## 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 to double the value of alpha for both ridge and lasso? What will be the most important predictor variables after the change is implemented?

The optimum alpha values of Ridge and Lasso regression are 0.8 and 0.0001. When the alpha values are doubled, the Lasso and Ridge prediction score became almost the same with the R2 value of 88%.

```
1
2
3   ridge = Ridge(alpha=1.6) # 0.8 * 2 = 1.6
4   ridge.fit(X_train_rfe, y_train_scaled)
5
6   y_train_pred = ridge.predict(X_train_rfe)
7   y_test_pred = ridge.predict(X_test)
8
9   # Scores
10
11  print('R2 score (train) : ',r2_score(y_train_scaled,y_train_pred))
12  print('R2 score (test) : ',r2_score(y_test_scaled,y_test_pred))
13  print('RMSE (train) : ', np.sqrt(mean_squared_error(y_train_scaled, y_train_pred)))
14  print('RMSE (test) : ', np.sqrt(mean_squared_error(y_test_scaled, y_test_pred)))
15
16
```

▸ (1) MLflow run

```
R2 score (train) :  0.899542788569235
R2 score (test) :  0.8819672914311996
RMSE (train) :  0.04135640698456591
RMSE (test) :  0.04397073096310872
```

```python


lasso = Lasso(alpha=0.0002) # 0.0001 * 2 = 0.0002
lasso.fit(X_train_rfe, y_train_scaled)

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

# Scores

print('R2 score (train) : ',r2_score(y_train_scaled,y_train_pred))
print('R2 score (test) : ',r2_score(y_test_scaled,y_test_pred))
print('RMSE (train) : ', np.sqrt(mean_squared_error(y_train_scaled, y_train_pred)))
print('RMSE (test) : ', np.sqrt(mean_squared_error(y_test_scaled, y_test_pred)))


```

▾ (1) MLflow run

Logged 1 run to an experiment in MLflow. Learn more

```
R2 score (train) :  0.8927921224213486
R2 score (test) :  0.888650110484395
RMSE (train) :  0.04272337866482642
RMSE (test) :  0.042707819044503624
```

The most important predictors for Lasso are below,

```python
model_coef_1[model_coef_1['Lasso (alpha=0.0002)']!=0][['Lasso (alpha=0.0002)']].sort_values(by='Lasso (alpha=0.0002)', ascending=False)
```

| | Lasso (alpha=0.0002) |
|---|---|
| GrLivArea | 0.224467 |
| OverallQual | 0.187642 |
| TotalBsmtSF | 0.135337 |
| OverallCond | 0.077752 |
| GarageCars | 0.069602 |
| LotArea | 0.049616 |
| KitchenQual | 0.045449 |
| MSZoning_FV | 0.033596 |
| MSZoning_RL | 0.021558 |
| SaleType_New | 0.014374 |
| MSZoning_RH | 0.003176 |
| GarageQual | 0.001920 |
| BsmtUnfSF | -0.047578 |
| Functional | -0.052571 |
| age | -0.104610 |

The most important predictors for Ridge are below,

```
model_coef_1[['Ridge (alpha=1.6)']].sort_values(by='Ridge (alpha=1.6)', ascending=False)
```

| | Ridge (alpha=1.6) |
|---|---|
| GrLivArea | 0.221255 |
| OverallQual | 0.174458 |
| TotalBsmtSF | 0.135842 |
| MSZoning_FV | 0.090149 |
| MSZoning_RL | 0.075796 |
| OverallCond | 0.074215 |
| GarageCars | 0.073071 |
| MSZoning_RH | 0.068992 |
| MSZoning_RM | 0.056515 |
| LotArea | 0.056401 |
| KitchenQual | 0.051462 |
| SaleType_New | 0.039013 |
| GarageQual | 0.035141 |
| SaleType_Con | 0.029339 |
| Heating_Grav | 0.026478 |
| GarageFinish_NOT APPLICABLE | 0.017955 |
| Heating_OthW | -0.021035 |
| SaleCondition_Partial | -0.022564 |
| RoofStyle_Shed | -0.029762 |
| Foundation_Wood | -0.032239 |
| SaleType_CWD | -0.041571 |
| Exterior1st_BrkComm | -0.044418 |
| BsmtUnfSF | -0.049158 |
| Functional | -0.063487 |
| age | -0.099918 |

