In [1]:
```python
import pandas as pd
import numpy as np
df = pd.read_csv(
    filepath_or_buffer = "C:\\Users\\knc5576\\Downloads\\output.csv",
    engine = 'pyarrow',
    dtype = {
    'car_ID': str,
    'symboling': float,
    'doornumber': float,
    'wheelbase': float,
    'carlength': float,
    'carwidth': float,
    'carheight': float,
    'curbweight': float,
    'cylindernumber': float,
    'enginesize': float,
    'boreratio': float,
    'stroke': float,
    'compressionratio': float,
    'horsepower': float,
    'peakrpm': float,
    'citympg': float,
    'highwaympg': float,
    'price': float,
    'TrainTest': str,
    'CarName_1_low': float,
    'CarName_2_moderate': float,
    'CarName_3_high': float,
    'drivewheel_1_low': float,
    'drivewheel_2_moderate': float,
    'drivewheel_3_high': float,
    'enginelocation_2_moderate': float,
    'enginelocation_3_high': float,
    'aspiration_2_moderate': float,
    'aspiration_3_high': float,
    'fuelsystem_1_low': float,
    'fuelsystem_2_moderate': float,
    'fuelsystem_3_high': float,
    'enginetype_2_moderate': float,
    'enginetype_3_high': float,
    'fueltype_2_moderate': float,
    'fueltype_3_high': float,
    'carbody_1_low': float,
    'carbody_2_moderate': float,
    'carbody_3_high': float
    }
)
```

In [2]:
```python
X_Train = df.loc[df['TrainTest'] == 'Train'].drop(columns = ['car_ID','TrainTest','price'])
y_Train = df.loc[df['TrainTest'] == 'Train']['price']
```

In [3]:
```python
from sklearn.linear_model import LassoCV, LinearRegression

# Initialize LassoCV
LassoCV_price = LassoCV(
    cv=5,
    random_state=823,
    max_iter=10000
).fit(
    X = X_Train,
    y = y_Train
)
print(
    "Fraction of kept predictors: " + np.mean(LassoCV_price.coef_ != 0).astype(str)
)
X_Train.columns[(LassoCV_price.coef_ != 0)]
list_features = [
 'symboling',
 'doornumber',
 'wheelbase',
 'carlength',
 'carwidth',
 'carheight',
 'curbweight',
 'cylindernumber',
 'enginesize',
 'boreratio',
 'stroke',
 'compressionratio',
 'horsepower',
 'peakrpm',
 'citympg',
 'highwaympg',
 'CarName_1_low',
 'drivewheel_1_low',
 'enginelocation_3_high',
 'aspiration_3_high',
 'fuelsystem_3_high',
 'enginetype_2_moderate',
 'fueltype_2_moderate',
 'carbody_1_low',
   ]
X_Train.columns[(LassoCV_price.coef_ != 0)]
```

```
Fraction of kept predictors: 0.5555555555555556
```

Out[3]:
```
Index(['symboling', 'doornumber', 'wheelbase', 'carwidth', 'carheight',
       'curbweight', 'cylindernumber', 'enginesize', 'boreratio', 'stroke',
       'compressionratio', 'horsepower', 'citympg', 'CarName_1_low',
       'CarName_3_high', 'enginelocation_2_moderate', 'enginelocation_3_high',
       'fuelsystem_2_moderate', 'carbody_2_moderate', 'carbody_3_high'],
      dtype='object')
```

In [4]:
```python
# Fit a LinearRegression model with selected features
LinearRegression_SalePrice = LinearRegression().fit(
    X = X_Train,
    y = y_Train
)
```

In [5]:
```python
# Get fitted values
df['predict'] = LinearRegression_SalePrice.predict(
    X = df[X_Train.columns]
)
```

In [6]:
```python
df_metric = pd.DataFrame({
    'TrainTest' : ['Train','Validation','Test']
}).set_index('TrainTest')
df_metric
```

Out[6]:

TrainTest
Train
Validation
Test

In [7]:
```python
from sklearn.metrics import mean_absolute_error
df_metric['mean_absolute_error'] = [
    mean_absolute_error(
        y_true = df.loc[(df['TrainTest'] == partition)]['price'],
        y_pred = df.loc[(df['TrainTest'] == partition)]['predict']
    ) for partition in ['Train','Validation','Test']
]
df_metric
```

Out[7]:

TrainTest	mean_absolute_error
Train	0.078433
Validation	0.104222
Test	0.199748

In [8]:
```python
from sklearn.metrics import mean_squared_error
df_metric['mean_squared_error'] = [
    mean_squared_error(
        y_true = df.loc[(df['TrainTest'] == partition)]['price'],
        y_pred = df.loc[(df['TrainTest'] == partition)]['predict']
    ) for partition in ['Train','Validation','Test']
]
df_metric
```

Out[8]:

TrainTest	mean_absolute_error	mean_squared_error
Train	0.078433	0.010509
Validation	0.104222	0.020717
Test	0.199748	0.063035

In [9]:
```python
from sklearn.metrics import mean_squared_error
df_metric['root_mean_squared_error'] = [
    mean_squared_error(
        y_true = df.loc[(df['TrainTest'] == partition)]['price'],
        y_pred = df.loc[(df['TrainTest'] == partition)]['predict'],
        squared = False
    ) for partition in ['Train','Validation','Test']
]
df_metric
```

```
c:\Users\knc5576\AppData\Local\Programs\Python\Python313\Lib\site-packages\sklearn\metrics\_regression.py:492: FutureWarning: 'squared' is deprecated in version 1.4 and will be removed in 1.
6. To calculate the root mean squared error, use the function'root_mean_squared_error'.
  warnings.warn(
c:\Users\knc5576\AppData\Local\Programs\Python\Python313\Lib\site-packages\sklearn\metrics\_regression.py:492: FutureWarning: 'squared' is deprecated in version 1.4 and will be removed in 1.
6. To calculate the root mean squared error, use the function'root_mean_squared_error'.
  warnings.warn(
c:\Users\knc5576\AppData\Local\Programs\Python\Python313\Lib\site-packages\sklearn\metrics\_regression.py:492: FutureWarning: 'squared' is deprecated in version 1.4 and will be removed in 1.
6. To calculate the root mean squared error, use the function'root_mean_squared_error'.
  warnings.warn(
```

Out[9]:

TrainTest	mean_absolute_error	mean_squared_error	root_mean_squared_error
Train	0.078433	0.010509	0.102512
Validation	0.104222	0.020717	0.143934
Test	0.199748	0.063035	0.251068

