Advanced Regression Assignment Part 2

Q1. 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?

Ans : The optimum value of alpha for Lasso regression was 0.001 and for Ridge regression it was 4.

For Lasso Regression:

- After doubling the alpha value the train R2 score slightly decreases from 88.21% to 87.14%. But the test score remains almost same.
- As we can see that 5 variables **Neighborhood_Crawfor, OverallCond, 1stFlrSF, KitchenQual, Condition1_PosA** remains the same with change in alpha value, however the coefficient values have undergone a change. In the remaining six variables the coefficient value has been either reduced/increased or they are replaced by some other variable.

Lasso model when alpha was chosen as 0.001

```
lasso = Lasso(alpha=.001 , random_state = 42)
lasso.fit(X_train,y_train)

y_train_pred = lasso.predict(X_train)
y_test_pred = lasso.predict(X_test)

print(r2_score(y_true=y_train,y_pred=y_train_pred))
print(r2_score(y_true=y_test,y_pred=y_test_pred))
```

```
0.8821385332052926
0.8990120902476462
```

As we can see that r2_score for train and test is coming as 88.21% and 89.90%

```
model_param = list(lasso.coef_)
model_param.insert(0,lasso.intercept_)
cols = X_train.columns
cols.insert(0,'const')
lasso_coef = pd.DataFrame(list(zip(cols,model_param)))
lasso_coef.columns = ['Feature','Coef']
```

```
lasso_coef.sort_values(by='Coef',ascending=False).head(10)
```

	Feature	Coef
45	Exterior1st_BrkComm	0.396091
23	Neighborhood_Crawfor	0.392858
40	RoofMatl_Metal	0.337284
43	RoofMatl_WdShake	0.294118
8	1stFlrSF	0.271862
1	OverallCond	0.265681
12	KitchenQual	0.264773
73	SaleCondition_Normal	0.206485
29	Condition1_Feedr	0.192175
31	Condition1_PosA	0.17648

Lasso model when alpha was chosen as 0.002

```
# Building model with alpha value 0.002
lasso = Lasso(alpha=.002 , random_state = 42)
lasso.fit(X_train,y_train)

y_train_pred = lasso.predict(X_train)
y_test_pred = lasso.predict(X_test)

print(r2_score(y_true=y_train,y_pred=y_train_pred))
print(r2_score(y_true=y_test,y_pred=y_test_pred))
```

```
0.8714283358711908
0.8952432747191325
```

```
model_param = list(lasso.coef_)
model_param.insert(0,lasso.intercept_)
cols = X_train.columns
cols.insert(0,'const')
lasso_coef = pd.DataFrame(list(zip(cols,model_param)))
lasso_coef.columns = ['Feature','Coef']
lasso_coef.sort_values(by='Coef',ascending=False).head(10)
```

	Feature	Coef
23	Neighborhood_Crawfor	0.306116
1	OverallCond	0.272208
8	1stFlrSF	0.265735
12	KitchenQual	0.260246
60	Foundation_Stone	0.158767
31	Condition1_PosA	0.1513
3	BsmtFinSF1	0.151201
68	SaleType_CWD	0.145
20	Neighborhood_Blueste	0.143648
24	Neighborhood_Edwards	0.13804

For Ridge Regression:

- After doubling the alpha value the train R2 score slightly decreases from 88.26% to 87.78%. But the test score remains almost same.
- As we can see that 5 variables **Neighborhood_Crawfor, OverallCond, 1stFlrSF, KitchenQual, Condition1_PosA** remains the same with change in alpha value, however the coefficient values have undergone a change. In the remaining six variables the coefficient value has been either reduced/increased or they are replaced by some other variable.

Ridge model results when alpha was selected as 4.

```
ridge = Ridge(alpha = 4)
ridge.fit(X_train,y_train)

y_pred_train = ridge.predict(X_train)
print(r2_score(y_train,y_pred_train))

y_pred_test = ridge.predict(X_test)
print(r2_score(y_test,y_pred_test))
```

```
0.8826968340413447
0.8987167278643621
```

As we can see the value of train and test score is 88% and 89.87%, we can say the

```
model_parameter = list(ridge.coef_)
model_parameter.insert(0,ridge.intercept_)
cols = X_train.columns
cols.insert(0,'const')
ridge_coef = pd.DataFrame(list(zip(cols,model_param)))
ridge_coef.columns = ['Feature','Coef']
```

```
ridge_coef.sort_values(by='Coef',ascending=False).head(10)
```

