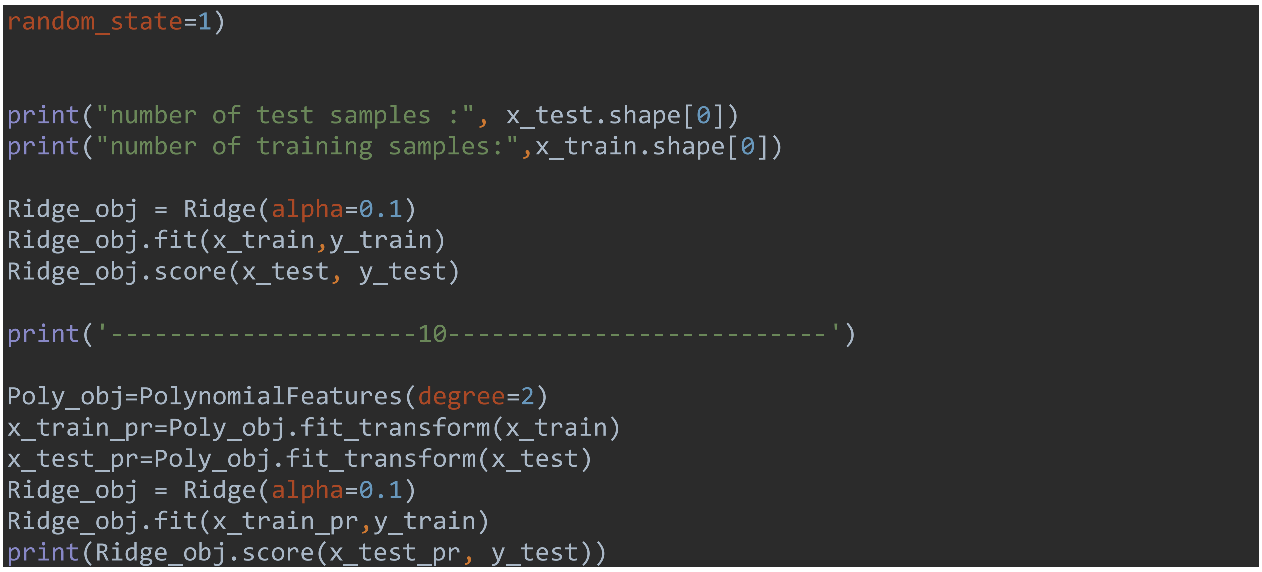
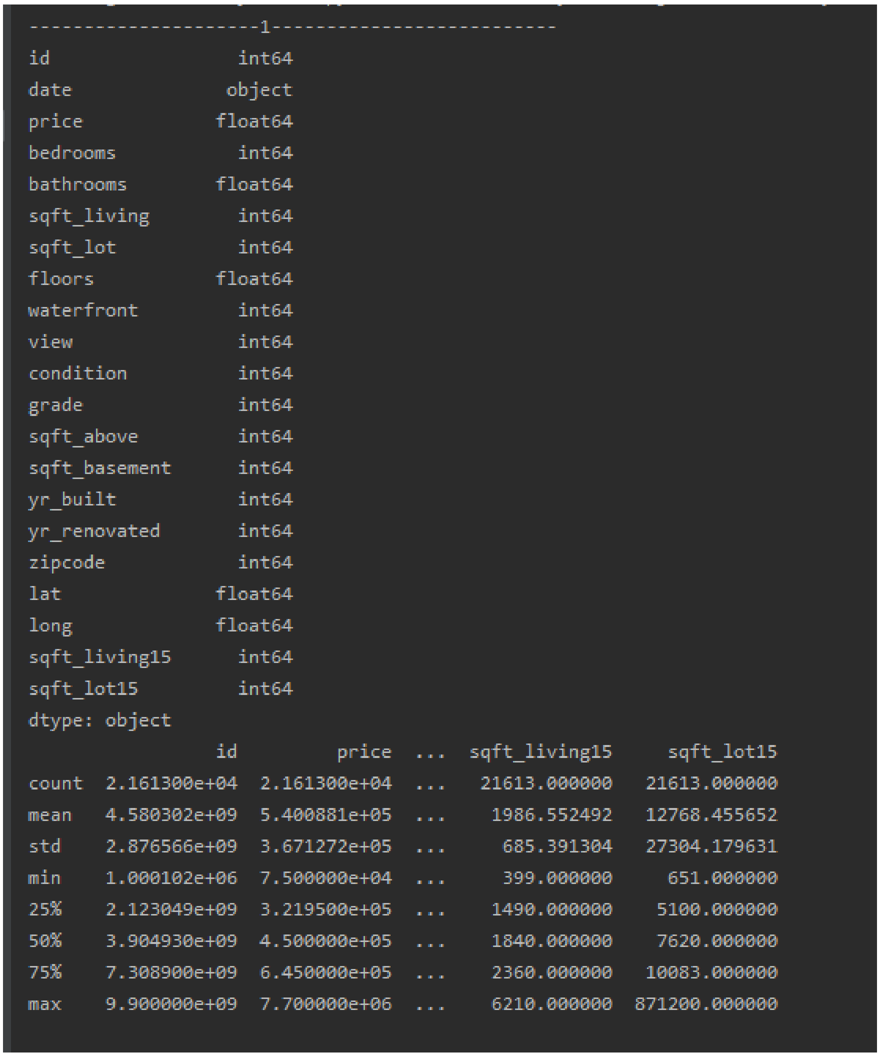
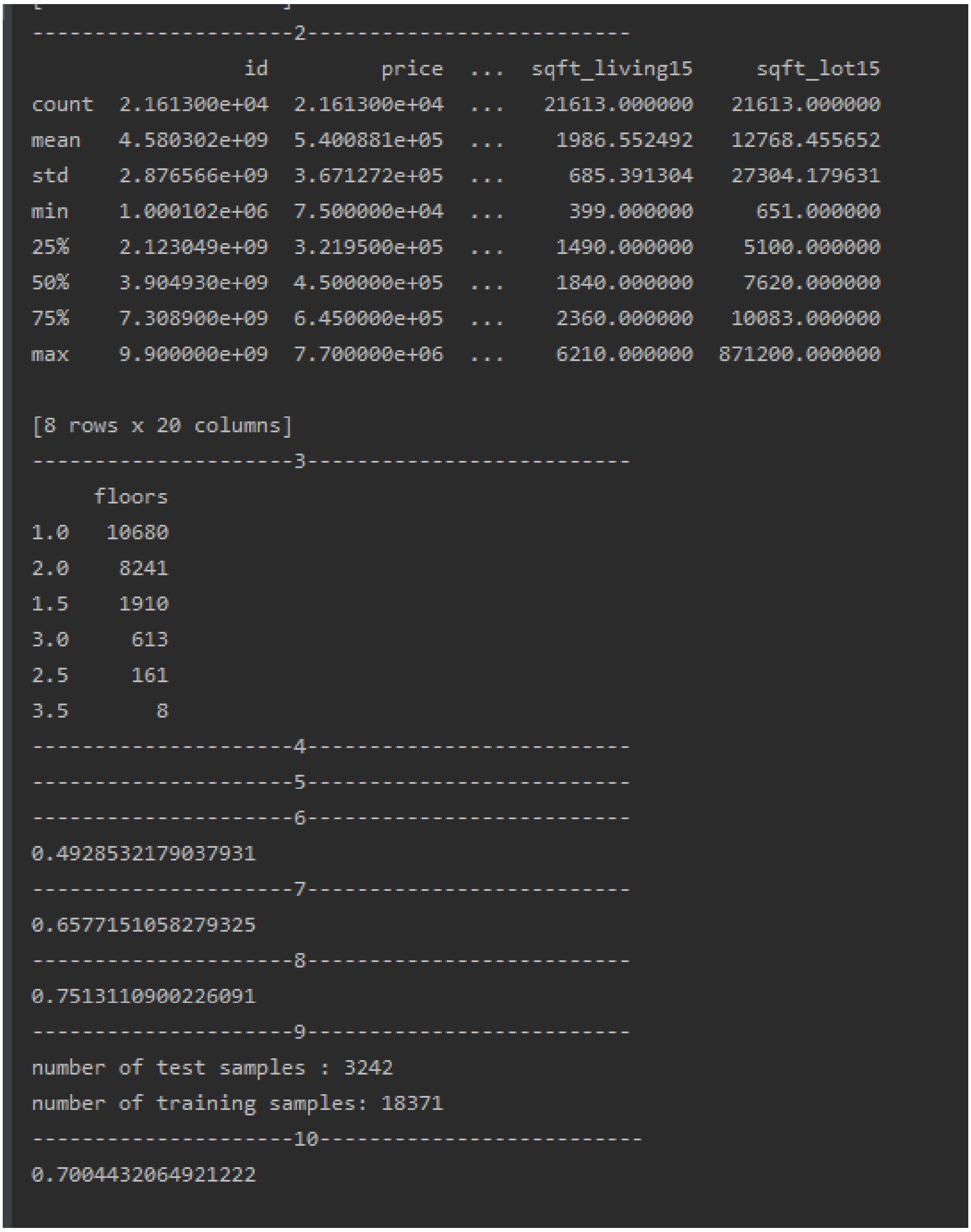
```
from sklearn.linear_model import LinearRegression
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler,PolynomialFeatures
from sklearn.linear_model import Ridge

file_name='kc_house_data.csv'
df=pd.read_csv(file_name)

print('---------------------1--------------------------')
print(df.dtypes)
print(df.describe())
print('---------------------2--------------------------')
# df.drop(["id","Unnamed: 0"], axis = 1,inplace = True)
print(df.describe())
print('---------------------3--------------------------')
print(df['floors'].value_counts().to_frame())
print('---------------------4--------------------------')
sns.boxplot(x="floors", y="price", data=df)
plt.show()
print('---------------------5--------------------------')
sns.regplot(x="sqft_above", y="price", data=df)
plt.ylim(0,)
plt.show()
print('---------------------6--------------------------')
X = df[['sqft_living']]
Y = df['price']
lm = LinearRegression()
lm
lm.fit(X,Y)
print(lm.score(X, Y))
print('---------------------7--------------------------')

