Question 1

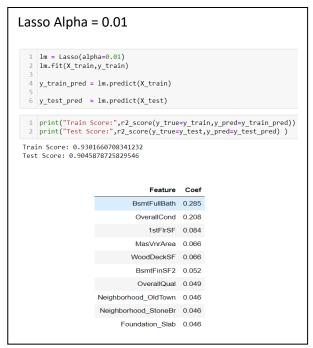
What is the optimal value of alpha for ridge and lasso regression? What will be the changes in the model if you choose double the value of alpha for both ridge and lasso? What will be the most important predictor variables after the change is implemented?

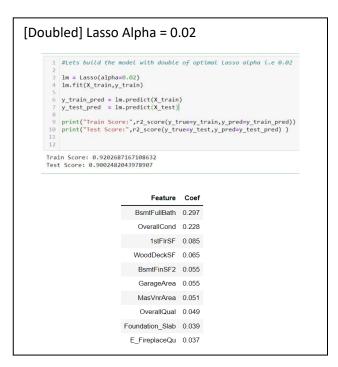
Answer:

Optimal value of alpha for Ridge and Lasso regression models are

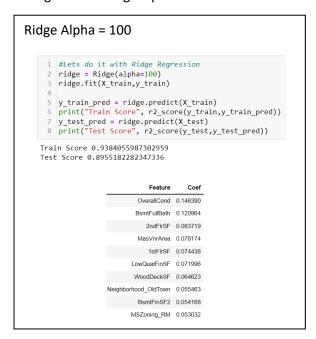
Lasso model: 0.01 Ridge model: 100

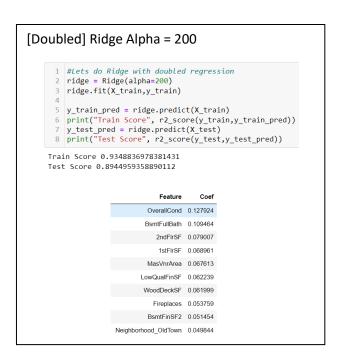
Changes when Lasso alpha is doubled:





Changes when Ridge alpha is doubled:





Question 2

You have determined the optimal value of lambda for ridge and lasso regression during the assignment. Now, which one will you choose to apply and why?

Answer:

- The R-Squared value and other metrics are better with Lasso in our assignment so we choose Lasso over Ridge here.
- Also, Lasso removes the overhead of doing RFE as it automatically removes the insignificant variables by making their coefficients 0.

Question 3

After building the model, you realised that the five most important predictor variables in the lasso model are not available in the incoming data. You will now have to create another model excluding the five most important predictor variables. Which are the five most important predictor variables now?

Answer:

The initial lasso optimal model with top 5 features:

```
1 \mid lm = Lasso(alpha=0.01)
 2 lm.fit(X train,y train)
 4 y train pred = lm.predict(X train)
 6 y_test_pred = lm.predict(X_test)
 7 print("Train Score:",r2_score(y_true=y_train,y_pred=y_train_pred))
 8 print("Test Score:",r2_score(y_true=y_test,y_pred=y_test_pred) )
Train Score: 0.9301660708341232
Test Score: 0.9045878725829546
1 model parameter = list(lm.coef )
 2 model parameter.insert(0,lm.intercept )
 3 model_parameter = [round(x,3) for x in model_parameter]
 4 col = df train.columns
 5 col.insert(0, 'Constant')
 6 lasso_coef = pd.DataFrame(list(zip(col,model_parameter)))
 7 lasso_coef.columns = ['Feature','Coef']
 8 # Significant variables which predict price of house when used lass
 9 lasso_coef.sort_values(by='Coef',ascending=False).head()
        Feature Coef
14 BsmtFullBath 0.285
    OverallCond 0.208
10
       1stFIrSF 0.084
 5 MasVnrArea 0.066
25 WoodDeckSF 0.066
```

Now after removing those 5 features:

```
1 #Removing the above top 5 features from the dataset
 df train=df train.drop(['BsmtFullBath','OverallCond','1stFlrSF','MasVnrArea','WoodDeckSF'],axis='colu
 df test=df test.drop(['BsmtFullBath','OverallCond','1stFlrSF','MasVnrArea','WoodDeckSF'],axis='columne
 1 #Our new data set
 2 X train2 = df train
 3 X_test2 = df_test
 1 \mid lm = Lasso(alpha=0.01)
 2 lm.fit(X train2,y train)
 4 y train pred = lm.predict(X train2)
 6 y test pred = lm.predict(X test2)
 7 print("Train Score:",r2_score(y_true=y_train,y_pred=y_train_pred))
 8 print("Test Score:",r2_score(y_true=y_test,y_pred=y_test_pred) )
Train Score: 0.9255481464734157
Test Score: 0.8975628279003851
 1 model parameter = list(lm.coef )
 2 model parameter.insert(0,lm.intercept )
 3 model_parameter = [round(x,3) for x in model_parameter]
 4 col = df train.columns
 5 col.insert(0, 'Constant')
 6 lasso_coef = pd.DataFrame(list(zip(col,model_parameter)))
 7 lasso_coef.columns = ['Feature','Coef']
 8 # Significant variables which predict price of house when used lasso
 9 lasso_coef.sort_values(by='Coef',ascending=False).head()
        Feature Coef
11 BsmtHalfBath 0.301
    BsmtFinSF1 0.211
8
       2ndFlrSF 0.083
21 OpenPorchSF 0.074
    BsmtFinSF2 0.070
```

Question 4: Continued on next page

Question 4

How can you make sure that a model is robust and generalisable? What are the implications of the same for the accuracy of the model and why?

Answer:

Any model must be generalized so that the resultant model must perform equally well on test data. Also, the model should be working well on all other datasets. These are few metrics to ensure that:

- Model must be kept as simple as possible, complex models are weaker in terms of generalisation.
- Bias-Variance trade-off graph needs be to understood which suggests optimum trade-off for greater generalization
- Model must be scaled and transformed accordingly.
- Model with good R2-Square and less features is preferable than a model with Best R2-squared with too many features. Its more generalisable