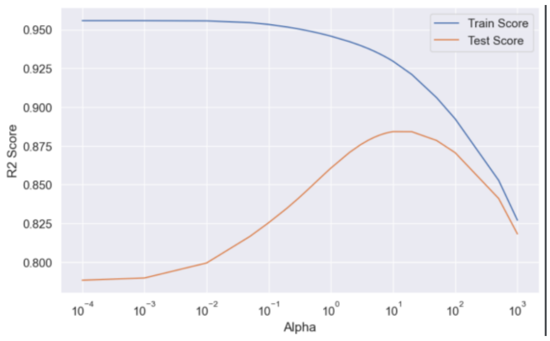
Advances Regression – Subjective Questions

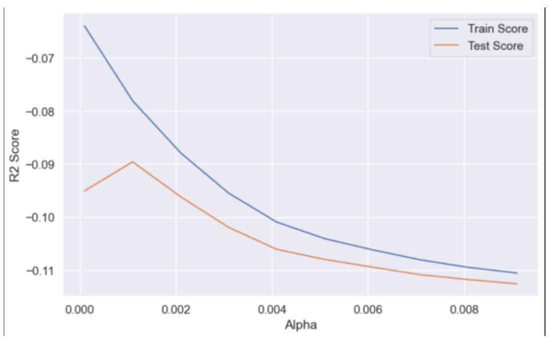
Question – 1

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?

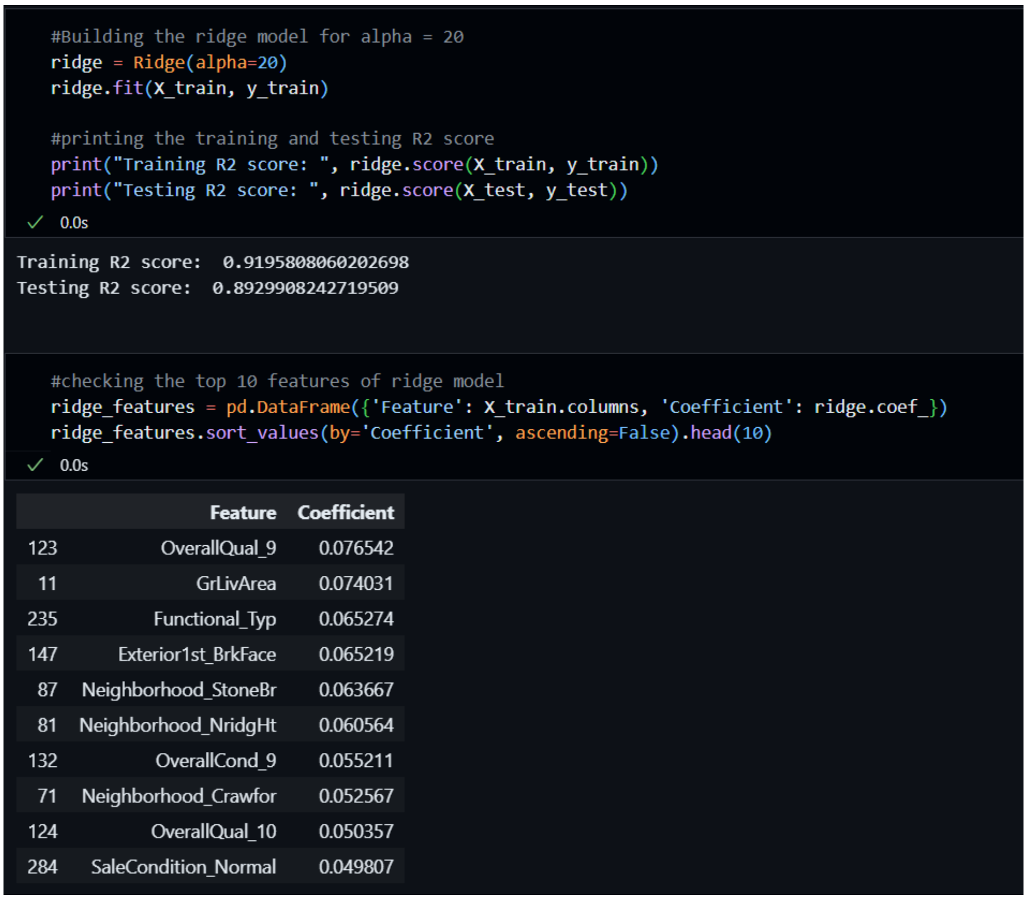
Answer:

The optimal value of alpha for Ridge regression is 10 and Lasso regression is 0.0011. If we double the value of alpha for both ridge and lasso regression, the model will start to underfit and reducing the accuracy of the model. In case of Lasso regression, as the alpha value increased , more coefficients will be forced to zero, leading to underfitting model. The same can be observed on the below plots.





