Advanced Regression Assignment Problem Statement - Part II (House Price Prediction)

Name- Swaraj Parida

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?

Ans.

We got the following values

- For Ridge Regression, optimal value of alpha = **3.0**
- For Lasso Regression, optimal value of alpha = **0.0001**

If we double the value of alpha for alpha for ridge i.e. alpha = 6**,**

```
# If we double the value of alpha for ridge regression, i.e. alpha = 6
ridge = Ridge(alpha=6)
ridge.fit(X_train, y_train)
```

```
Ridge
Ridge(alpha=6)
```

```
# predict
y_pred_train = ridge.predict(X_train)
y_pred_test = ridge.predict(X_test)

# Model evaluation
r2_train_rr = r2_score(y_train, y_pred_train)
print("R2 score for train (Ridge Regression): ",r2_train_rr)

r2_test_rr = r2_score(y_test, y_pred_test)
print("R2 score for test (Ridge Regression): ",r2_test_rr)
```

```
R2 score for train (Ridge Regression):  0.9397199274145321
R2 score for test (Ridge Regression):  0.9022386007847552
```

As we can see we found R2 score of **0.9397199274145321** and **0.9022386007847552** for train and test data in ridge regression.

Similarly, if we double the value of alpha for alpha for lasso i.e. alpha = 0.0002**,**

```
# If we double the value of alpha for lasso regression, i.e. alpha = 0.0002
lasso = Lasso(alpha=0.0002)
lasso.fit(X_train, y_train)
```

```
Lasso(alpha=0.0002)
```

```
# predict
y_pred_train = lasso.predict(X_train)
y_pred_test = lasso.predict(X_test)

# Model evaluation
r2_train_lr = r2_score(y_train, y_pred_train)
print("R2 score for train (Lasso Regression): ",r2_train_lr)

r2_test_lr = r2_score(y_test, y_pred_test)
print("R2 score for test (Lasso Regression): ",r2_test_lr)
```

```
R2 score for train (Lasso Regression):  0.9371254870259446
R2 score for test (Lasso Regression):  0.9075438768209891
```

In lasso, got R2 score of **0.9371254870259446** and **0.9075438768209891** for train and test data in ridge regression.

After the implementation, we got the following features to be the most important,

GrLivArea, TotalBsmtSF, OverallQual, Neighborhood_StoneBr, BsmtFinSF1, GarageArea, OverallCond, Neighborhood_NoRidge, ExterQual_TA and Functional_Typ

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?

Ans.

In ridge regression, using optimal value of alpha, R2 score for train = 0.944495 and R2 score for test = 0.903457

In lasso regression, using optimal value of alpha, R2 score for train = 0.942484 and R2 score for test = 0.906751

Now, as the R2 score is slightly better in case Lasso than ridge, we will choose lasso regression.

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?

Ans.

Following are the top 5 important features before dropping:

1. GrLivArea
2. TotalBsmtSF
3. OverallQual
4. Neighborhood_StoneBr
5. OverallCond

Let's drop the mentioned feature and perform lasso regression.

```
X_train_2nd = X_train.drop(['GrLivArea', 'TotalBsmtSF', 'OverallQual', 'Neighborhood_StoneBr', 'OverallCond'], axis=1)
X_test_2nd = X_test.drop(['GrLivArea', 'TotalBsmtSF', 'OverallQual', 'Neighborhood_StoneBr', 'OverallCond'], axis=1)
```

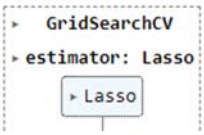
```
# Lasso Regression
```

```
# list of alphas to tune
params = {'alpha':[0.0001, 0.001, 0.01, 0.05, 0.1,
 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 2.0, 3.0,
 4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 10.0, 20, 50, 100, 500, 1000]}

lasso = Lasso()

# cross validation
folds = 5
model_cv_lasso_2nd = GridSearchCV(estimator=lasso,
                        param_grid=params,
                        scoring='neg_mean_absolute_error',
                        cv=folds,
                        return_train_score=True,
                        verbose=1)
# fit
model_cv_lasso_2nd.fit(X_train_2nd, y_train)
```

```
Fitting 5 folds for each of 28 candidates, totalling 140 fits
```




```
# top 5 features after dropping
betas_lasso_2nd = pd.DataFrame(index=X_train_2nd.columns)
betas_lasso_2nd.row = X_train_2nd.columns
betas_lasso_2nd['Coefficient'] = lasso.coef_
betas_lasso_2nd['Absolute Coefficient'] = abs(lasso.coef_)
betas_lasso_2nd.sort_values(by=['Absolute Coefficient'], ascending=False).head()
```

	Coefficient	Absolute Coefficient
1stFlrSF	0.291920	0.291920
2ndFlrSF	0.141123	0.141123
BsmtFinSF1	0.130968	0.130968
BsmtUnfSF	0.081755	0.081755
GarageArea	0.055340	0.055340

Now, followings are the top 5 important features we obtained after dropping:

1. 1stFlrSF
2. 2ndFlrSF
3. BsmtFinSF1
4. BsmtUnfSF
5. GarageArea

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?

Ans.

As we know a model is considered robust and generalisable when it equally performs well on both the training and test data. In order to make sure this, we can perform following procedures:

1. We must first treat missing value by impute or replace them with appropriate value.
2. Then, we can check for the outliers in the data
3. We must perform scaling for our independent variable and target variable.
4. Then. We must go for cross validation to improve the performance on unseen data.

If we talk about the implication, in case of making the simple we have to Bias-Variance trade-off as we have to sacrifice the Bias in order to get less variance