Assignment 4

Bekzhanov Namazbek

```
from sklearn.datasets import fetch_openml
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

data = fetch_openml('house_prices', as_frame=True)
df = data.frame
df.head(25)
```

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	...	PoolArea	PoolQC	Fence
0	1	60	RL	65.0	8450	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN
1	2	20	RL	80.0	9600	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN
2	3	60	RL	68.0	11250	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN
3	4	70	RL	60.0	9550	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN
4	5	60	RL	84.0	14260	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN
5	6	50	RL	85.0	14115	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	MnPrv
6	7	20	RL	75.0	10084	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN
7	8	60	RL	NaN	10382	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN
8	9	50	RM	51.0	6120	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN
9	10	190	RL	50.0	7420	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN
10	11	20	RL	70.0	11200	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN
11	12	60	RL	85.0	11924	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN
12	13	20	RL	NaN	12968	Pave	NaN	IR2	Lvl	AllPub	...	0	NaN	NaN
13	14	20	RL	91.0	10652	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN
14	15	20	RL	NaN	10920	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	GdWo
15	16	45	RM	51.0	6120	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	GdPrv
16	17	20	RL	NaN	11241	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN
17	18	90	RL	72.0	10791	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN
18	19	20	RL	66.0	13695	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN
19	20	20	RL	70.0	7560	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	MnPrv
20	21	60	RL	101.0	14215	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN
21	22	45	RM	57.0	7449	Pave	Grvl	Reg	Bnk	AllPub	...	0	NaN	GdPrv
22	23	20	RL	75.0	9742	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN
23	24	120	RM	44.0	4224	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN
24	25	20	RL	NaN	8246	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	MnPrv

25 rows × 81 columns

Q2.1 Answers:

1. number samples and features = (1460, 81)
2. data types = df.dtypes
3. missing values = LotFrontage 259

```
df.shape, df.dtypes, df.isnull().sum()
```

```
((1460, 81),
 Id               int64
 MSSubClass       int64
 MSZoning        object
 LotFrontage    float64
 LotArea          int64
                 ...
 MoSold           int64
 YrSold           int64
 SaleType        object
 SaleCondition   object
 SalePrice        int64
 Length: 81, dtype: object,
 Id               0
 MSSubClass       0
```

```
MSZoning           0
LotFrontage      259
LotArea            0
                ...
MoSold             0
YrSold             0
SaleType           0
SaleCondition      0
SalePrice          0
Length: 81, dtype: int64)
```

Q2.2 Answer: regression

Q2.3 Answer: target = 'SalePrice'

Q2.4 Answer:

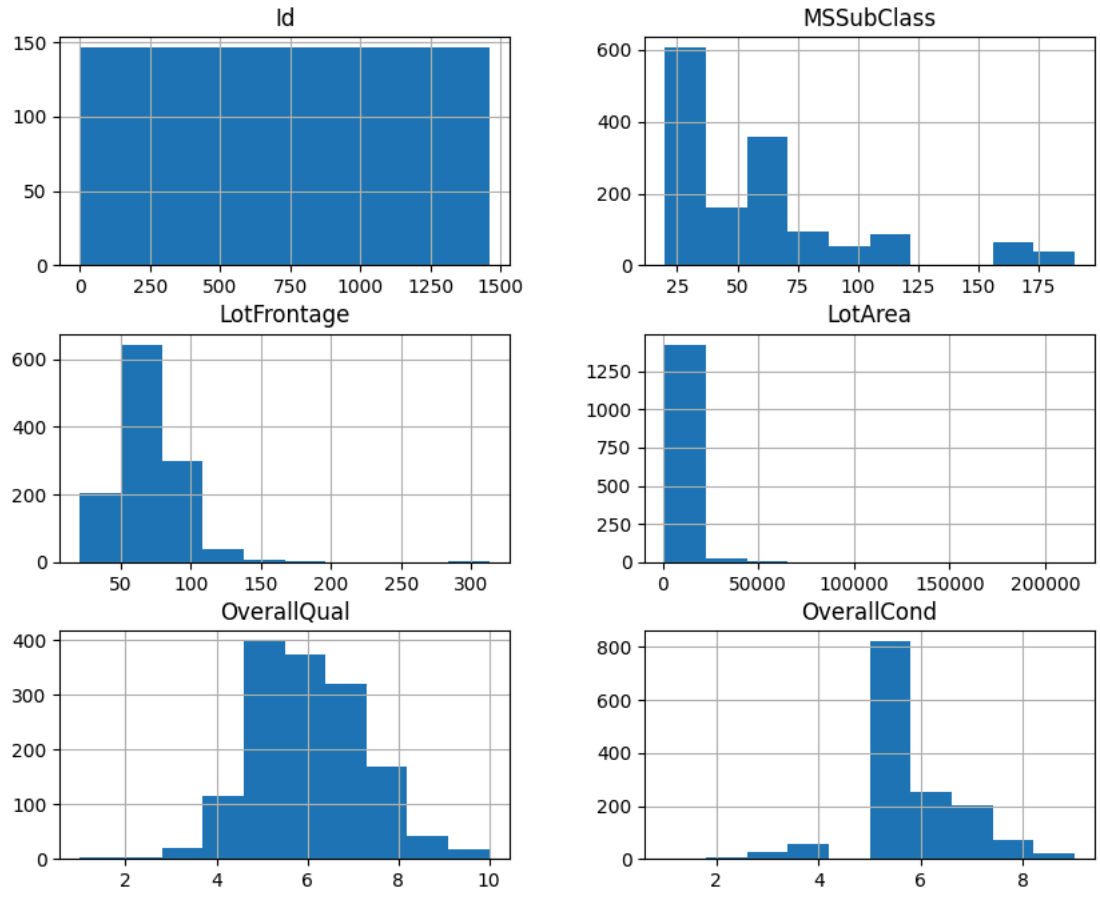
1. MAE = average absolute error
2. RMSE = stronger penalty for big errors
3. RMSE good when big mistakes bad, MAE good for simple read

```
df.describe()
```