Building the Ridge model with Alpha = 20 and checking the important predictor variables



```
#Building the ridge model for alpha = 20
ridge = Ridge(alpha=20)
ridge.fit(X_train, y_train)

#printing the training and testing R2 score
print("Training R2 score: ", ridge.score(X_train, y_train))
print("Testing R2 score: ", ridge.score(X_test, y_test))
✓ 0.0s
```

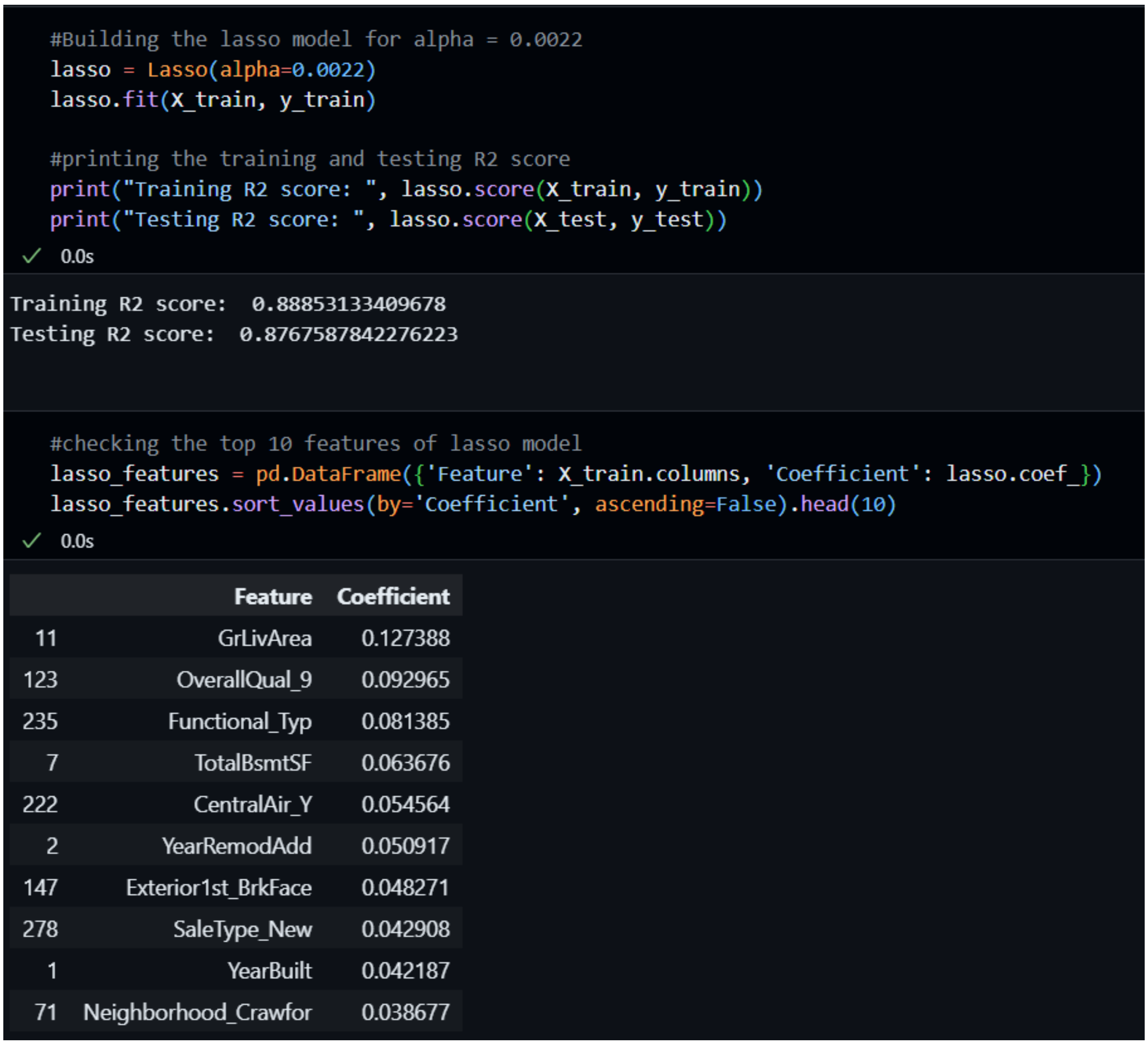
```
Training R2 score:  0.9195808060202698
Testing R2 score:  0.8929908242719509
```

```
#checking the top 10 features of ridge model
ridge_features = pd.DataFrame({'Feature': X_train.columns, 'Coefficient': ridge.coef_})
ridge_features.sort_values(by='Coefficient', ascending=False).head(10)
✓ 0.0s
```

