Programming for Artificial Intelligence Lab

SUPERIOR UNIVERSITY

Name:

Mahak Farhan

Roll no:

068

Class:

BSAI

Section:

4B

Subject:

PAI(Lab)

Submitted to:

Sir Rasikh Ali

LAB 1

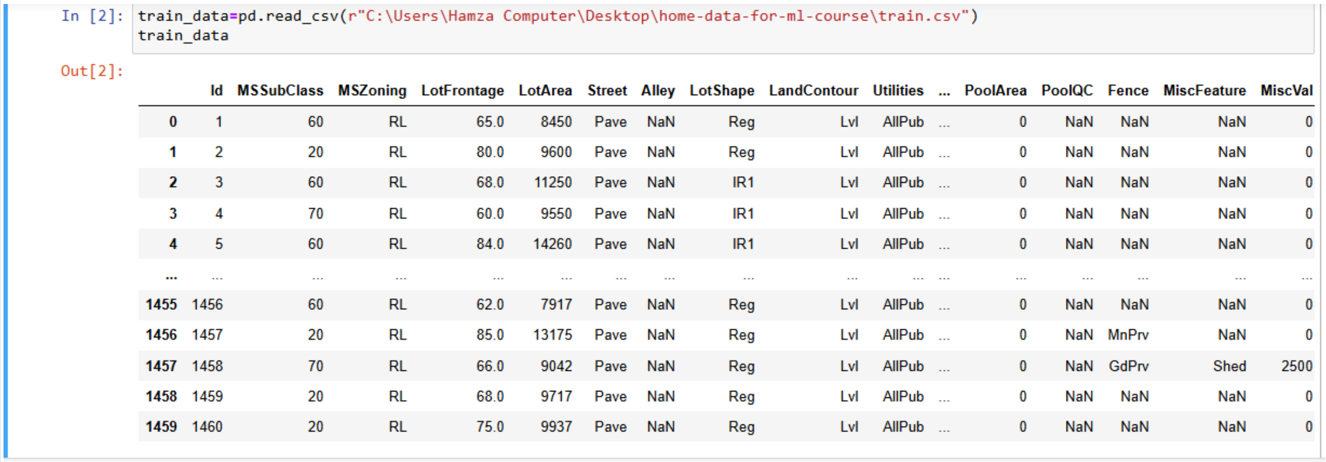
HOUSE PRICE PREDICTION

Code:

```
import pandas as pd

import numpy as np

from sklearn.svm import SVC

from sklearn.metrics import accuracy_score

from sklearn.preprocessing import LabelEncoder

from sklearn.model_selection import train_test_split
```

Reading training data:

```
train_data=pd.read_csv(r"C:\Users\Hamza Computer\Desktop\home-data-for-ml-course\train.csv")

train_data
```



```
In [2]: train_data=pd.read_csv(r"C:\Users\Hamza Computer\Desktop\home-data-for-ml-course\train.csv")
        train_data
```