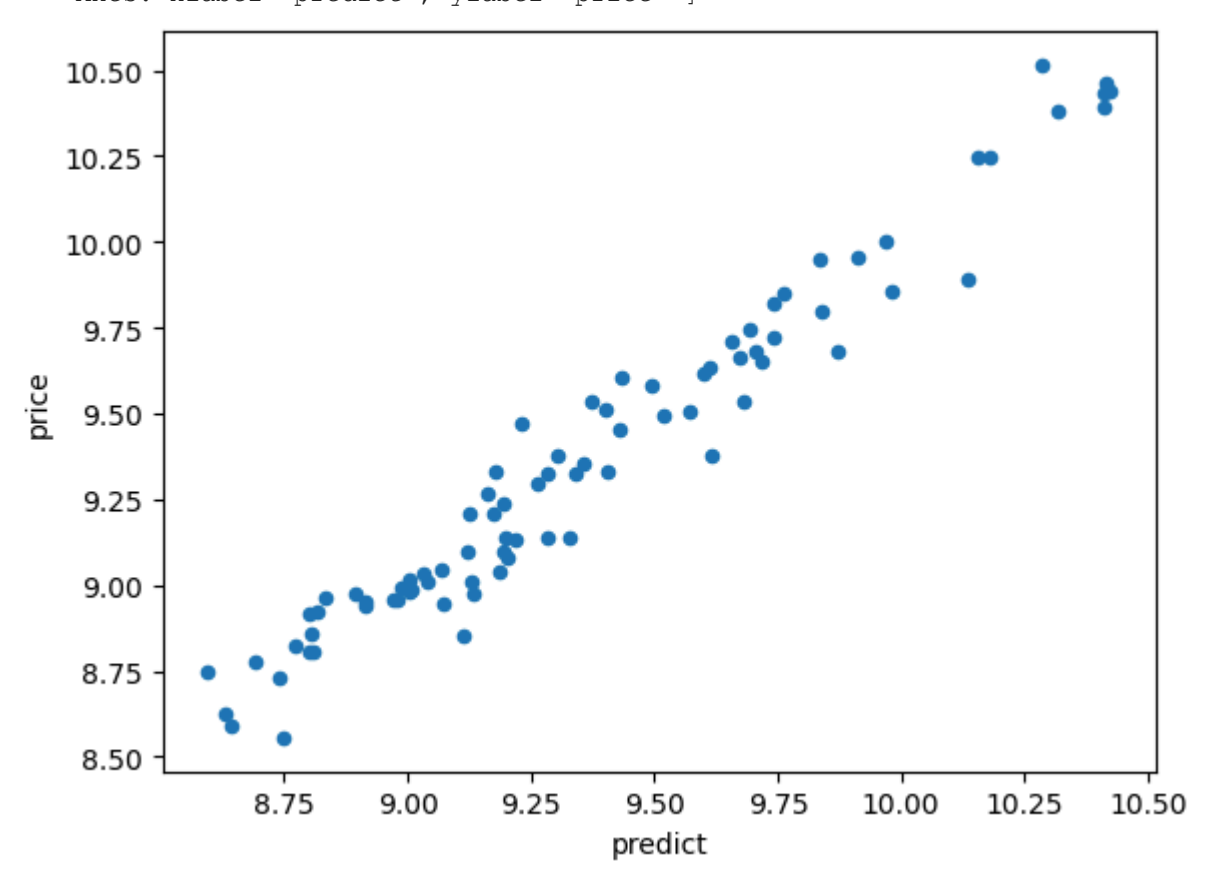
In [10]:
```python
df_metric['corr'] = [
    df.loc[(df['TrainTest'] == partition)][['price','predict']].corr().iloc[0,1] for partition in ['Train','Validation','Test']
]
df_metric
```

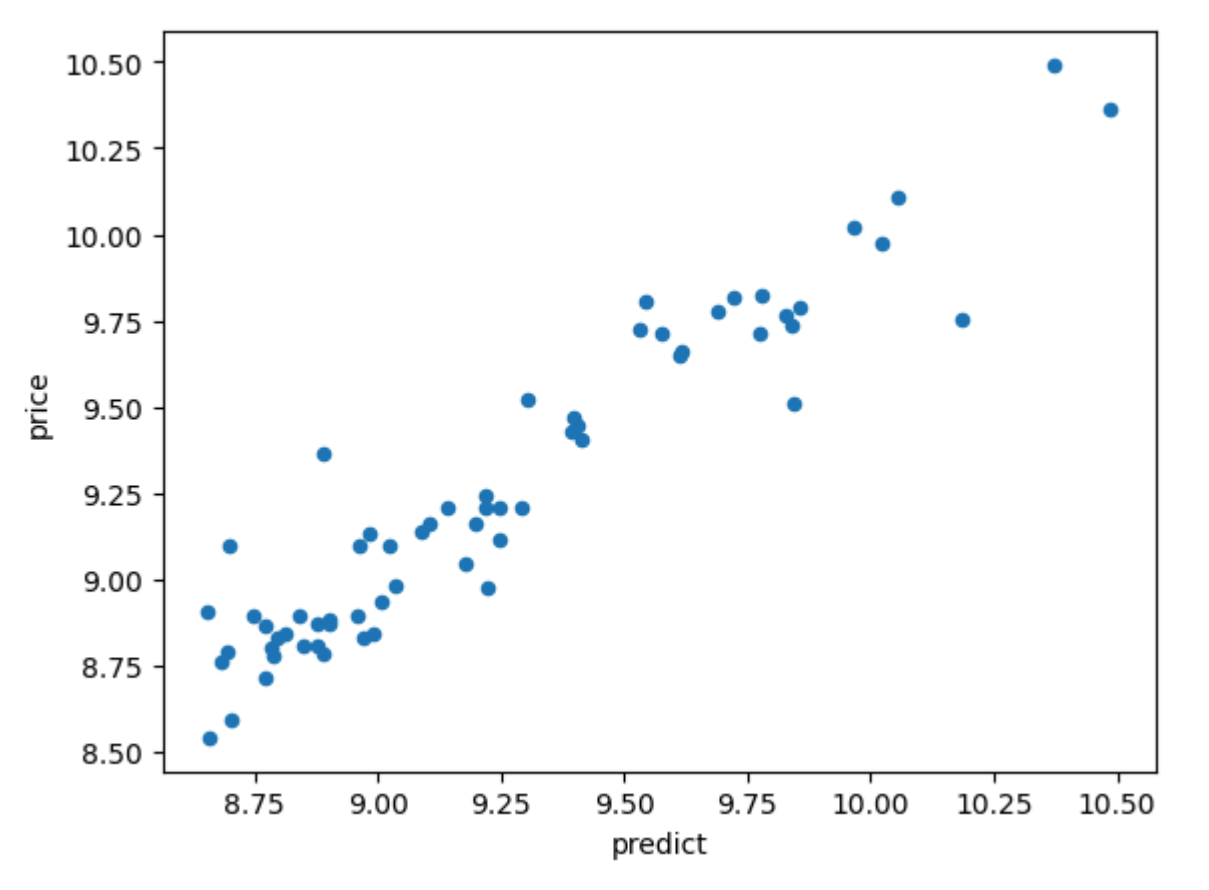
Out[10]:

