Problem Statement -Part 2

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

1. The optimal value of alpha for ridge and lasso regression

   Ridge Alpha 1

   lasso Alpha 10

Ridge Regression

In [97]:

```
#Change the alpha value from 1 to 2
alpha = 3
ridge2 = Ridge(alpha=alpha)
ridge2.fit(X_train1, y_train)
```

Out[97]:

```
Ridge
```

```
Ridge(alpha=3)
```

In [98]:

```
# Lets calculate some metrics such as R2 score, RSS and RMSE
y_pred_train = ridge2.predict(X_train1)
y_pred_test = ridge2.predict(X_test1)

metric2 = []
r2_train_lr = r2_score(y_train, y_pred_train)
print(r2_train_lr)
metric2.append(r2_train_lr)

r2_test_lr = r2_score(y_test, y_pred_test)
print(r2_test_lr)
metric2.append(r2_test_lr)

rss1_lr = np.sum(np.square(y_train - y_pred_train))
print(rss1_lr)
metric2.append(rss1_lr)

rss2_lr = np.sum(np.square(y_test - y_pred_test))
print(rss2_lr)
```

```
metric2.append(rss2_lr)

mse_train_lr = mean_squared_error(y_train, y_pred_train)
print(mse_train_lr)
metric2.append(mse_train_lr**0.5)

mse_test_lr = mean_squared_error(y_test, y_pred_test)
print(mse_test_lr)
metric2.append(mse_test_lr**0.5)

#Alpha 1
#R2score(train) 0.884340040460635
#R2score(test)  0.869613280468847
```

```
0.8797315810932456
0.8710282148272899
607995142958.1411
320928407278.46216
680845624.8131479
729382743.8146868
```

2. R2score on training data has decreased but it has increased on testing data

Lasso

In [99]:

```
#Changed alpha 10 to 20
alpha =20
lasso20 = Lasso(alpha=alpha)
lasso20.fit(X_train1, y_train)
```

Out[99]:

```
Lasso
```

```
Lasso(alpha=20)
```

In [100]:

```
# Lets calculate some metrics such as R2 score, RSS and RMSE
y_pred_train = lasso20.predict(X_train1)
y_pred_test = lasso20.predict(X_test1)

metric3 = []
r2_train_lr = r2_score(y_train, y_pred_train)
print(r2_train_lr)
metric3.append(r2_train_lr)
```

```
r2_test_lr = r2_score(y_test, y_pred_test)
print(r2_test_lr)
metric3.append(r2_test_lr)

rss1_lr = np.sum(np.square(y_train - y_pred_train))
print(rss1_lr)
metric3.append(rss1_lr)

rss2_lr = np.sum(np.square(y_test - y_pred_test))
print(rss2_lr)
metric3.append(rss2_lr)

mse_train_lr = mean_squared_error(y_train, y_pred_train)
print(mse_train_lr)
metric3.append(mse_train_lr**0.5)

mse_test_lr = mean_squared_error(y_test, y_pred_test)
print(mse_test_lr)
metric3.append(mse_test_lr**0.5)

#R2score at alpha-10
#0.8859222400899005
#0.8646666084570094
```

```
0.8854019697956436
0.8670105921065014
579329522996.7144
330925704432.26794
648745266.5136778
752103873.7096999
```

R2score of training data has decrease and it has increase on testing data

In [101]:

```
#important predictor variables
betas = pd.DataFrame(index=X_train1.columns)
betas.rows = X_train1.columns
betas['Ridge2'] = ridge2.coef_
betas['Ridge'] = ridge.coef_
betas['Lasso'] = lasso.coef_
betas['Lasso20'] = lasso20.coef_
pd.set_option('display.max_rows', None)
betas.head(68)
```

Out[101]:

	Ridge2	Ridge	Lasso	Lasso20
LotArea	52892.418502	59778.431939	63955.064210	63617.887669
OverallQual	106429.293471	115599.252408	119957.483345	121719.072148
OverallCond	30969.119664	35638.745398	37354.981812	36948.765235
YearBuilt	53872.884932	54545.692314	53864.332906	53764.548095
BsmtFinSF1	53388.964692	51586.657410	50216.539701	50458.153814
TotalBsmtSF	71811.348552	76674.754264	78348.099735	78209.333502
1stFlrSF	70196.443400	73061.086063	8832.898863	8244.958141
2ndFlrSF	33666.888170	37149.879346	0.000000	0.000000
GrLivArea	83295.309506	87839.676484	163982.920640	162804.680303
BedroomAbvGr	-38094.981167	-52962.603870	-62831.358381	-61134.170375
TotRmsAbvGrd	54102.652478	52937.952456	51280.023696	50757.774874
Street_Pave	34001.153057	49959.412426	63045.460825	59515.001052
LandSlope_Sev	-17857.132747	-27846.862924	-37188.510825	-29661.614776

	Ridge2	Ridge	Lasso	Lasso20
Condition2_PosN	-3031.699352	-11908.785655	-21920.323877	-11645.855795
RoofStyle_Shed	5474.383816	11641.731102	17801.452620	1966.058339
RoofMatl_Metal	8130.068994	18201.049929	32845.684073	16580.031007
Exterior1st_Stone	-17057.383837	-37132.047065	-69633.615929	-59674.587283
Exterior2nd_CBlock	-15569.072249	-32941.699298	-60463.906721	-49678.514531
ExterQual_Gd	-49400.503457	-54900.543840	-58459.152105	-57016.336034
ExterQual_TA	-59179.903853	-62317.508218	-64902.622534	-63508.829030
BsmtCond_Po	-4343.870481	-2488.039788	0.000000	-0.000000
KitchenQual_TA	-7060.140437	-5437.664855	-4495.491440	-4450.468043
Functional_Maj2	-10968.231950	-23574.925049	-40743.007254	-31654.783158
SaleType_CWD	-16897.367011	-27224.575631	-35460.118834	-30830.830798
SaleType_Con	13636.660731	21036.193759	25659.755739	21222.403113

- LotArea---------------Lot size in square feet
- OverallQual---------Rates the overall material and finish of the house
- OverallCond--------Rates the overall condition of the house

- YearBuilt-------------Original construction date
- BsmtFinSF1--------Type 1 finished square feet
- TotalBsmtSF------- Total square feet of basement area
- GrLivArea-----------Above grade (ground) living area square feet
- TotRmsAbvGrd----Total rooms above grade (does not include bathrooms)
- Street_Pave--------Pave road access to property
- RoofMatl_Metal----Roof material_Metal

Predictors are same but the coefficent of these predictor has changed

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 r2_score of lasso is slightly higher than lasso for the test dataset so we will choose lasso regression to solve this problem

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?

In [104]:

```
X_train1.columns
```

Out[104]:

```
Index(['LotArea', 'OverallQual', 'OverallCond', 'YearBuilt', 'BsmtFinSF1', 'T
otalBsmtSF', '1stFlrSF', '2ndFlrSF', 'GrLivArea', 'BedroomAbvGr', 'TotRmsAbvG
rd', 'Street_Pave', 'LandSlope_Sev', 'Condition2_PosN', 'RoofStyle_Shed', 'Ro
ofMatl_Metal', 'Exterior1st_Stone', 'Exterior2nd_CBlock', 'ExterQual_Gd', 'Ex
terQual_TA', 'BsmtCond_Po', 'KitchenQual_TA', 'Functional_Maj2', 'SaleType_CW
D', 'SaleType_Con'], dtype='object')
```

LotArea,OverallQual,YearBuilt,BsmtFinSF1,TotalBsmtSF are the top 5 important predictors.

Let's drop these columns

In [105]:

```
X_train2 = X_train1.drop(['LotArea','OverallQual','YearBuilt','BsmtFinSF1','TotalBsmtSF'],axis=1)
X_test2 = X_test1.drop(['LotArea','OverallQual','YearBuilt','BsmtFinSF1','TotalBsmtSF'],axis=1)
```

In [106]:

X_train2.head()

Out[106]:

	OverallCond	1stFlrSF	2ndFlrSF	GrLivArea	BedroomAbvGr	TotRmsAbvGrd	Street_Pave	LandSlope_Sev	Condition2_F
1108	0.500	0.170306	0.460583	0.407819	0.500000	0.444444	1	0	
745	1.000	0.252911	0.955928	0.753286	0.666667	0.888889	1	0	
1134	0.500	0.158661	0.424581	0.377486	0.500000	0.444444	1	0	
512	0.500	0.139738	0.000000	0.129424	0.500000	0.222222	1	0	
43	0.625	0.166667	0.000000	0.154365	0.500000	0.222222	1	0	