	Feature	Coef
45	Exterior1st_BrkComm	0.396091
23	Neighborhood_Crawfor	0.392858
40	RoofMatl_Metal	0.337284
43	RoofMatl_WdShake	0.294118
8	1stFlrSF	0.271862
1	OverallCond	0.265681
12	KitchenQual	0.264773
73	SaleCondition_Normal	0.206485
29	Condition1_Feedr	0.192175
31	Condition1_PosA	0.17648

Ridge model when alpha was doubled to 8

```
ridge = Ridge(alpha = 8)
ridge.fit(X_train,y_train)

y_pred_train = ridge.predict(X_train)
print(r2_score(y_train,y_pred_train))

y_pred_test = ridge.predict(X_test)
print(r2_score(y_test,y_pred_test))
```

```
0.87773861474159
0.8984059728100188
```

```
model_parameter = list(ridge.coef_)
model_parameter.insert(0,ridge.intercept_)
cols = X_train.columns
cols.insert(0,'const')
ridge_coef = pd.DataFrame(list(zip(cols,model_param)))
ridge_coef.columns = ['Feature','Coef']
ridge_coef.sort_values(by='Coef',ascending=False).head(10)
```

	Feature	Coef
23	Neighborhood_Crawfor	0.306116
1	OverallCond	0.272208
8	1stFlrSF	0.265735
12	KitchenQual	0.260246
60	Foundation_Stone	0.158767
31	Condition1_PosA	0.1513
3	BsmtFinSF1	0.151201
68	SaleType_CWD	0.145
20	Neighborhood_Blueste	0.143648
24	Neighborhood_Edwards	0.13804

Most Important Predictors after changes in alpha values were implemented are:

- **Exterior1st_BrkComm**
- **Neighborhood_Crawfor**
- **RoofMatl_Metal**
- **RoofMatl_WdShake**
- **1stFlrSF**
- **OverallCond**
- **KitchenQual**
- **SaleCondition_Normal**
- **Condition1_Feedr**
- **Condition1_PosA**

Q2. 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?

Ans: The optimal value of lambda (alpha) for ridge and lasso is selected based on below parameters:

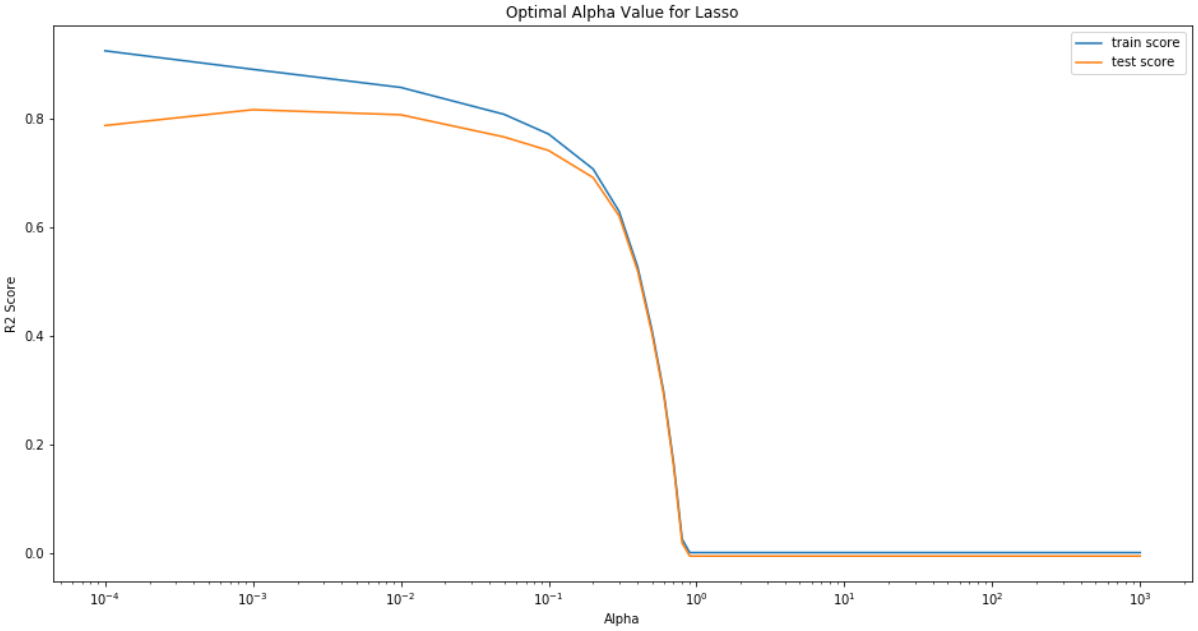
1. High value of r2 score for test and train set.
2. And Balance between the r2 score of both train and test set.