features =["floors", "waterfront","lat" ,"bedrooms" ,"sqft_basement" ,"view"
,"bathrooms","sqft_living15","sqft_above","grade","sqft_living"]
Multi_X = df[features]
lm1 = LinearRegression()
lm1
lm1.fit(Multi_X,Y)
print(lm1.score(Multi_X, Y))
print('---------------------8--------------------------')
Input=[('scale',StandardScaler()),('polynomial',
PolynomialFeatures(include_bias=False)),('model',LinearRegression())]
pipe=Pipeline(Input)
pipe.fit(Multi_X,Y)
print(pipe.score(Multi_X,Y))


print('---------------------9--------------------------')

from sklearn.model_selection import cross_val_score
from sklearn.model_selection import train_test_split

features =["floors", "waterfront","lat" ,"bedrooms" ,"sqft_basement" ,"view"
,"bathrooms","sqft_living15","sqft_above","grade","sqft_living"]
X = df[features ]
Y = df['price']

x_train, x_test, y_train, y_test = train_test_split(X, Y, test_size=0.15,
```



```
random_state=1)


print("number of test samples :", x_test.shape[0])
print("number of training samples:",x_train.shape[0])

Ridge_obj = Ridge(alpha=0.1)
Ridge_obj.fit(x_train,y_train)
Ridge_obj.score(x_test, y_test)

print('--------------------10------------------------')