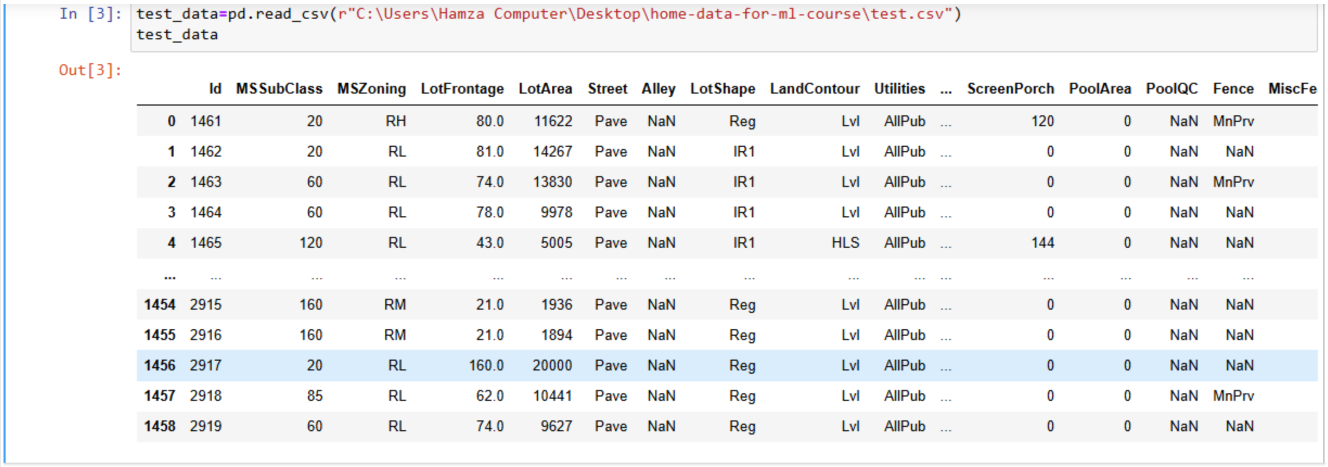
Out[2]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	...	PoolArea	PoolQC	Fence	MiscFeature	MiscVal
0	1	60	RL	65.0	8450	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	NaN	0
1	2	20	RL	80.0	9600	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	NaN	0
2	3	60	RL	68.0	11250	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	NaN	0
3	4	70	RL	60.0	9550	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	NaN	0
4	5	60	RL	84.0	14260	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	NaN	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1455	1456	60	RL	62.0	7917	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	NaN	0
1456	1457	20	RL	85.0	13175	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	MnPrv	NaN	0
1457	1458	70	RL	66.0	9042	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	GdPrv	Shed	2500
1458	1459	20	RL	68.0	9717	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	NaN	0
1459	1460	20	RL	75.0	9937	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	NaN	0

Reading testing data:

test_data=pd.read_csv(r"C:\Users\Hamza Computer\Desktop\home-data-for-ml-course\test.csv")

test_data

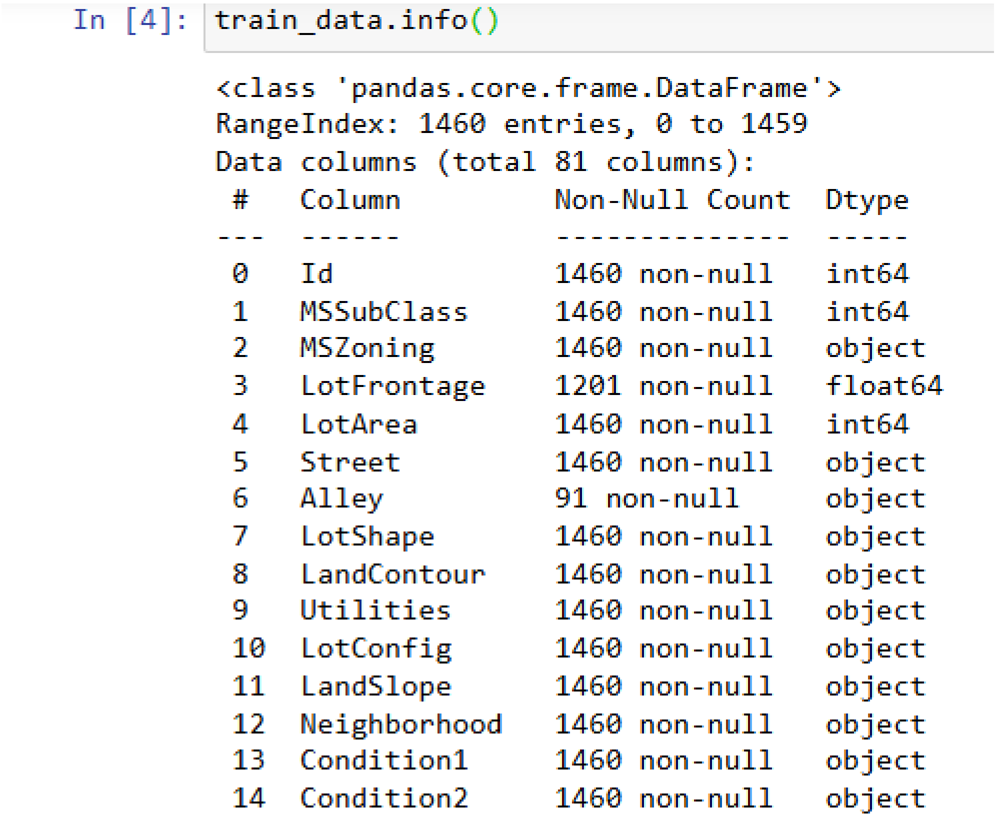


```
In [3]: test_data=pd.read_csv(r"C:\Users\Hamza Computer\Desktop\home-data-for-ml-course\test.csv")
        test_data
Out[3]:
```

