	63	1	True	True	True	156	1.850000e+09	61666.67
1	60	1	True	True	True	156	1.850000e+09	61666.67
2	79	2	True	True	True	117	5.500000e+08	18333.33
3	95	2	True	True	True	152	9.025000e+08	30083.33
4	123	2	True	True	True	150	7.000000e+09	233333.33

حذف ستون قیمت

In [12]:
```
df.drop(["Price"], axis=1, inplace=True)
```

ابعاد دیتافریم را نشان می دهد.

In [13]:
```
df.shape
```

Out[13]: (3479, 7)

مشاهده اطلاعات دیتا فریم

In [14]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3479 entries, 0 to 3478
Data columns (total 7 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   Area         3479 non-null   object 
 1   Room         3479 non-null   int64  
 2   Parking      3479 non-null   bool   
 3   Warehouse    3479 non-null   bool   
 4   Elevator     3479 non-null   bool   
 5   Address      3479 non-null   int64  
 6   Price(USD)   3479 non-null   float64
dtypes: bool(3), float64(1), int64(2), object(1)
memory usage: 119.0+ KB
```

مشاهده و انجام یکسری عملیات آماری روی دیتا فریم مثل میانگین وغیره

In [18]:
```
df.describe()
```

Out[18]:

	Room	Address	Price(USD)
count	3479.000000	3479.000000	3.479000e+03
mean	2.079908	105.536648	1.786341e+05
std	0.758275	50.653530	2.699978e+05
min	0.000000	0.000000	1.200000e+02
25%	2.000000	62.000000	4.727500e+04
50%	2.000000	117.000000	9.666667e+04
75%	2.000000	146.000000	2.000000e+05
max	5.000000	192.000000	3.080000e+06

تبدیل متغیر آبجکت به عددی

In [23]:
```
pd.to_numeric(df['Area'])
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
File lib.pyx:2368, in pandas._libs.lib.maybe_convert_numeric()

ValueError: Unable to parse string " 3,310,000,000 "

During handling of the above exception, another exception occurred:

ValueError                                Traceback (most recent call last)
Cell In[23], line 1
----> 1 pd.to_numeric(df['Area'])

File ~/miniconda3/lib/python3.11/site-packages/pandas/core/tools/numeric.py:
222, in to_numeric(arg, errors, downcast, dtype_backend)
    220 coerce_numeric = errors not in ("ignore", "raise")
    221 try:
--> 222     values, new mask = lib.maybe_convert_numeric(  # type: ignore[ca
ll-overload]  # noqa: E501
    223         values,
    224         set(),
    225         coerce numeric=coerce numeric,
    226         convert_to_masked_nullable=dtype_backend is not lib.no_defau
lt
    227         or isinstance(values_dtype, StringDtype),
    228     )
    229 except (ValueError, TypeError):
    230     if errors == "raise":

File lib.pyx:2410, in pandas._libs.lib.maybe_convert_numeric()

ValueError: Unable to parse string " 3,310,000,000 " at position 570
```

حذف سطرهایی که داده های پرت دارند

In [24]:
```
df["Area"] = df['Area'].drop([570, 709, 807, 1604, 2171, 2802])
```

تبدیل ستون مساحت که نوع آن آبجکت بود به ماتریس عددی در پانداس

In [25]:
```
pd.to_numeric(df['Area'])
```

Out[25]:
```
0        63.0
1        60.0
2        79.0
3        95.0
4       123.0
        ...
3474     86.0
3475     83.0
3476     75.0
3477    105.0
3478     82.0
Name: Area, Length: 3479, dtype: float64
```

In [26]:
```
df.info()
```

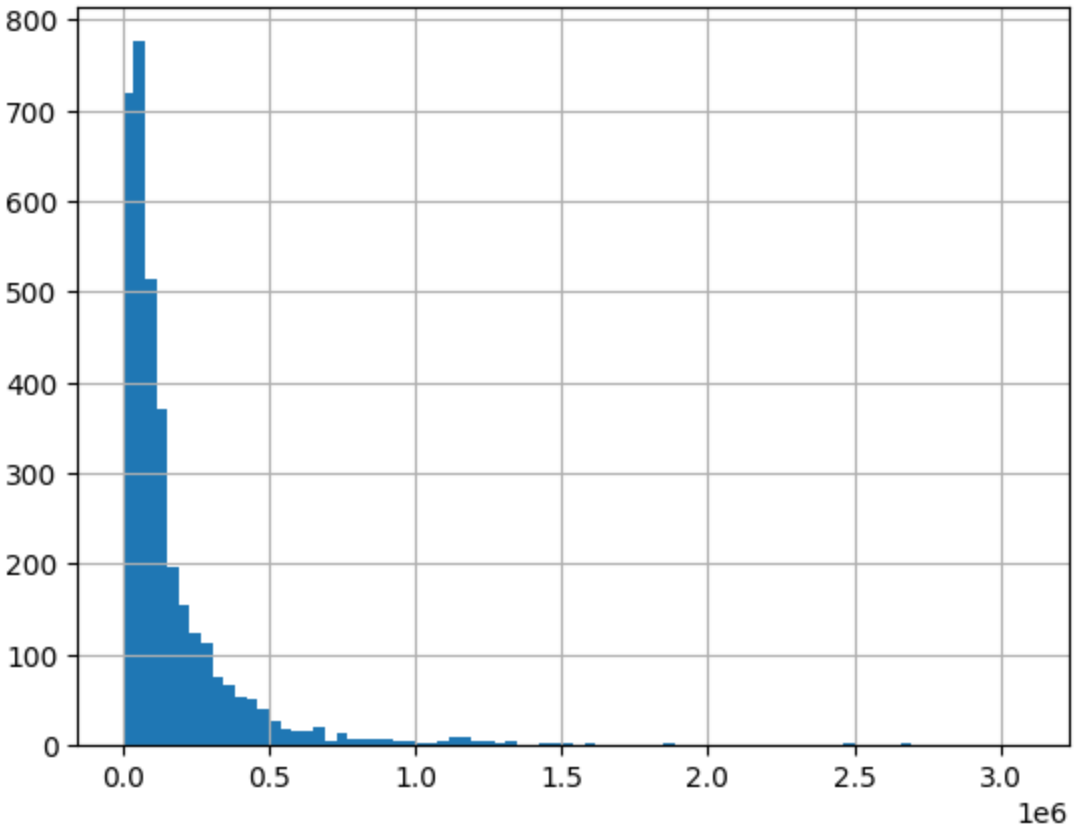
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3479 entries, 0 to 3478
Data columns (total 7 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   Area        3473 non-null   object
 1   Room        3479 non-null   int64
 2   Parking     3479 non-null   bool
 3   Warehouse   3479 non-null   bool
 4   Elevator    3479 non-null   bool
 5   Address     3479 non-null   int64
 6   Price(USD)  3479 non-null   float64
dtypes: bool(3), float64(1), int64(2), object(1)
memory usage: 119.0+ KB
```