Poly_obj=PolynomialFeatures(degree=2)
x_train_pr=Poly_obj.fit_transform(x_train)
x_test_pr=Poly_obj.fit_transform(x_test)
Ridge_obj = Ridge(alpha=0.1)
Ridge_obj.fit(x_train_pr,y_train)
print(Ridge_obj.score(x_test_pr, y_test))
```

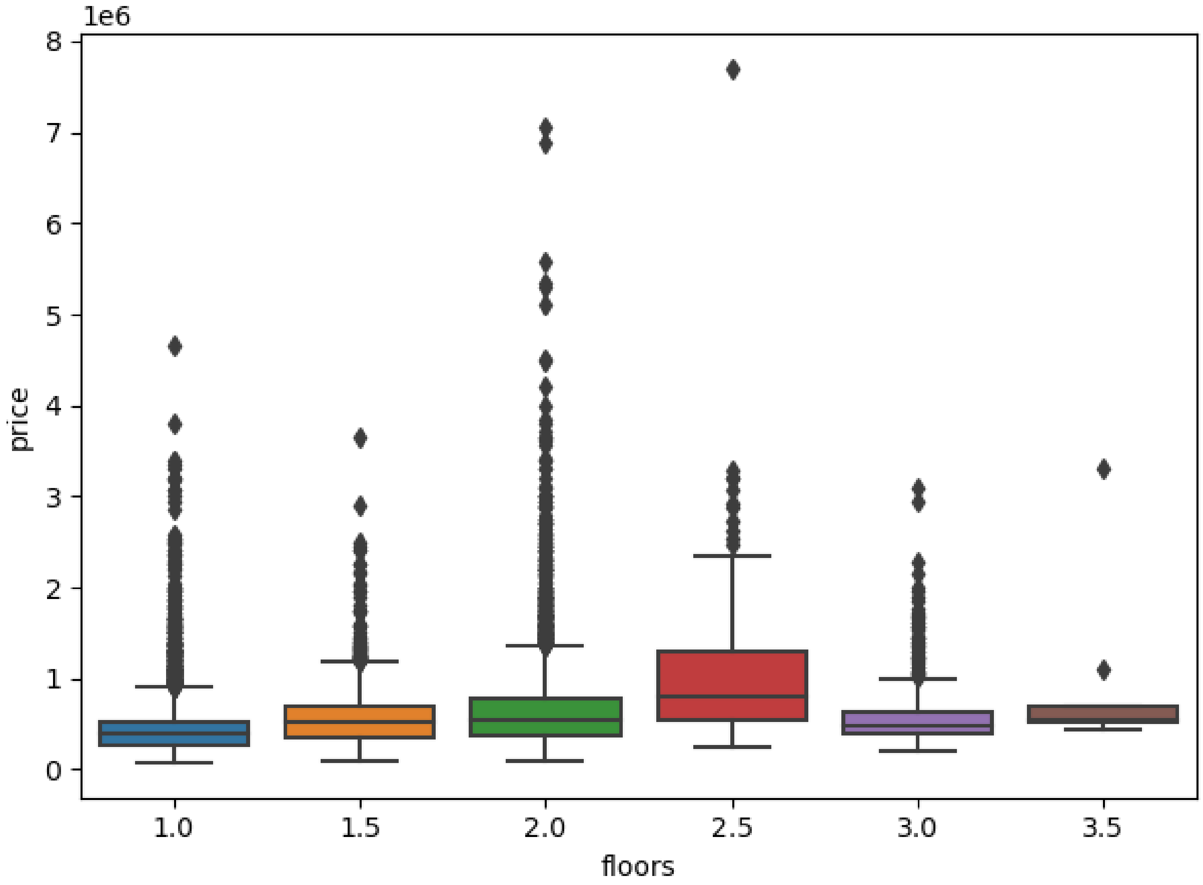


```
---------------------1-------------------------
id                 int64
date              object
price            float64
bedrooms           int64
bathrooms        float64
sqft_living        int64
sqft_lot           int64
floors           float64
waterfront         int64
view               int64
condition          int64
grade              int64
sqft_above         int64
sqft_basement      int64
yr_built           int64
yr_renovated       int64
zipcode            int64
lat              float64
long             float64
sqft_living15      int64
sqft_lot15         int64
dtype: object
                 id         price  ...  sqft_living15     sqft_lot15
count  2.161300e+04  2.161300e+04  ...   21613.000000   21613.000000
mean   4.580302e+09  5.400881e+05  ...    1986.552492   12768.455652
std    2.876566e+09  3.671272e+05  ...     685.391304   27304.179631