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	...	ScreenPorch	PoolArea	PoolQC	Fence	MiscFe
0	1461	20	RH	80.0	11622	Pave	NaN	Reg	Lvl	AllPub	...	120	0	NaN	MnPrv	
1	1462	20	RL	81.0	14267	Pave	NaN	IR1	Lvl	AllPub	...	0	0	NaN	NaN	
2	1463	60	RL	74.0	13830	Pave	NaN	IR1	Lvl	AllPub	...	0	0	NaN	MnPrv	
3	1464	60	RL	78.0	9978	Pave	NaN	IR1	Lvl	AllPub	...	0	0	NaN	NaN	
4	1465	120	RL	43.0	5005	Pave	NaN	IR1	HLS	AllPub	...	144	0	NaN	NaN	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
1454	2915	160	RM	21.0	1936	Pave	NaN	Reg	Lvl	AllPub	...	0	0	NaN	NaN	
1455	2916	160	RM	21.0	1894	Pave	NaN	Reg	Lvl	AllPub	...	0	0	NaN	NaN	
1456	2917	20	RL	160.0	20000	Pave	NaN	Reg	Lvl	AllPub	...	0	0	NaN	NaN	
1457	2918	85	RL	62.0	10441	Pave	NaN	Reg	Lvl	AllPub	...	0	0	NaN	MnPrv	
1458	2919	60	RL	74.0	9627	Pave	NaN	Reg	Lvl	AllPub	...	0	0	NaN	NaN	

Information of training data:

train_data.info()



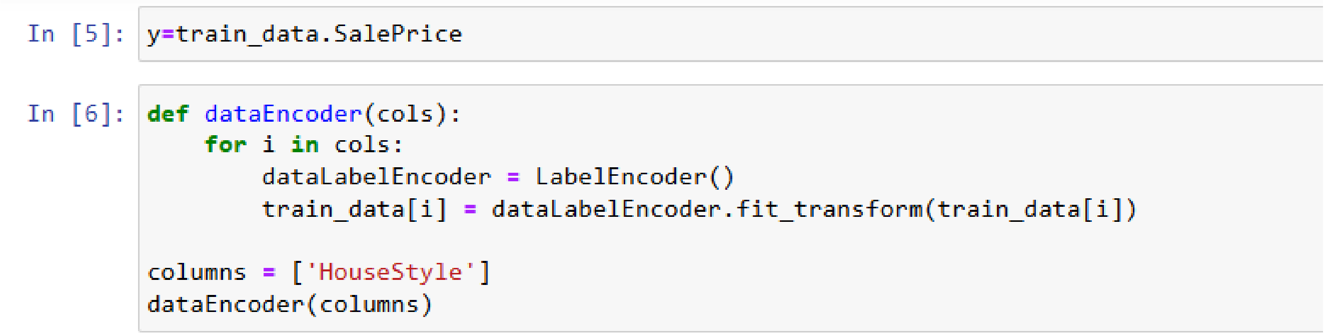
```
In [4]: train_data.info()

        <class 'pandas.core.frame.DataFrame'>
        RangeIndex: 1460 entries, 0 to 1459
        Data columns (total 81 columns):
         #   Column         Non-Null Count  Dtype
        ---  ------         --------------  -----
         0   Id             1460 non-null   int64
         1   MSSubClass     1460 non-null   int64
         2   MSZoning       1460 non-null   object
         3   LotFrontage    1201 non-null   float64
         4   LotArea        1460 non-null   int64
         5   Street         1460 non-null   object
         6   Alley          91 non-null     object
         7   LotShape       1460 non-null   object
         8   LandContour    1460 non-null   object
         9   Utilities      1460 non-null   object
         10  LotConfig      1460 non-null   object
         11  LandSlope      1460 non-null   object
         12  Neighborhood   1460 non-null   object
         13  Condition1     1460 non-null   object
         14  Condition2     1460 non-null   object
```