Thus, we ensured that model doesn't over fit and is a more robust and generalizable model.

So to find the optimal value of alpha it is required to build models with different alpha values and then checking mean train score and mean test score for different alpha values. To perform this GridSearchCV with 5

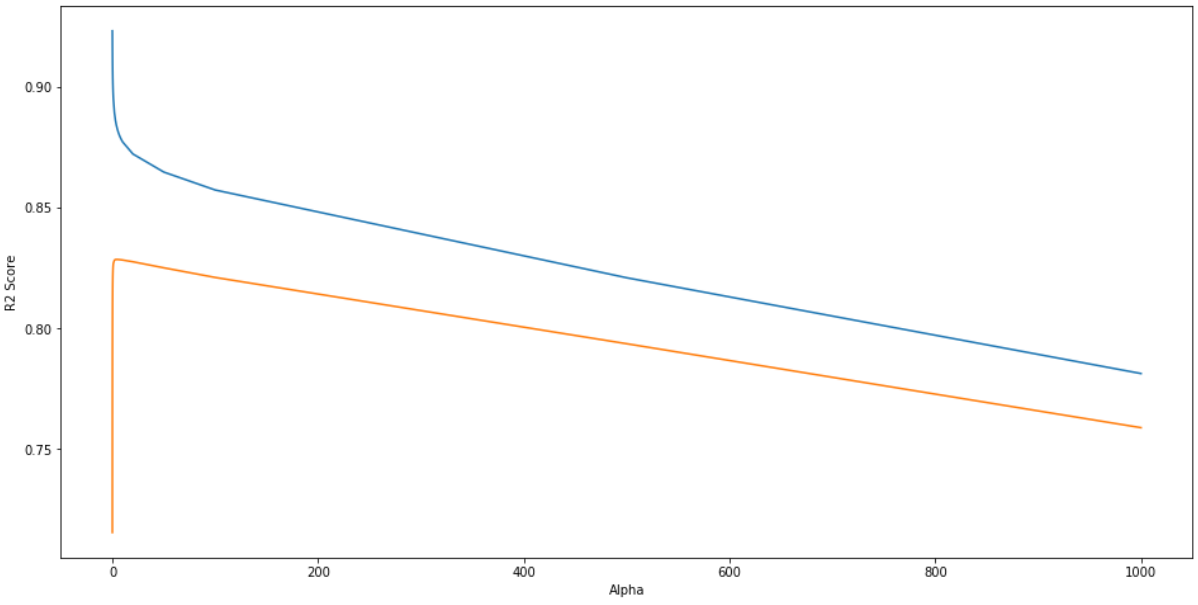
fold cross validation was used. The result of the GridSearchCV was plotted with Alpha on X-axis and Mean Train and Test Score on Y axis. Based on the graph the optimal value was selected.

Lasso Regression:



Based on the above plot (and model_cv.best_params) it was concluded that at optimal value of alpha is 0.001 as both test and train score are optimal at this point.

Ridge Regression:



Based on the above plot (and model_cv.best_params) it was concluded that at optimal value of alpha is 4.0 as both test and train score are optimal at this point.

In lasso when alpha value was selected as 0.001 r2 score for train and test was 88% and 89.87% respectively whereas when alpha was selected as 0.0001 r2 score for train and test was 92.11% and 81.51% respectively. So

it is important that a value of alpha is chosen in such a way that accuracy for test and train set is almost same. This makes model more reliable and robust.

Q3. After building the model, you realized 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?

Ans: Top 5 predictors we got after Ridge and Lasso model building were:

- Exterior1st_BrkComm
- Neighborhood_Crawfor
- RoofMatl_Metal
- RoofMatl_WdShake
- 1stFlrSF

So for new model we needed to drop above mentioned columns, but assuming that all the columns that are derived after dummy variable creation for such columns we need to drop parent attributes as well.