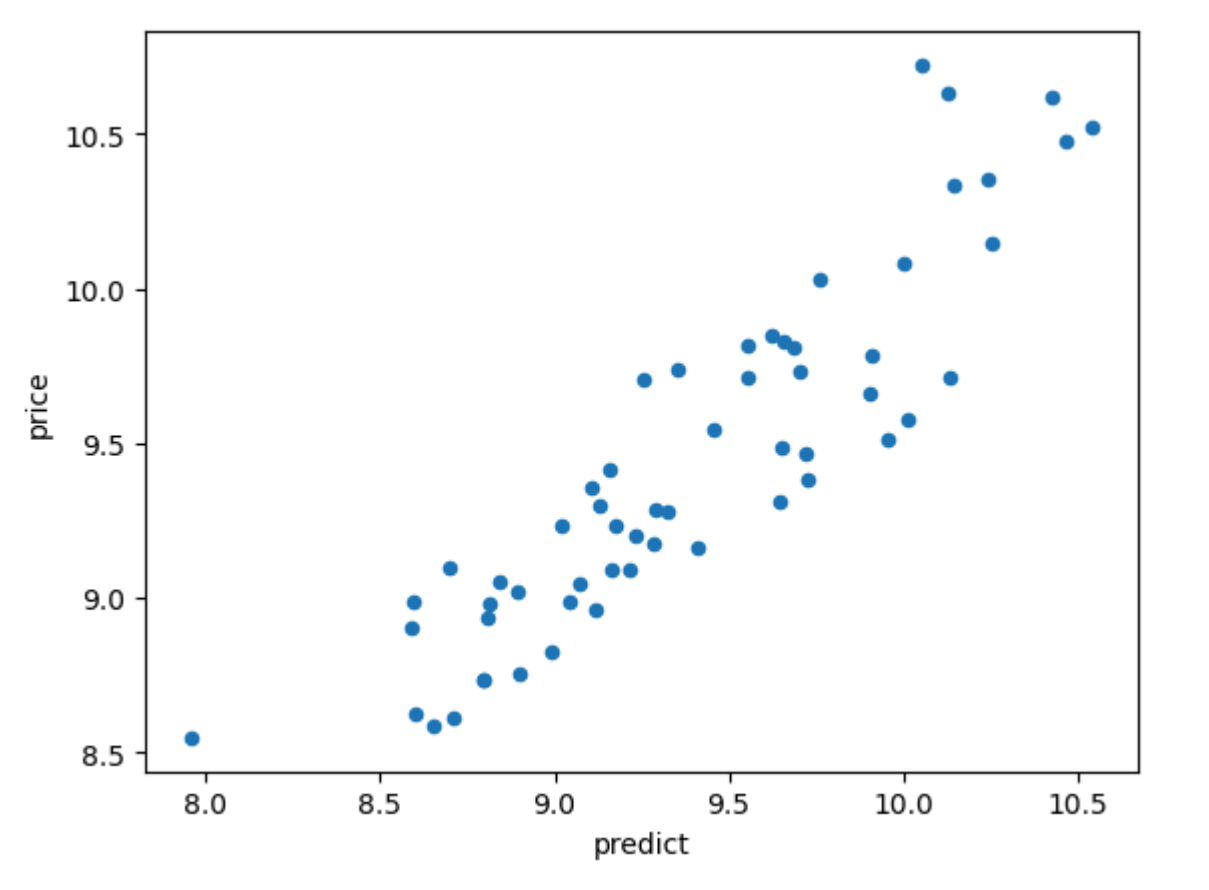
TrainTest	mean_absolute_error	mean_squared_error	root_mean_squared_error	corr
Train	0.078433	0.010509	0.102512	0.977449
Validation	0.104222	0.020717	0.143934	0.950875
Test	0.199748	0.063035	0.251068	0.903215