Selecting the column on which we have to predict:

```
y=train_data.SalePrice
```

Converting object columns into int of training data:

```
def dataEncoder(cols):
    for i in cols:
        dataLabelEncoder = LabelEncoder()
        train_data[i] = dataLabelEncoder.fit_transform(train_data[i])
columns = ['HouseStyle']
dataEncoder(columns)
```

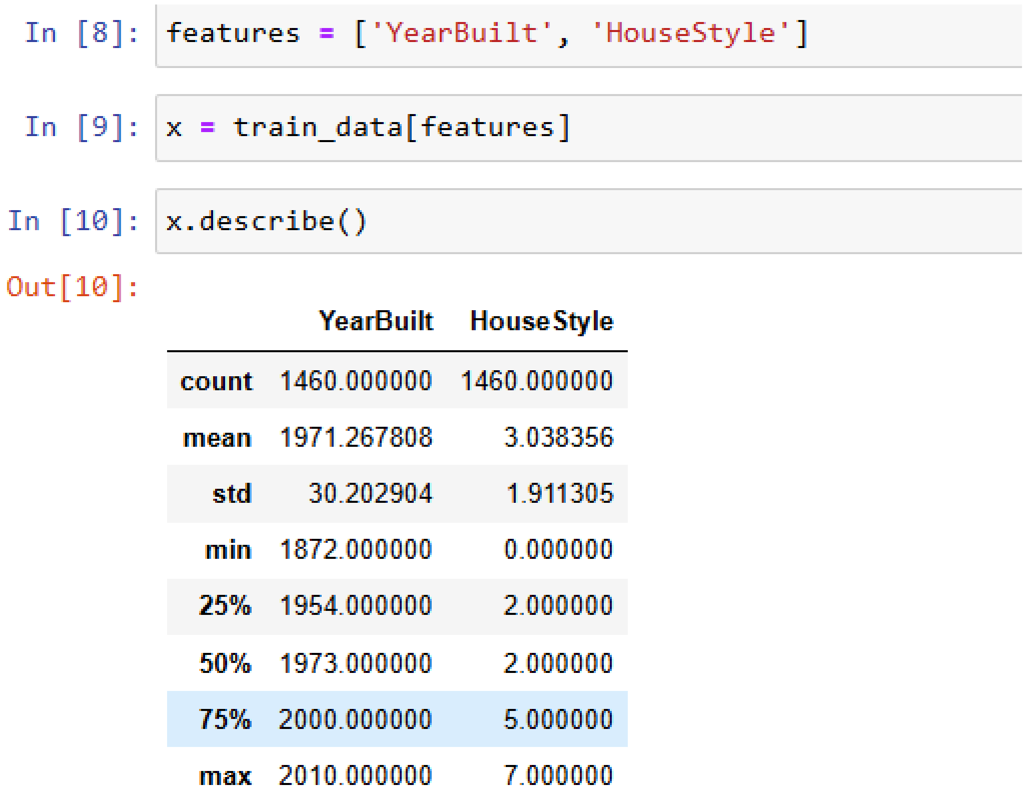


```
In [5]: y=train_data.SalePrice
```

```
In [6]: def dataEncoder(cols):
            for i in cols:
                dataLabelEncoder = LabelEncoder()
                train_data[i] = dataLabelEncoder.fit_transform(train_data[i])

        columns = ['HouseStyle']
        dataEncoder(columns)
```