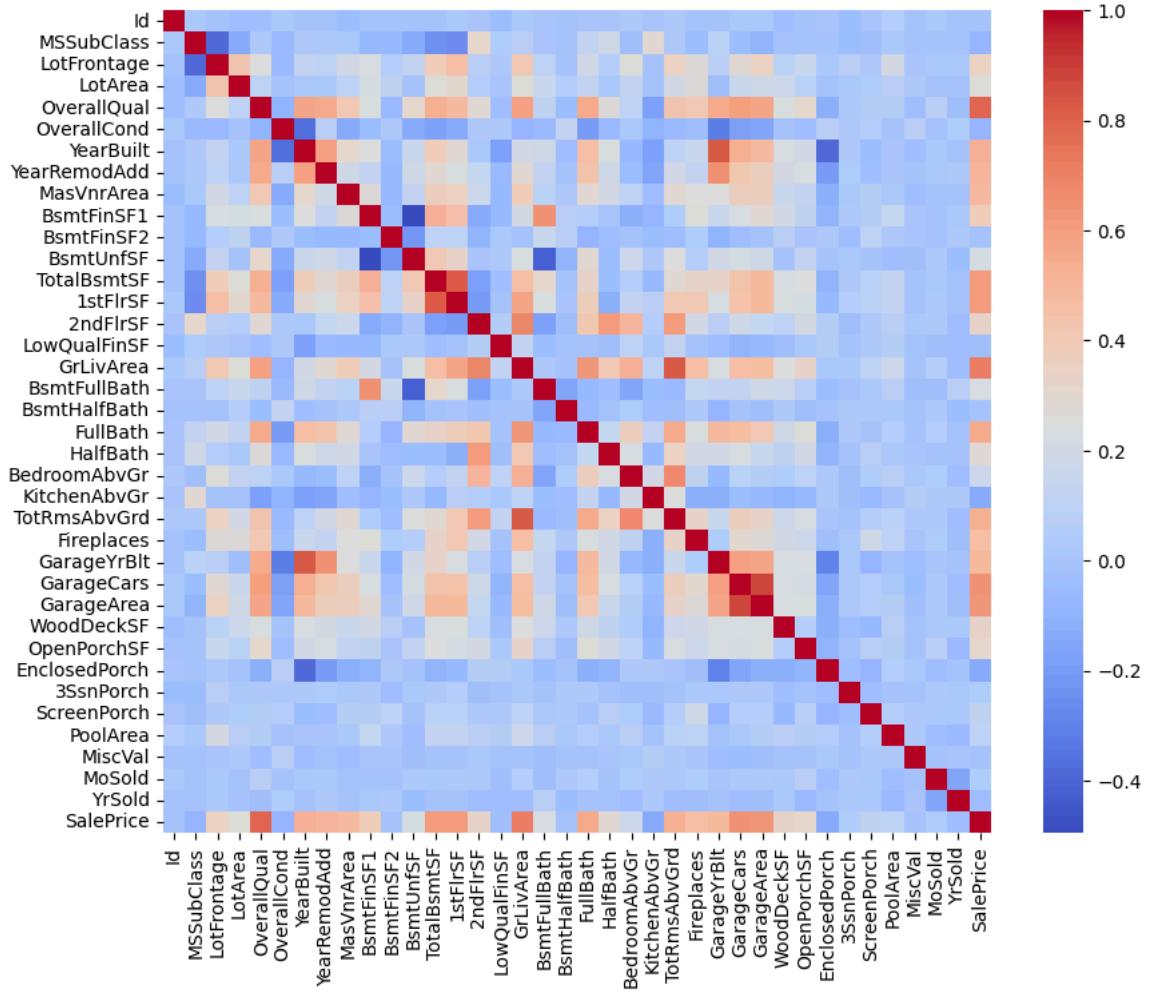
	Id	MSSubClass	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt	YearRemodAdd	MasVnrArea	BsmtFinS
count	1460.000000	1460.000000	1201.000000	1460.000000	1460.000000	1460.000000	1460.000000	1460.000000	1452.000000	1460.0000
mean	730.500000	56.897260	70.049958	10516.828082	6.099315	5.575342	1971.267808	1984.865753	103.685262	443.6397
std	421.610009	42.300571	24.284752	9981.264932	1.382997	1.112799	30.202904	20.645407	181.066207	456.0980
min	1.000000	20.000000	21.000000	1300.000000	1.000000	1.000000	1872.000000	1950.000000	0.000000	0.0000
25%	365.750000	20.000000	59.000000	7553.500000	5.000000	5.000000	1954.000000	1967.000000	0.000000	0.0000
50%	730.500000	50.000000	69.000000	9478.500000	6.000000	5.000000	1973.000000	1994.000000	0.000000	383.5000
75%	1095.250000	70.000000	80.000000	11601.500000	7.000000	6.000000	2000.000000	2004.000000	166.000000	712.2500
max	1460.000000	190.000000	313.000000	215245.000000	10.000000	9.000000	2010.000000	2010.000000	1600.000000	5644.0000