min    1.000102e+06  7.500000e+04  ...     399.000000     651.000000
25%    2.123049e+09  3.219500e+05  ...    1490.000000    5100.000000
50%    3.904930e+09  4.500000e+05  ...    1840.000000    7620.000000
75%    7.308900e+09  6.450000e+05  ...    2360.000000   10083.000000
max    9.900000e+09  7.700000e+06  ...    6210.000000  871200.000000
```

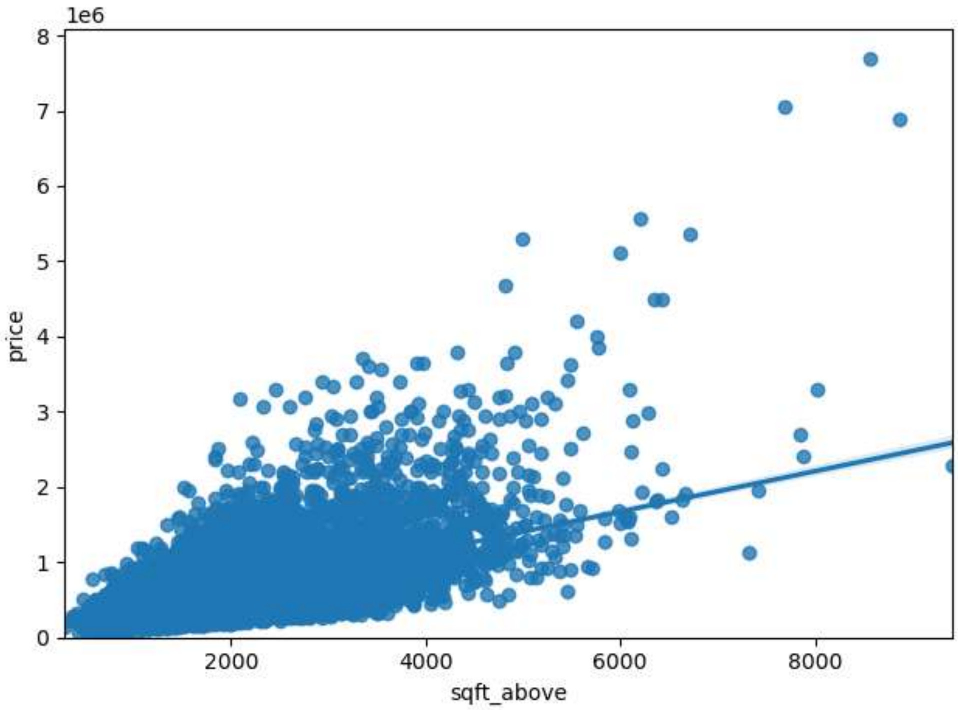


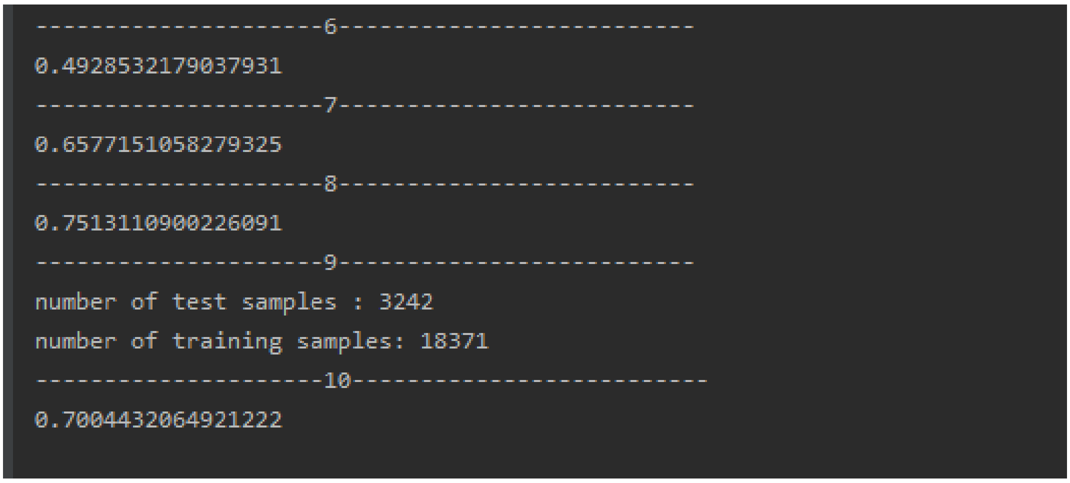
```
---------------------2--------------------------
                 id         price  ...  sqft_living15     sqft_lot15
count  2.161300e+04  2.161300e+04  ...   21613.000000   21613.000000
mean   4.580302e+09  5.400881e+05  ...    1986.552492   12768.455652
std    2.876566e+09  3.671272e+05  ...     685.391304   27304.179631
min    1.000102e+06  7.500000e+04  ...     399.000000     651.000000
25%    2.123049e+09  3.219500e+05  ...    1490.000000    5100.000000
50%    3.904930e+09  4.500000e+05  ...    1840.000000    7620.000000
75%    7.308900e+09  6.450000e+05  ...    2360.000000   10083.000000
max    9.900000e+09  7.700000e+06  ...    6210.000000  871200.000000

[8 rows x 20 columns]
---------------------3--------------------------
     floors
1.0   10680
2.0    8241
1.5    1910
3.0     613
2.5     161
3.5       8
---------------------4--------------------------
---------------------5--------------------------
---------------------6--------------------------
0.4928532179037931
---------------------7--------------------------
0.6577151058279325
---------------------8--------------------------
0.7513110900226091
---------------------9--------------------------
number of test samples : 3242
number of training samples: 18371
---------------------10--------------------------
0.7004432064921222
```

---------------------4-------------------------


1e6
8
7
6
5
4
3
2
1
0
price
1.0
1.5
2.0
2.5
3.0
3.5
floors

---------------------5--------------------------


1e6
8
7
6
5
4
3
2
1
0
price
2000
4000
6000
8000
sqft_above



```
---------------------6--------------------------
0.4928532179037931
---------------------7--------------------------
0.6577151058279325
---------------------8--------------------------
0.7513110900226091
---------------------9--------------------------
number of test samples : 3242
number of training samples: 18371
---------------------10--------------------------
0.7004432064921222
```