Selecting the attributes on which we have to predict:

```
features = ['YearBuilt', 'HouseStyle']
x = train_data[features]
x.describe()
```



```
In [8]: features = ['YearBuilt', 'HouseStyle']

In [9]: x = train_data[features]

In [10]: x.describe()

Out[10]:
```

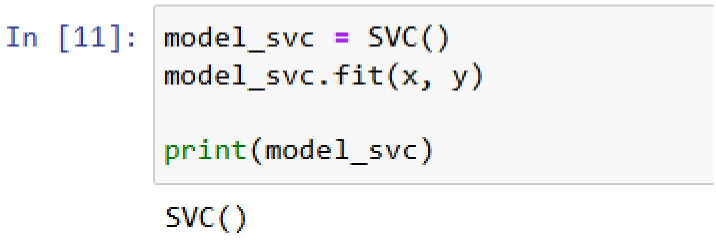
	YearBuilt	HouseStyle
count	1460.000000	1460.000000
mean	1971.267808	3.038356
std	30.202904	1.911305
min	1872.000000	0.000000
25%	1954.000000	2.000000
50%	1973.000000	2.000000
75%	2000.000000	5.000000
max	2010.000000	7.000000

Training the model:

model_svc = SVC()

model_svc.fit(x, y)

print(model_svc)



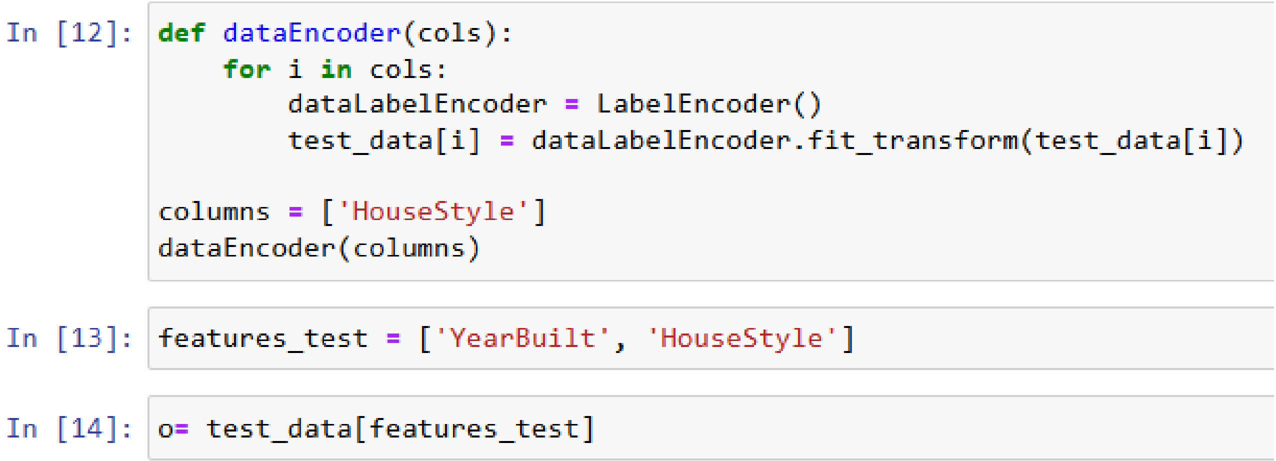
```
In [11]: model_svc = SVC()
         model_svc.fit(x, y)

         print(model_svc)

         SVC()
```

Convert object columns into int of testing data:

```
def dataEncoder(cols):
    for i in cols:
        dataLabelEncoder = LabelEncoder()
        test_data[i] = dataLabelEncoder.fit_transform(test_data[i])

columns = ['HouseStyle']
dataEncoder(columns)
```