	Feature	Coefficient
123	OverallQual_9	0.076542
11	GrLivArea	0.074031
235	Functional_Typ	0.065274
147	Exterior1st_BrkFace	0.065219
87	Neighborhood_StoneBr	0.063667
81	Neighborhood_NridgHt	0.060564
132	OverallCond_9	0.055211
71	Neighborhood_Crawfor	0.052567
124	OverallQual_10	0.050357
284	SaleCondition_Normal	0.049807

The above output is listed with the Most to least important features after changing the alpha value to 20 for ridge regression. The R^2 score of train set decreased from 0.9266 to 0. 919 and test set decreased from 0.894 to 0.892 for ridge regression.

Building the Lasso regression model with alpha = 0.0022



```
#Building the lasso model for alpha = 0.0022
lasso = Lasso(alpha=0.0022)
lasso.fit(X_train, y_train)

#printing the training and testing R2 score
print("Training R2 score: ", lasso.score(X_train, y_train))
print("Testing R2 score: ", lasso.score(X_test, y_test))
```

✓ 0.0s

```
Training R2 score:  0.88853133409678
Testing R2 score:  0.8767587842276223
```

```
#checking the top 10 features of lasso model
lasso_features = pd.DataFrame({'Feature': X_train.columns, 'Coefficient': lasso.coef_})
lasso_features.sort_values(by='Coefficient', ascending=False).head(10)
```

✓ 0.0s

	Feature	Coefficient
11	GrLivArea	0.127388
123	OverallQual_9	0.092965
235	Functional_Typ	0.081385
7	TotalBsmtSF	0.063676
222	CentralAir_Y	0.054564
2	YearRemodAdd	0.050917
147	Exterior1st_BrkFace	0.048271
278	SaleType_New	0.042908
1	YearBuilt	0.042187
71	Neighborhood_Crawfor	0.038677

The above listed features are the important features after changing the alpha value to 0.0022 for Lasso regression. The R^2 score of train set decreased from 0.909 to 0.888 and test set decreased from 0.893 to 0.876 for ridge regression

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:

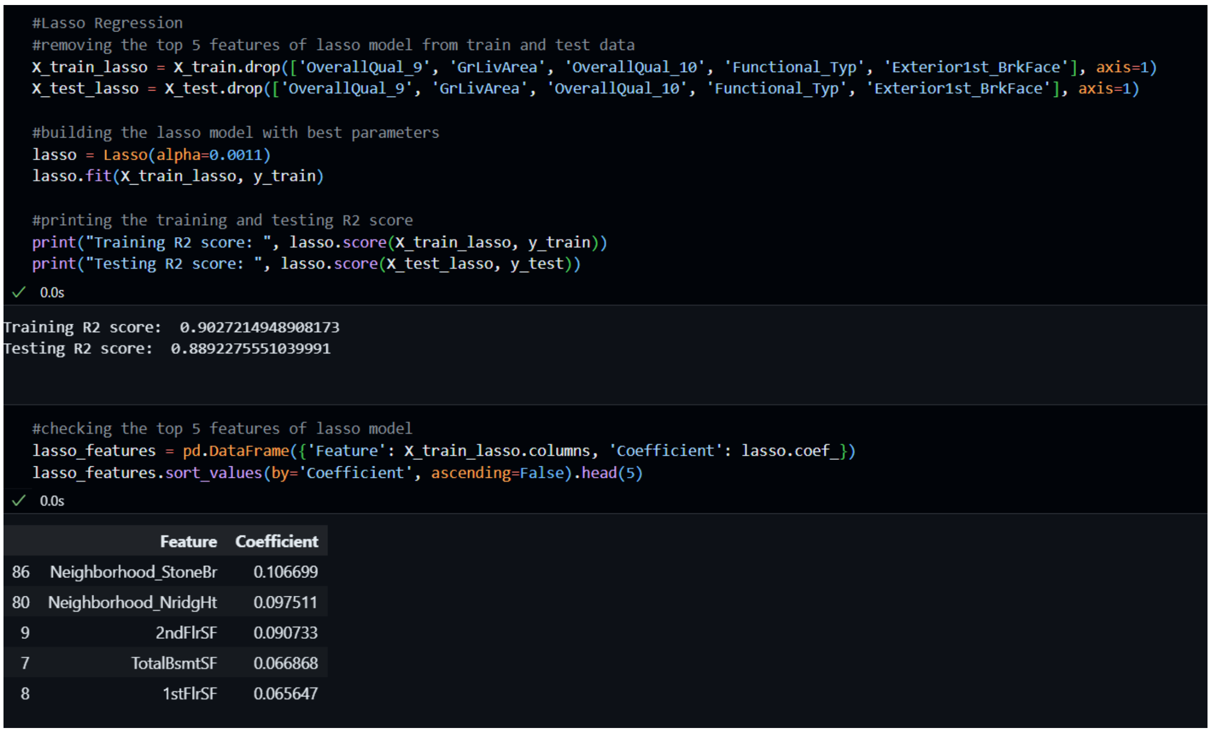
Both the models have nearly the similar R2 score for test and training set. Hence choosing the model depends on the use case. We can use Lasso regression if we have large number of independent variables and we need to perform feature selection as Lasso regression will force the coefficients to zero, thus limiting the number of features. We can select ridge regression if we want to have all the features while building the model.

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:

Lasso Regression



```
#Lasso Regression
#removing the top 5 features of lasso model from train and test data
X_train_lasso = X_train.drop(['OverallQual_9', 'GrLivArea', 'OverallQual_10', 'Functional_Typ', 'Exterior1st_BrkFace'], axis=1)
X_test_lasso = X_test.drop(['OverallQual_9', 'GrLivArea', 'OverallQual_10', 'Functional_Typ', 'Exterior1st_BrkFace'], axis=1)

#building the lasso model with best parameters
lasso = Lasso(alpha=0.0011)
lasso.fit(X_train_lasso, y_train)

#printing the training and testing R2 score
print("Training R2 score: ", lasso.score(X_train_lasso, y_train))
print("Testing R2 score: ", lasso.score(X_test_lasso, y_test))
```

✓ 0.0s

```
Training R2 score:  0.9027214948908173
Testing R2 score:  0.8892275551039991
```

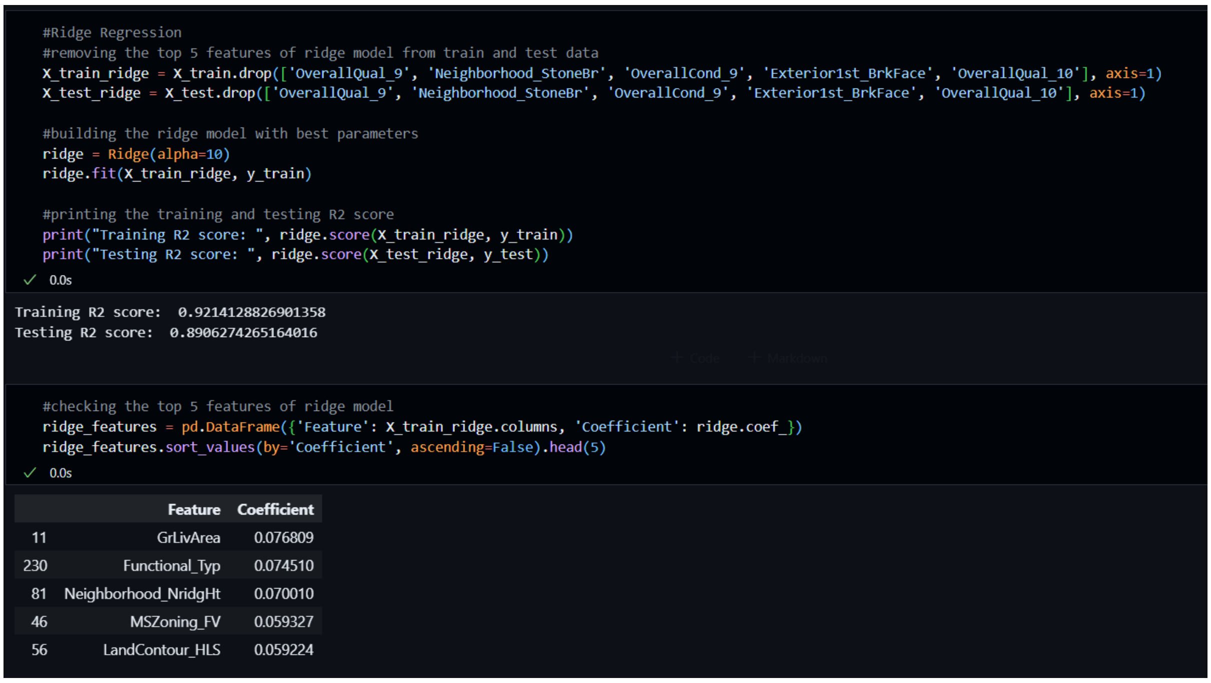
```
#checking the top 5 features of lasso model
lasso_features = pd.DataFrame({'Feature': X_train_lasso.columns, 'Coefficient': lasso.coef_})
lasso_features.sort_values(by='Coefficient', ascending=False).head(5)
```

