DSI Project II

DSIR-1116 - Registration Challenge
Test Automation Response

Oscar Caraballo - December 14 - 2020

Introduction

The following study was based on Kagel House Dataset used to predict a specific parameter such as SalePrice estimates the importance of Test Automation in performing structured tasks

Methodology

- ❋ Perform a Data Cleaning on all Columns of Data Set relevant to predictions.
- ❋ Create a list with all the Features to be explored.
- ❋ Create a Var to be predicted by our Features.
- ❋ Call Method project_two_lr(list, var)
- ❋ Method will create X_train, X_test, y_train, y_test parameters.
- ❋ Method will Initiate Linear Regression Model
- ❋ Method will Fit Model using X,y data
- ❋ Method will calculate Coeffs for M and B or B0 and B1.
- ❋ Method will calculate Score for Feature.
- ❋ Method will calculate Predictions on Test Data.
- ❋ From there we can select the Highest Score Features - To be implemented in the future.
- ❋ The idea is to generate Score for all the Features and select the ones with Highest Score.
- ❋ The second step is to find the combinations of Features that Maximizes the R2 Score.

Methodology

```
# features = ['Lot Area','Overall Qual','Pool Area','Yr Sold','MS SubClass']
# y = df['SalePrice']

def project_two_lr(feature_list, predict_var):

   # Input Parameters
   X = df_train[feature_list]
   y = df_train[predict_var]

   # Split for Train and Test Model
   X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 2020)

   # Liner Model Instantiation
   lr = LinearRegression()

   # Model Fit
   lr.fit(X_train,y_train)
```

Methodology

```
 # Coeffs for Slope of M
  lr.coef_
  print(f'### COEFFS M #### {lr.coef_}')

  # Coeffs for Slope of B
  lr.intercept_
  print(f'### COEFFS B #### {lr.intercept_}')

  # Score Calculation
  score = lr.score(X_test,y_test)
  print(f'### SCORE #### {lr.score(X_test,y_test)}')

  return score

print(project_two_lr(['Lot Area','Overall Qual','Pool Area','Yr Sold','MS SubClass'], 'SalePrice'))
```