Selecting the features on which we have to test:

```
features_test = ['YearBuilt', 'HouseStyle']
o= test_data[features_test]
```

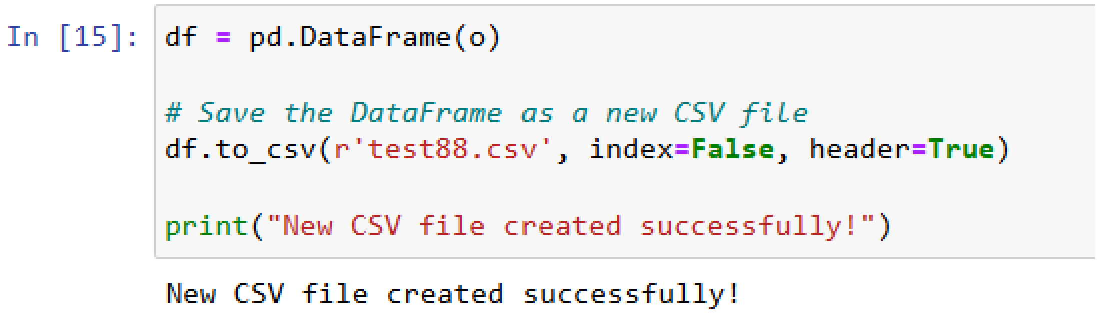


```
In [12]: def dataEncoder(cols):
             for i in cols:
                 dataLabelEncoder = LabelEncoder()
                 test_data[i] = dataLabelEncoder.fit_transform(test_data[i])

         columns = ['HouseStyle']
         dataEncoder(columns)

In [13]: features_test = ['YearBuilt', 'HouseStyle']

In [14]: o= test_data[features_test]
```

Converting testing features into a new CSV file

```
df = pd.DataFrame(o)
# Save the DataFrame as a new CSV file
```

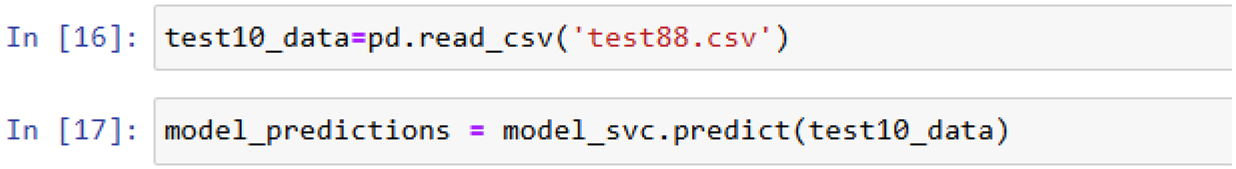
```
df.to_csv(r'test88.csv', index=False, header=True)
print("New CSV file created successfully!")
```



```
In [15]: df = pd.DataFrame(o)

         # Save the DataFrame as a new CSV file
         df.to_csv(r'test88.csv', index=False, header=True)

         print("New CSV file created successfully!")

         New CSV file created successfully!
```

Now we are predicting on new CSV that we made from testing features:

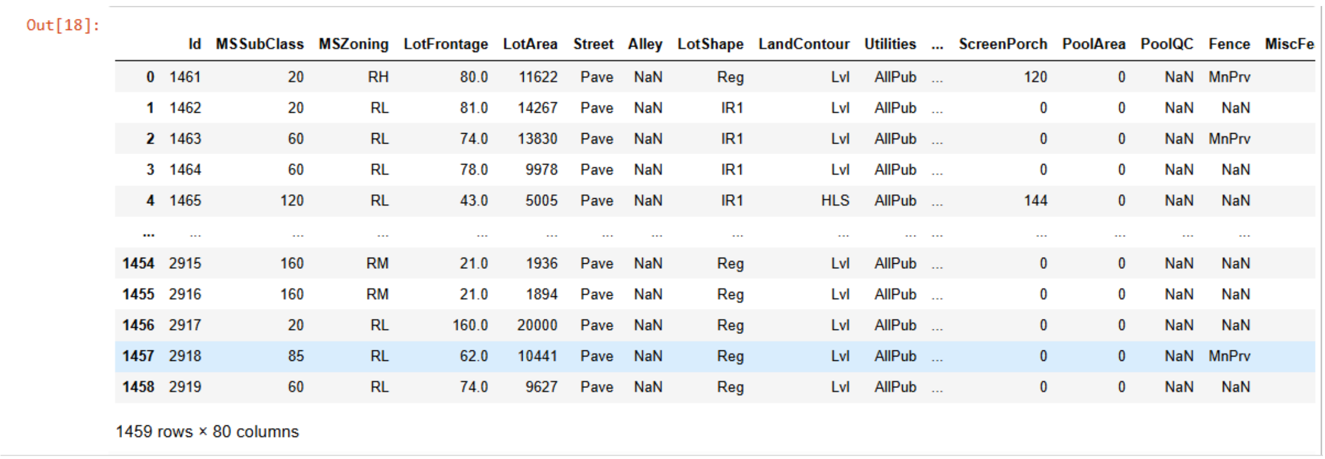
```
test10_data=pd.read_csv('test88.csv')
model_predictions = model_svc.predict(test10_data)
```



```
In [16]: test10_data=pd.read_csv('test88.csv')

In [17]: model_predictions = model_svc.predict(test10_data)
```

Again we are reading testing data CSV file because the format of submission file is ID and price so from this we will extract the Id column:

```
x=pd.read_csv(r"C:\Users\Hamza Computer\Desktop\home-data-for-ml-course\test.csv")
x
```



Out[18]:

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	...	ScreenPorch	PoolArea	PoolQC	Fence	MiscFe
0	1461	20	RH	80.0	11622	Pave	NaN	Reg	Lvl	AllPub	...	120	0	NaN	MnPrv	
1	1462	20	RL	81.0	14267	Pave	NaN	IR1	Lvl	AllPub	...	0	0	NaN	NaN	
2	1463	60	RL	74.0	13830	Pave	NaN	IR1	Lvl	AllPub	...	0	0	NaN	MnPrv	
3	1464	60	RL	78.0	9978	Pave	NaN	IR1	Lvl	AllPub	...	0	0	NaN	NaN	
4	1465	120	RL	43.0	5005	Pave	NaN	IR1	HLS	AllPub	...	144	0	NaN	NaN	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
1454	2915	160	RM	21.0	1936	Pave	NaN	Reg	Lvl	AllPub	...	0	0	NaN	NaN	
1455	2916	160	RM	21.0	1894	Pave	NaN	Reg	Lvl	AllPub	...	0	0	NaN	NaN	
1456	2917	20	RL	160.0	20000	Pave	NaN	Reg	Lvl	AllPub	...	0	0	NaN	NaN	
1457	2918	85	RL	62.0	10441	Pave	NaN	Reg	Lvl	AllPub	...	0	0	NaN	MnPrv	
1458	2919	60	RL	74.0	9627	Pave	NaN	Reg	Lvl	AllPub	...	0	0	NaN	NaN	

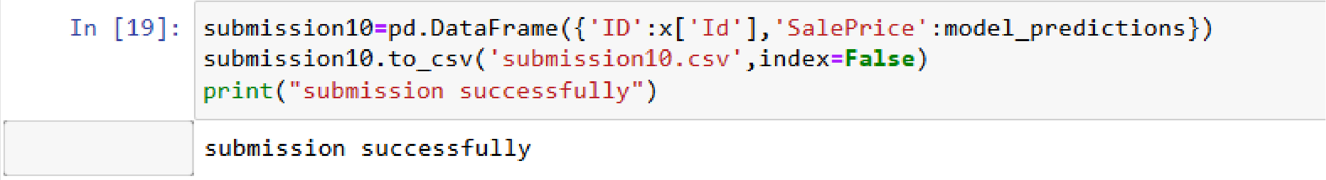
1459 rows × 80 columns

Submission file:

submission10=pd.DataFrame({'ID':x['Id'],'SalePrice':model_predictions})

submission10.to_csv('submission10.csv',index=False)