In [11]:
```python
[
    df.loc[(df['TrainTest'] == partition)][['price','predict']].plot.scatter(y = 'price',x = 'predict') for partition in ['Train','Validation','Test']
]
```

Out[11]:
```
[<Axes: xlabel='predict', ylabel='price'>,
 <Axes: xlabel='predict', ylabel='price'>,
 <Axes: xlabel='predict', ylabel='price'>]
```

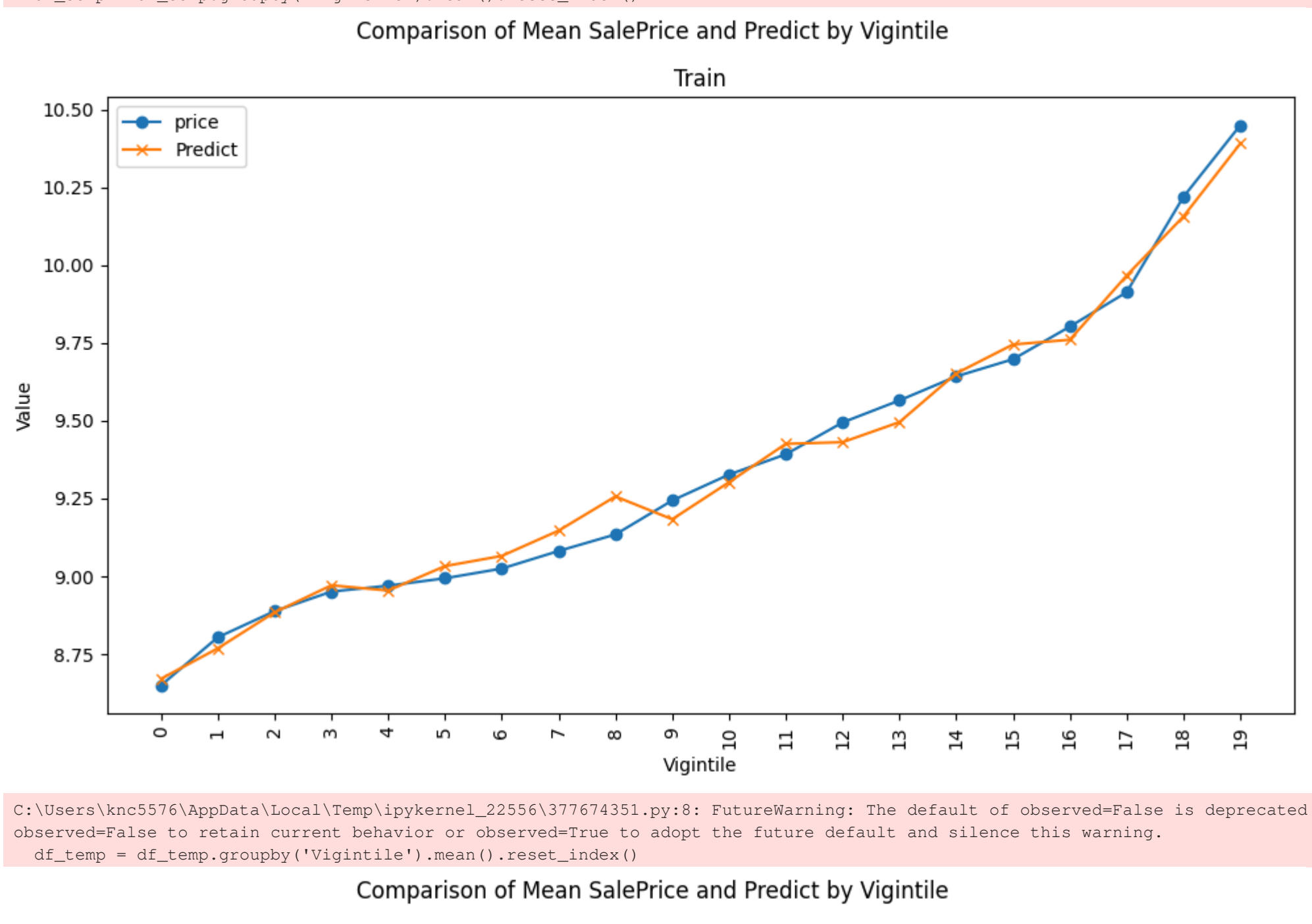




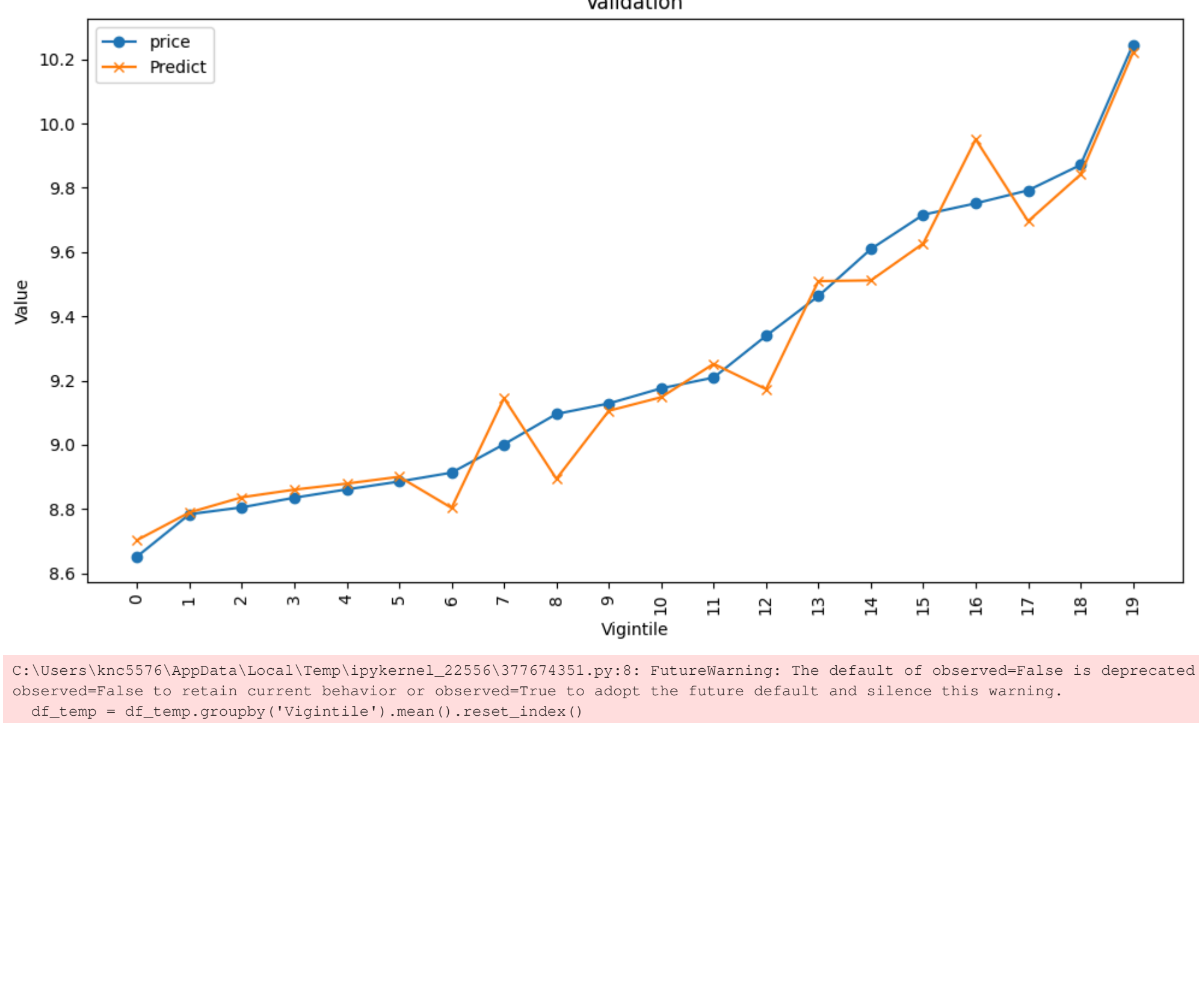


In [12]:
```python
import matplotlib.pyplot as plt
for partition in ['Train','Validation','Test']:
    df_temp = df.loc[(df['TrainTest'] == partition)][['price','predict']]
    df_temp['Vigintile'] = pd.qcut(
        x = df_temp['price'],
        q = 20
    )
    df_temp = df_temp.groupby('Vigintile').mean().reset_index()
    df_temp['Vigintile'] = range(len(df_temp['Vigintile']))
    # Plotting
    plt.figure(figsize=(10, 6))
    plt.plot(df_temp['Vigintile'], df_temp['price'], label='price', marker='o')
    plt.plot(df_temp['Vigintile'].astype(str), df_temp['predict'], label='Predict', marker='x')
    plt.xticks(rotation=90)  # Rotate x-axis labels for better readability
    plt.xlabel('Vigintile')
    plt.ylabel('Value')
    plt.suptitle('Comparison of Mean SalePrice and Predict by Vigintile ')
    plt.title(partition)
    plt.legend()
    plt.tight_layout()  # Adjust layout to make room for rotated x-axis labels