So all the columns selected for deletions are
'Neighborhood_Crawfor','OverallCond','1stFlrSF','Exterior1st_BrkComm','Exterior1st_CBlock', 'Exterior1st_HdBoard', 'Exterior1st_Plywood', 'Exterior1st_Stone', 'Exterior1st_Wd Sdng', 'Exterior1st_WdShing','Neighborhood_Blueste','Neighborhood_BrDale','Neighborhood_ClearCr','Neighborhood_Edwards','Neighborhood_IDOTRR', 'Neighborhood_MeadowV', 'Neighborhood_StoneBr','Neighborhood_Veenker', 'RoofMatl_CompShg', 'RoofMatl_Metal', 'RoofMatl_Roll', 'RoofMatl_Tar&Grv', 'RoofMatl_WdShake', 'RoofMatl_WdShngl'

After model building it is observed that the best alpha value for Ridge is 20.

```
ridge = Ridge(alpha = 20)
ridge.fit(df_top_train,y_train)

y_pred_train = ridge.predict(df_top_train)
print(r2_score(y_train,y_pred_train))

y_pred_test = ridge.predict(df_top_test)
print(r2_score(y_test,y_pred_test))
```

```
0.852324948616984
0.8778746555436929
```

```
_parameter = list(ridge.coef_)
model_parameter.insert(0,ridge.intercept_)
cols = df_top_train.columns
cols.insert(0,'const')
ridge_coef = pd.DataFrame(list(zip(cols,model_param)))
ridge_coef.columns = ['Feature','Coef']
ridge_coef.sort_values(by='Coef',ascending=False).head(10)
```

	Feature	Coef
23	Condition1_RRNe	0.306116
1	BsmtQual	0.272208
8	LowQualFinSF	0.265735
12	GarageArea	0.260246
31	Exterior2nd_Plywood	0.1513
3	BsmtFinSF2	0.151201
20	Condition1_PosA	0.143648
24	Condition1_RRNn	0.13804
15	MSZoning_RL	0.128985
13	MSZoning_FV	0.098941

New R2 Score for Lasso is 85.23% and 87.78% for Train and Test Set.

After model building it is observed that the best alpha value for Ridge is 0.001.

```
# Building model with alpha value 0.01
lasso = Lasso(alpha=.001 , random_state = 42)
lasso.fit(df_top_train,y_train)

y_train_pred = lasso.predict(df_top_train)
y_test_pred = lasso.predict(df_top_test)

print(r2_score(y_true=y_train,y_pred=y_train_pred))
print(r2_score(y_true=y_test,y_pred=y_test_pred))
```

```
0.8598063272297403
0.8805687571235958
```

```
model_param = list(lasso.coef_)
model_param.insert(0,lasso.intercept_)
cols = df_top_train.columns
cols.insert(0,'const')
lasso_coef = pd.DataFrame(list(zip(cols,model_param)))
lasso_coef.columns = ['Feature','Coef']
lasso_coef.sort_values(by='Coef',ascending=False).head(10)
```

	Feature	Coef
7	2ndFlrSF	0.353428
1	BsmtQual	0.301076
10	KitchenQual	0.30044
49	SaleCondition_Normal	0.196862
50	SaleCondition_Partial	0.182081
18	Condition1_Feedr	0.179484
20	Condition1_PosA	0.174314
44	SaleType_CWD	0.17383
16	MSZoning_RM	0.139845
11	Fireplaces	0.134861

New R2 Score for Lasso is 85.98% and 88.05% for Train and Test Set.

Top 5 predictors after building new model after deleting previously selected top 5 columns are :

- 2ndFlrSF
- BsmtQual
- KitchenQual
- SaleCondition_Normal
- SaleCondition_Partial

Q4. How can you make sure that a model is robust and generalizable? What are the implications of the same for the accuracy of the model and why?

Ans: A **model** is considered to be **robust** if its output dependent variable (label) is consistently accurate even if one or more of the input independent variables (features) or assumptions are drastically changed due to unforeseen circumstances.

Generalization refers to your **model's** ability to adapt properly to new, previously unseen data, drawn from the same distribution as the one used to create the **model**.

So to make a model more robust and Generalizable, we should follow various techniques like:

a. **Use different metrics** : We have used Negative Mean Absolute error and R2 Scoring to see which metrics is helping in better prediction.
b. **Transform data**. If data has a very pronounced right tail, we can try different transformation, like in this assignment we made use of power transformation.
c. **Remove the outliers**. This works if there are very few of them and we are fairly certain they're anomalies and not worth predicting. But in current assignment we didn't treated outliers since removing them or capping them was fetching better result but still we were compromising on critical information so for this assignment we choose not to treat outliers
d. **Try other models :** We have used both Ridge and Lasso to predict final model
e. **Feature Selection:** We used RFE and Lasso
f. **Treating Missing values**
g. **Cross Validation: To select the best value of alpha we used GridSearchCV**
h. **Develop intuition about overfitting** : With certain value selection we were getting high value of training accuracy but test accuracy was low, so this show overfitting and hence we didn't selected those values for our final model building.

The implication of robust and generalizable model is that it **gives similar accuracy results both for train as well as test data sets. In other words there is not much gap in training and test accuracy results produced by the model.**