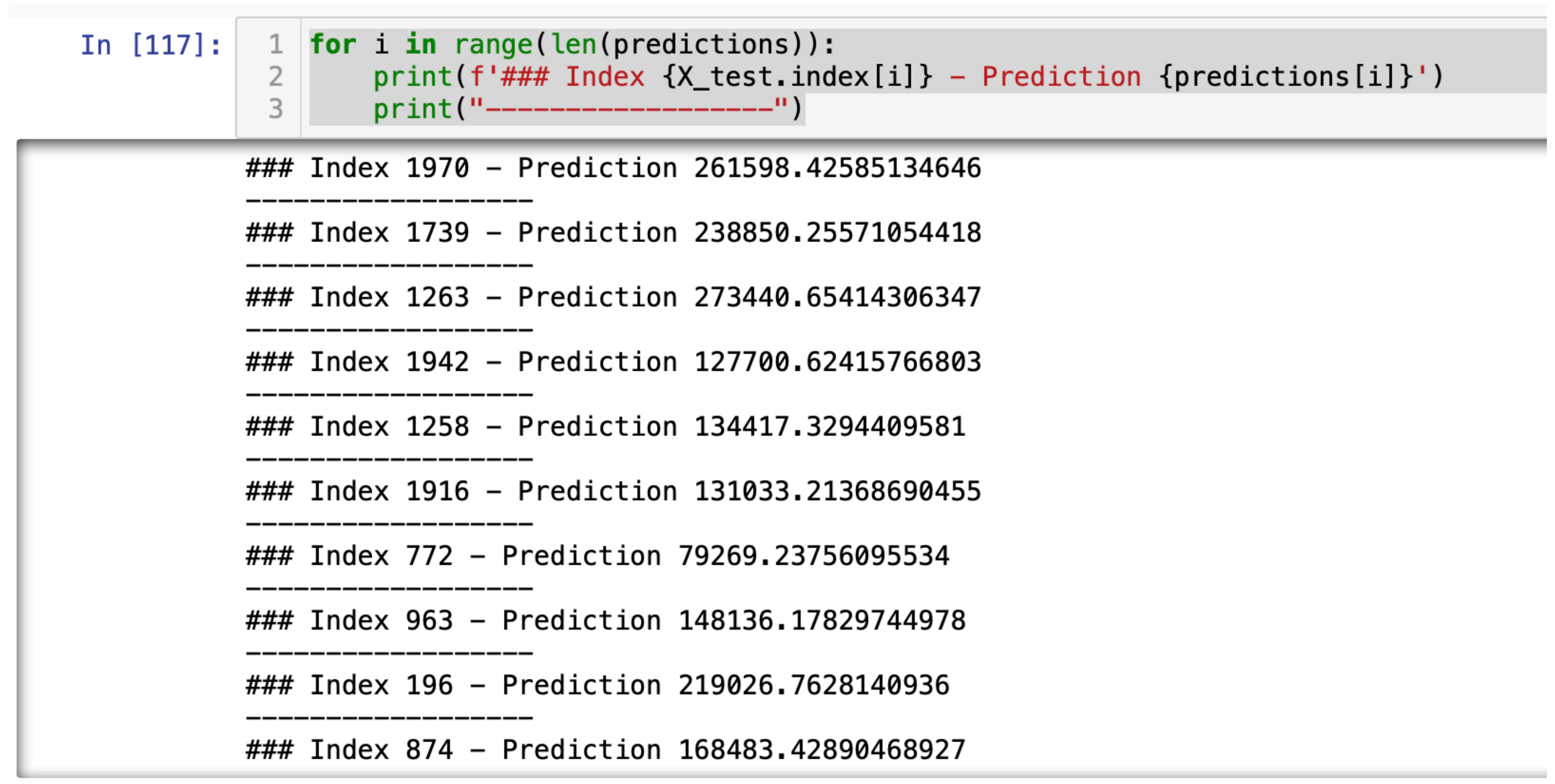
Findings

```
### COEFFS M #### [ 2.08336256e+00  4.34217305e+04 -5.96901459e+00  1.60450202e+02
 -1.31650182e+02]
### COEFFS B #### -419709.43981279276
### SCORE #### 0.7109823041075626
0.7109823041075626
```

# Generating Predictions

df_test['SalePrice'] = lr.predict(X_test)

predictions = lr.predict(X_test)

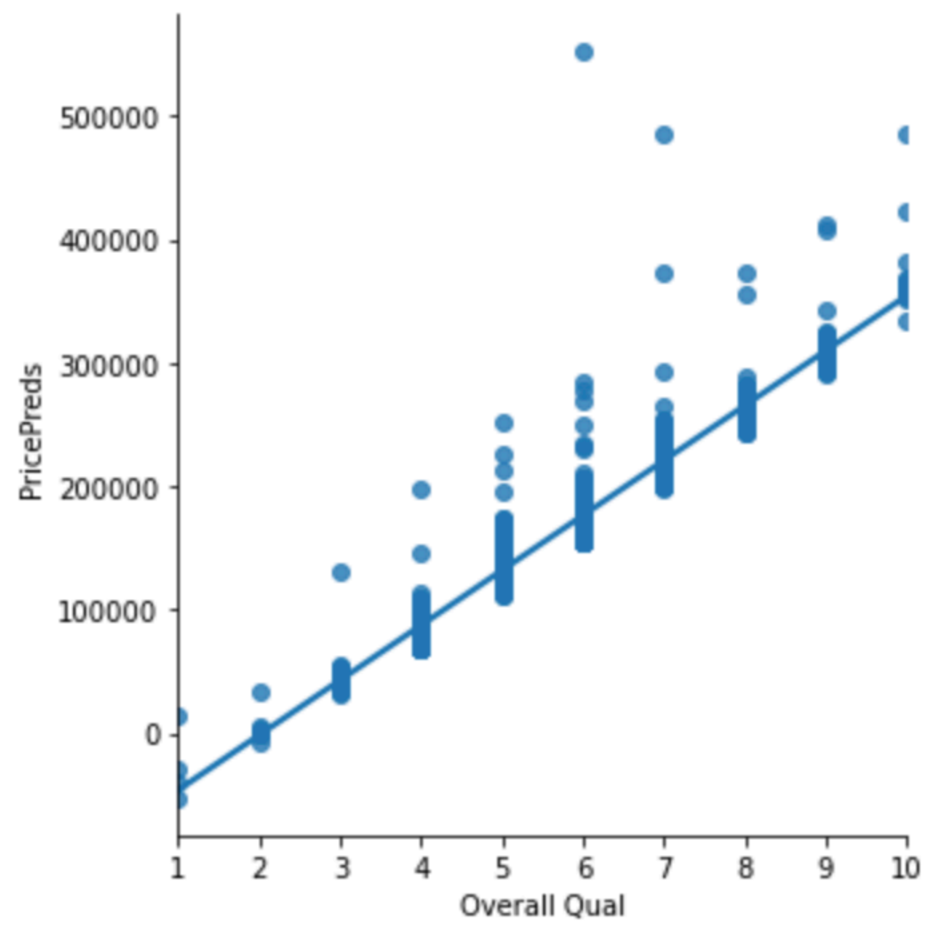


```
In [117]:
for i in range(len(predictions)):
    print(f'### Index {X_test.index[i]} - Prediction {predictions[i]}')
    print("------------------")
```