## 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?

I will choose Lasso for the bellow reasons,

1. Lasso Performed better than Ridge and Linear Regression
2. Lasso prediction score on train(0.897) and test(0.890) is close to zero variance(0.007)
3. Ridge prediction score on train(0.901) and test(0.876) is close to 3% variance

4. RMSE value of Prediction on test is slightly lesser for lasso compared to Ridge, smaller the better. So Lasso wins even with RMSE
5. In Lasso, some of these coefficients became 0, thus resulting in an easier interpretation with predictors

## 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?

The 5 most important predictor variables excluding the initial 5 predictors by the original lasso model are below,

```
model_coef_2 = pd.DataFrame(index=X_train_rfe.columns)
model_coef_2.rows = X_train_rfe.columns
model_coef_2['Lasso (alpha=0.001)'] = final_model_lasso.coef_
model_coef_2[model_coef_2['Lasso (alpha=0.001)']!=0][['Lasso (alpha=0.001)']].sort_values(by='Lasso (alpha=0.001)', ascending=False)
```

| | Lasso (alpha=0.001) |
|---|---|
| GarageCars | 0.177129 |
| KitchenQual | 0.175559 |
| LotArea | 0.146529 |
| OverallCond | 0.051638 |
| SaleType_New | 0.012155 |

Here the dropList contains the initial 5 most important predictors, the below approach is followed to calculate the next 5 most important predictors

```
dropList=['GrLivArea','OverallQual','TotalBsmtSF','MSZoning_FV','MSZoning_RL']
```

Command took 0.09 seconds -- by pradeep.ravi@databricks.com at 1/20/2024, 11:44:20 PM on Pradeep Ravi's Cluster

Cmd 134

```
X_train_rfe=X_train_rfe.drop(dropList,axis=1)
X_test=X_test.drop(dropList,axis=1)
estimator_model = Lasso()
cv = GridSearchCV(estimator = estimator_model,
                  param_grid = params,
                  scoring= 'neg_mean_absolute_error',
                  cv = 5,
                  return_train_score=True,
                  verbose = 1)
cv.fit(X_train_rfe, y_train_scaled)
alpha = cv.best_params_["alpha"]
print(f"Lasso Optimum alpha valaue is {alpha}")
final_model_lasso = cv.best_estimator_

final_model_lasso.fit(X_train_rfe, y_train_scaled)
y_train_pred = final_model_lasso.predict(X_train_rfe)
y_test_pred = final_model_lasso.predict(X_test)

# Scores

print('R2 score (train) : ',r2_score(y_train_scaled,y_train_pred))
print('R2 score (test) : ',r2_score(y_test_scaled,y_test_pred))
print('RMSE (train) : ', np.sqrt(mean_squared_error(y_train_scaled, y_train_pred)))
print('RMSE (test) : ', np.sqrt(mean_squared_error(y_test_scaled, y_test_pred)))


```

(7) MLflow runs

Logged 7 runs to an experiment in MLflow. Learn more

```
Fitting 5 folds for each of 27 candidates, totalling 135 fits
Lasso Optimum alpha valaue is 0.001
R2 score (train) :  0.7056979294448806
R2 score (test) :  0.702067181477535
```

## 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?

- The model should adhere to the optimal feature selection w.r.t  Bias/Variance Trade-off, we should not have more features and have a complex model or fewer features with more Bias. The optimal number of features will help us to build a robust and generalizable model
- The model has high accuracy in training but in tests, the scores are very low means the model memorised the pattern in the train data and resulted in overfitting. So the train/test prediction accuracy should not have a high variance
- Techniques like Regularization using Lasso and Ridge help too avoid overfitting