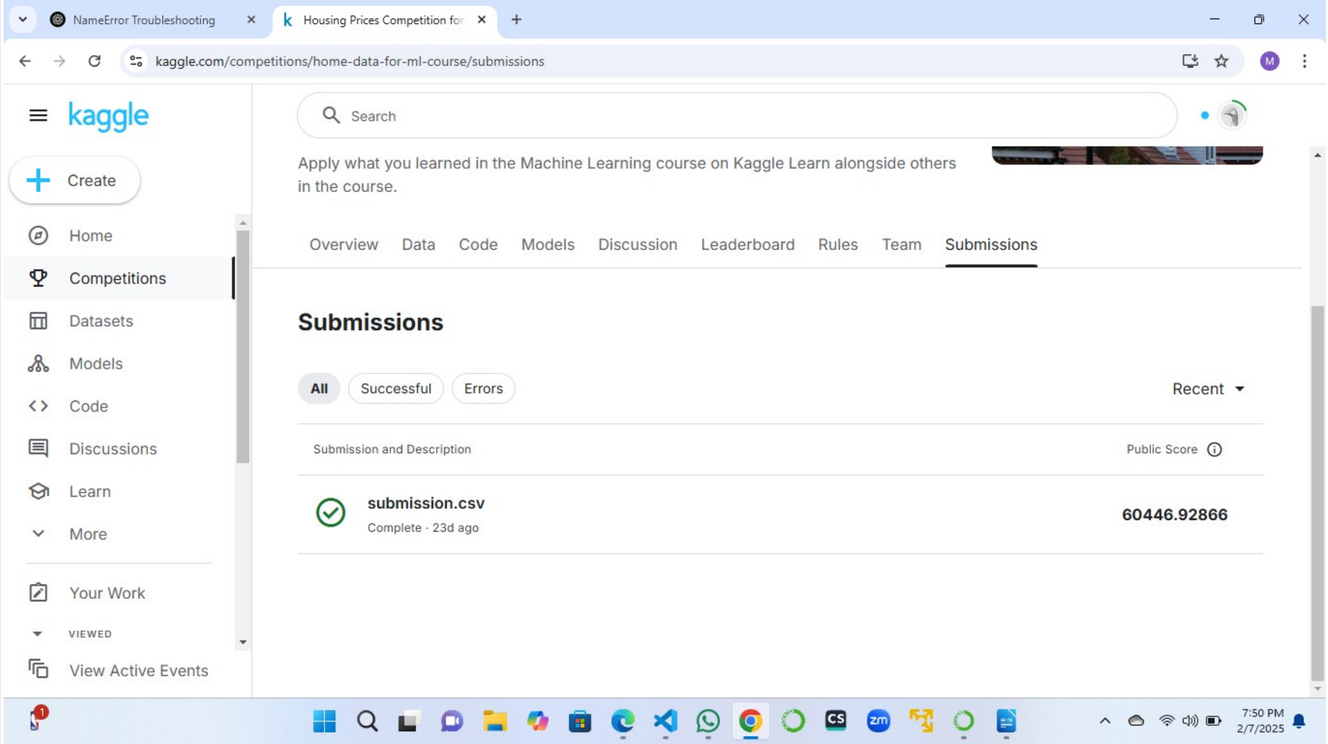
print("submission successfully")



```
In [19]: submission10=pd.DataFrame({'ID':x['Id'],'SalePrice':model_predictions})
         submission10.to_csv('submission10.csv',index=False)
         print("submission successfully")
```

```
submission successfully
```

Accuracy:



Apply what you learned in the Machine Learning course on Kaggle Learn alongside others in the course.

Submissions

Submission and Description	Public Score
submission.csv Complete · 23d ago	60446.92866