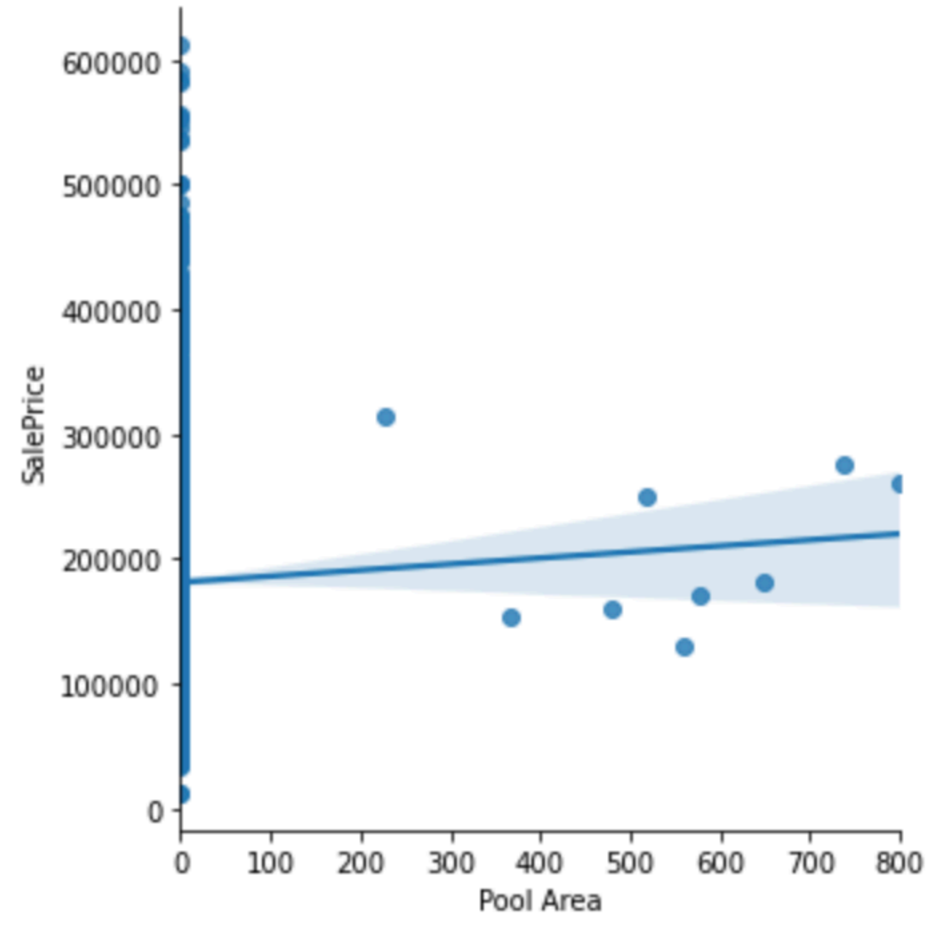
```
### Index 1970 - Prediction 261598.42585134646
------------------
### Index 1739 - Prediction 238850.25571054418
------------------
### Index 1263 - Prediction 273440.65414306347
------------------
### Index 1942 - Prediction 127700.62415766803
------------------
### Index 1258 - Prediction 134417.3294409581
------------------
### Index 1916 - Prediction 131033.21368690455
------------------
### Index 772 - Prediction 79269.23756095534
------------------
### Index 963 - Prediction 148136.17829744978
------------------
### Index 196 - Prediction 219026.7628140936
------------------
### Index 874 - Prediction 168483.42890468927
```

Findings

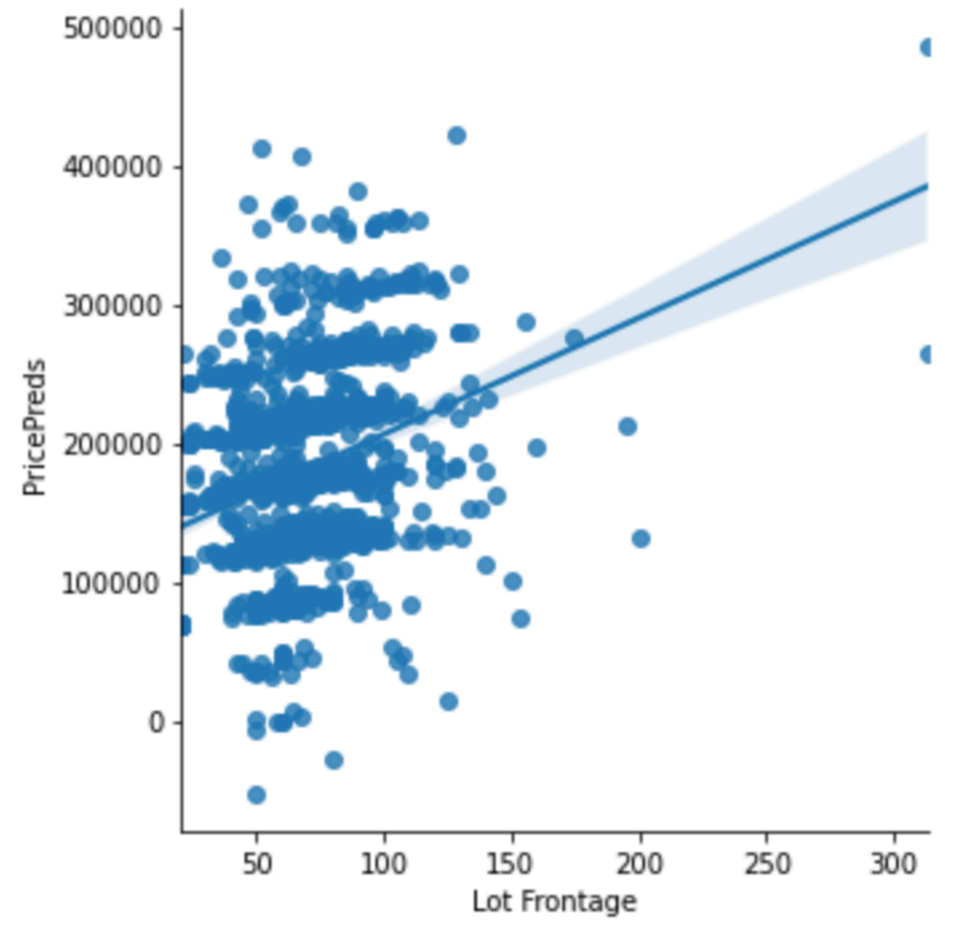
```python
sns.lmplot(x='Overall Qual', y='PricePreds', data=df_train)
plt.show()
```



```python
sns.lmplot(x='Pool Area', y='SalePrice', data=df_train)
plt.show()
```



```python
sns.lmplot(x='Lot Frontage', y='PricePreds', data=df_train)
plt.show()
```



Conclusion

A project like this will enormously benefit from Test Automation. Stages such as Cleaning the Data. Estimating the best set of Features - Evaluating Individuals plus Combinations of Parameters to Find the Highest Score for specific Models. Calculate Predictions and Populate resulting Data Frame.

```
print(project_two_lr(['Lot Area','Overall Qual','Pool Area','Yr Sold','MS
SubClass'], ‘SalePrice'))
```

QA

Thank You!