✓ 0.0s

	Feature	Coefficient
86	Neighborhood_StoneBr	0.106699
80	Neighborhood_NridgHt	0.097511
9	2ndFlrSF	0.090733
7	TotalBsmtSF	0.066868
8	1stFlrSF	0.065647

The above are the top 5 features are removing the top 5 features for previous model for Lasso Regression.

Ridge Regression



```
#Ridge Regression
#removing the top 5 features of ridge model from train and test data
X_train_ridge = X_train.drop(['OverallQual_9', 'Neighborhood_StoneBr', 'OverallCond_9', 'Exterior1st_BrkFace', 'OverallQual_10'], axis=1)
X_test_ridge = X_test.drop(['OverallQual_9', 'Neighborhood_StoneBr', 'OverallCond_9', 'Exterior1st_BrkFace', 'OverallQual_10'], axis=1)

#building the ridge model with best parameters
ridge = Ridge(alpha=10)
ridge.fit(X_train_ridge, y_train)

#printing the training and testing R2 score
print("Training R2 score: ", ridge.score(X_train_ridge, y_train))
print("Testing R2 score: ", ridge.score(X_test_ridge, y_test))
```

✓ 0.0s

```
Training R2 score:  0.9214128826901358
Testing R2 score:  0.8906274265164016
```

```
#checking the top 5 features of ridge model
ridge_features = pd.DataFrame({'Feature': X_train_ridge.columns, 'Coefficient': ridge.coef_})
ridge_features.sort_values(by='Coefficient', ascending=False).head(5)
```

✓ 0.0s

	Feature	Coefficient
11	GrLivArea	0.076809
230	Functional_Typ	0.074510
81	Neighborhood_NridgHt	0.070010
46	MSZoning_FV	0.059327
56	LandContour_HLS	0.059224

The above are the top 5 features are removing the top 5 features for previous model for Ridge Regression.

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 is said to be robust when any variation in the data doesn't affect the model's performance. A generalisable model tends to perform well and make accurate predictions on new and unseen data beyond the training set. It indicates the model's capacity to capture and understand the underlying patterns and relationships in the data, allowing it to make reliable predictions in various contexts.

We can make the model robust by training the model on large number of data having all the variations which will help the model to have a better understanding of underlying patterns. We can also use the regularization technique to add penalty to the model if it tends to become more complex and promotes simpler model. We can cross validation to identify the optimal value of alpha to have the better accuracy for the model. In addition to this, hyperparameter tuning and feature selection to increase the performance of the model.

Robust and generalisable model doesn't always tend to have the maximum accuracy, as some accuracy is sacrificed for the model to be generalised and prevent overfitting. The model which is too complex can have high accuracy on training data but it will not be able to perform well on the unseen data leading to overfitting. The best practice is to strike the balance between having a reliable accuracy and model being robust and generalisable, which is considered ideal for real life applications.