    plt.show()
```

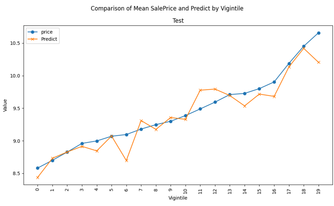
```
C:\Users\knc5576\AppData\Local\Temp\ipykernel_22556\377674351.py:8: FutureWarning: The default of observed=False is deprecated and will be changed to True in a future version of pandas. Pass observed=False to retain current behavior or observed=True to adopt the future default and silence this warning.
  df_temp = df_temp.groupby('Vigintile').mean().reset_index()
```




```
C:\Users\knc5576\AppData\Local\Temp\ipykernel_22556\377674351.py:8: FutureWarning: The default of observed=False is deprecated and will be changed to True in a future version of pandas. Pass observed=False to retain current behavior or observed=True to adopt the future default and silence this warning.
  df_temp = df_temp.groupby('Vigintile').mean().reset_index()
```




```
C:\Users\knc5576\AppData\Local\Temp\ipykernel_22556\377674351.py:8: FutureWarning: The default of observed=False is deprecated and will be changed to True in a future version of pandas. Pass observed=False to retain current behavior or observed=True to adopt the future default and silence this warning.
  df_temp = df_temp.groupby('Vigintile').mean().reset_index()
```


Comparison of Mean SalePrice and Predict by Vigintile
Test
price
Predict
Value
Vigintile
10.5
10.0
9.5
9.0
8.5
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19