In [27]:
```
df.dropna(inplace=True)
```

نمایش نمودار هیستوگرام برای داده های قیمت

In [28]:
```
df["Price(USD)"].hist(bins=80)
```

Out[28]:
```
<Axes: >
```



نمایش ستون اتاق

```
In [29]: df["Room"]
```

```
Out[29]: 0       1
         1       1
         2       2
         3       2
         4       2
                ..
         3474    2
         3475    2
         3476    2
         3477    2
         3478    2
         Name: Room, Length: 3473, dtype: int64
```

دیتافریم را به ماتریس نامپای تبدیل می کند و از سطراول تا آخر و ستون اول تا یکی مانده به آخر دیتافریم را در ماتریس نامپای در متغیری ذخیره می کند

```
In [30]: X = df.iloc[:, :-1].values
```

متدshapeابعاد ماتریس مشخص میکند

```
In [31]: X.shape
```

```
Out[31]: (3473, 6)
```

In [37]:
```
X
```

Out[37]:
```
array([['63', 1, True, True, True, 156],
       ['60', 1, True, True, True, 156],
       ['79', 2, True, True, True, 117],
       ...,
       ['75', 2, False, False, False, 115],
       ['105', 2, True, True, True, 39],
       ['82', 2, False, True, True, 115]], dtype=object)
```

ستون آخری دیتافریم را به ماتریس نامپای تبدیل می کند و در متغیری ذخیره می کند.

In [32]:
```
y = df.iloc[:, -1].values
```

In [70]:
```
y
```

Out[70]:
```
array([ 61666.67,  61666.67,  18333.33, ...,  12166.67, 186666.67,
        12000.  ])
```

In [33]:
```
y.shape
```

Out[33]:
```
(3473,)
```

اضافه کردن کتابخانه sklearnو پکیج مورد نظر برای داده های آموزشی

In [34]:
```
from sklearn.model_selection import train_test_split
```

انتخاب مقداری از داده ها بعنوان مجموعه آموزشی و مجموعه تست. بخش آموزش شامل 80 درصد داده ها و بخش تست شامل 20 درصد داده ها است.

In [35]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y)
```

In [36]:
```
X_train.shape, y_train.shape
```

Out[36]:
```
((2604, 6), (2604,))
```

In [37]:
```
X_test.shape, y_test.shape
```

Out[37]:
```
((869, 6), (869,))
```

استفاده از پکیج رگرسیون خطی از کتابخانه sklearn برای بکارگیری الگوریتم رگرسیون خطی

In [38]:
```
from sklearn.linear_model import LinearRegression
```

In [39]:
```
lr = LinearRegression()
```

برای یادگرفتن و فرمول ها و کارهایی که قرار است روی داده آموزشی انجام شود تا یادگیری انجام شودfitمتد

In [40]:
```
lr.fit(X_train, y_train)
```

Out[40]:
```
▾ LinearRegression

LinearRegression()
```

بدست آوردن ضریب ویژگیها یا همان وزن

In [41]:
```
# y = (w0x0 + w1x1 + w2x2 + w3x3 + w4x4 + w5x5) + b, x0=1
lr.coef_
```

Out[41]:
```
array([ 2361.47219363, 50320.7772594 , -7204.9719861 , 36931.80306697,
       36150.47802106,    62.18763294])
```

بدست آوردن مقدار بایاس

In [42]:
```
lr.intercept_# is the b(w0)
```

Out[42]:
```
-240321.66555144987
```

برای اینکه بفهمیم مدل ما تاچه اندازه داده های آموزشی را می فهمد

In [43]:
```
lr.score(X_train, y_train) #54%
```

Out[43]:
```
0.51824302796679
```

برای درک اینکه مدل چگونه اشتباه می کند می توانیم از داده های تست برای پیش بینی استفاده کنیم

In [44]:
```
y_pred = lr.predict(X_test)#
```

استفاده از پکیجmean_absolute_errorکتابخانه sklearnبرای گزارش میزان خطا

In [52]:
```
from sklearn.metrics import mean_absolute_error,mean_squared_error
```

استفاده ازloss function (MSE)برای میزان خطا بین داده های واقعی و مقدارپیش بینی شده

In [53]:
```
mean_squared_error(y_test, y_pred)
```

Out[53]:
```
28699614241.812115
```

استفاده ازloss function (MAE)برای میزان خطا بین داده های واقعی و مقدارپیش بینی شده

In [54]:
```
mean_absolute_error(y_test, y_pred)
```

Out[54]:
```
90367.41732618124
```

In [ ]:
```

```