In [107]:

X_test2.head()

Out[107]:

	OverallCond	1stFlrSF	2ndFlrSF	GrLivArea	BedroomAbvGr	TotRmsAbvGrd	Street_Pave	LandSlope_Sev	Condition2_F
990	0.50	0.337336	0.611421	0.644422	0.5	0.444444	1	0	
1161	0.75	0.422125	0.000000	0.390967	0.5	0.444444	1	0	
1369	0.50	0.432314	0.000000	0.400404	0.5	0.555556	1	0	
329	0.50	0.042213	0.369957	0.239973	0.5	0.333333	1	0	
262	0.75	0.266376	0.000000	0.246714	0.5	0.333333	1	0	

Lasso

In [108]:

```
# alpha 10
alpha =10
lasso21 = Lasso(alpha=alpha)
lasso21.fit(X_train2, y_train)
```

Out[108]:

```
Lasso
```

```
Lasso(alpha=10)
```

In [109]:

```
# Lets calculate some metrics such as R2 score, RSS and RMSE
y_pred_train = lasso21.predict(X_train2)
y_pred_test = lasso21.predict(X_test2)

metric3 = []
r2_train_lr = r2_score(y_train, y_pred_train)
print(r2_train_lr)
metric3.append(r2_train_lr)

r2_test_lr = r2_score(y_test, y_pred_test)
print(r2_test_lr)
metric3.append(r2_test_lr)

rss1_lr = np.sum(np.square(y_train - y_pred_train))
print(rss1_lr)
metric3.append(rss1_lr)

rss2_lr = np.sum(np.square(y_test - y_pred_test))
print(rss2_lr)
metric3.append(rss2_lr)

mse_train_lr = mean_squared_error(y_train, y_pred_train)
print(mse_train_lr)
metric3.append(mse_train_lr**0.5)

mse_test_lr = mean_squared_error(y_test, y_pred_test)
print(mse_test_lr)
metric3.append(mse_test_lr**0.5)
#R2score at alpha-10
#0.8859222400899005
#0.8646660845700094
```

```
0.7988346707068132
0.758810320925813
1016954777102.8657
600167078819.8159
1138807141.2126155
1364016088.2268543
```

R2score of training and testing data has decreased

In [110]:

```
#important predictor variables
betas = pd.DataFrame(index=X_train2.columns)
betas.rows = X_train1.columns
betas['Lasso21'] = lasso21.coef_
pd.set_option('display.max_rows', None)
betas.head(68)
```

Out[110]:

	Lasso21
OverallCond	7403.774043
1stFlrSF	163379.262938
2ndFlrSF	12227.759048
GrLivArea	186638.919740
BedroomAbvGr	-71218.036474
TotRmsAbvGrd	41610.305613
Street_Pave	101376.262107
LandSlope_Sev	-40205.679947

	Lasso21
Condition2_PosN	0.000000
RoofStyle_Shed	53262.728685
RoofMatl_Metal	84219.173436
Exterior1st_Stone	-124162.644239
Exterior2nd_CBlock	-139534.253019
ExterQual_Gd	-77170.982079
ExterQual_TA	-108569.936019
BsmtCond_Po	-122646.594039
KitchenQual_TA	-11135.858324
Functional_Maj2	-48462.215856
SaleType_CWD	-64725.438438
SaleType_Con	52937.625483

five most important predictor variables

- 11stFlrSF-----------First Floor square feet
- GrLivArea-----------Above grade (ground) living area square feet
- Street_Pave---------Pave road access to property

- RoofMatl_Metal------Roof material_Metal
- RoofStyle_Shed------Type of roof(Shed)

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

The model should be generalized so that the test accuracy is not lesser than the training score. The model should be accurate for datasets other than the ones which were used during training. Too much importance should not given to the outliers so that the accuracy predicted by the model is high. To ensure that this is not the case, the outliers analysis needs to be done and only those which are relevant to the dataset need to be retained. Those outliers which it does not make sense to keep must be removed from the dataset. If the model is not robust, It cannot be trusted for predictive analysis.