8 rows × 38 columns

```
num_cols = df.select_dtypes(include='number').columns[:6]
df[num_cols].hist(figsize=(10,8))
plt.show()
```



```
corr = df.corr(numeric_only=True)
plt.figure(figsize=(10,8))
sns.heatmap(corr, cmap='coolwarm')
plt.show()
```




```
target = 'SalePrice'
top10 = corr[target].abs().sort_values(ascending=False)[1:11]
top10
```

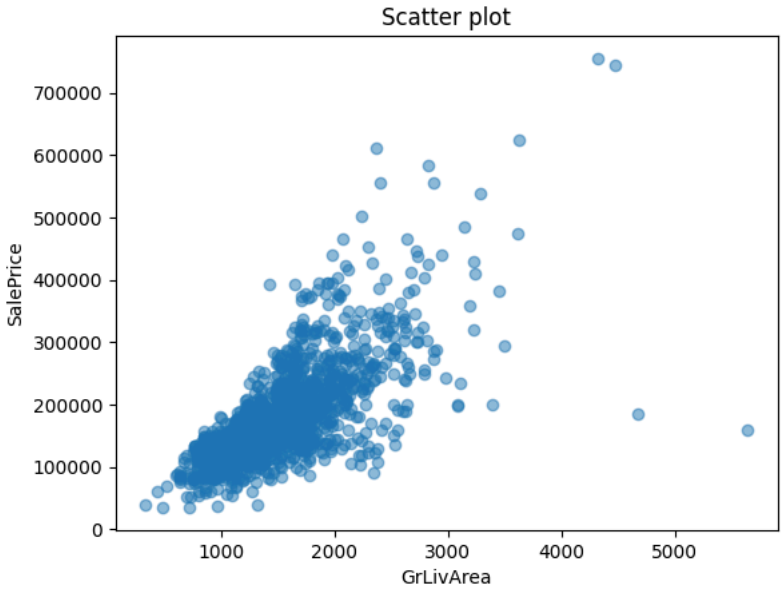
	SalePrice
OverallQual	0.790982
GrLivArea	0.708624
GarageCars	0.640409
GarageArea	0.623431
TotalBsmtSF	0.613581
1stFlrSF	0.605852
FullBath	0.560664
TotRmsAbvGrd	0.533723
YearBuilt	0.522897
YearRemodAdd	0.507101

dtype: float64

Q3.4.1 Answer: top 4 =

1. OverallQual 0.790982
2. GrLivArea 0.708624
3. GarageCars 0.640409
4. GarageArea 0.623431

```
second = top10.index[1]
plt.scatter(df[second], df[target], alpha=0.5)
plt.xlabel(second)
plt.ylabel(target)
plt.title('Scatter plot')
plt.show()
```


Scatter plot
SalePrice
GrLivArea
0
100000
200000
300000
400000
500000
600000
700000
1000
2000
3000
4000
5000