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?

A: The optimal value of Alpha for Ridge regression is 10.0 and the optimal value of Alpha for lasso regression is 0.001. The value of R2 score almost remains constant reducing by 0.01 for ridge expression and increasing by 0.02 for lass regression. Below is the screenshot of the statistics calculation

Out[2381]:

	Metric	Ridge Regression	Ridge Regression (2X Alpha)	Lasso Regression	Lasso Regression (2X Alpha)
0	R2 Score (Train)	0.954961	0.947234	0.929503	0.947234
1	R2 Score (Test)	0.898857	0.897932	0.893315	0.897932
2	RSS (Train)	5.878695	6.887333	9.201675	6.887333
3	RSS (Test)	5.152112	5.199204	5.434409	5.199204
4	RMSE (Train)	0.075880	0.082132	0.094934	0.082132
5	RMSE (Test)	0.108457	0.108951	0.111388	0.108951

Below is the screenshot for the change in coefficients and the most important predictor variables:

Top ten coefficients of Ridge regression

```
In [2375]: betas['Ridge'].sort_values(ascending=False)[:10]
Out[2375]: OverallQual_9          0.087202
           Neighborhood_Crawfor   0.078424
           OverallQual_8          0.067638
           OverallCond_9          0.062126
           Functional_Typ         0.056263
           Exterior1st_BrkFace    0.051642
           Neighborhood_StoneBr   0.050644
           GarageYrBlt_2.0        0.049803
           BsmtCond_Gd            0.045975
           OverallCond_8          0.045816
           Name: Ridge, dtype: float64
```

Top 10 Ridge regression coefficients with 2x Alpha

```
In [2385]: betas_2['Ridge2'].sort_values(ascending=False)[:10]
Out[2385]: OverallQual_9          0.066585
           Neighborhood_Crawfor   0.063865
           OverallQual_8          0.059824
           Functional_Typ         0.050927
           OverallCond_7          0.043785
           OverallCond_9          0.043653
           Exterior1st_BrkFace    0.042566
           CentralAir_Y           0.041252
           Condition1_Norm        0.040695
           OverallCond_8          0.039620
           Name: Ridge2, dtype: float64
```

Top ten coefficients of Lasso regression

```
In [2376]: betas['Lasso'].sort_values(ascending=False)[:10]
Out[2376]: OverallQual_9          0.136583
           OverallQual_8          0.104706
           KitchenAbvGr_1         0.092110
           Neighborhood_Crawfor   0.090040
           Functional_Typ         0.061357
           OverallQual_7          0.060040
           CentralAir_Y           0.056949
           Neighborhood_Somerst   0.051116
           Condition1_Norm        0.046598
           OverallCond_7          0.044366
           Name: Lasso, dtype: float64
```

Top ten lasso regression coeffients with 2x Alpha

```
In [2386]: betas_2['Lasso2'].sort_values(ascending=False)[:10]
Out[2386]: OverallQual_9          0.085897
           OverallQual_8          0.078406
           KitchenAbvGr_1         0.073825
           Neighborhood_Crawfor   0.068698
           CentralAir_Y           0.064850
           Functional_Typ         0.055678
           OverallQual_7          0.050408
           Condition1_Norm        0.042821
           OverallCond_7          0.042014
           Foundation_PConc       0.038271
           Name: Lasso2, dtype: float64
```

Screenshot for jupyter notebook solution for first question:

CHANGES FOR SUBJECTIVE QUESTIONS - Q1 [please refer pdf]

In [2377]:
```
## Ridge regression model with double value of alpha = 2*10
ridge2 = Ridge(alpha=20)

# Fit the model on training data
ridge2.fit(X_train, y_train)
```
Out[2377]:
```
▸ Ridge
```

In [2378]:
```
# Lets calculate some metrics such as R2 score, RSS and RMSE
y_pred_train = ridge2.predict(X_train)
y_pred_test = ridge2.predict(X_test)

metric_double = get_metrics_data(y_train, y_pred_train, y_test, y_pred_test)
```
```
R-Squared [Train data] ->  0.9472335637580859
R-Squared [Test data] ->  0.897932135861237
RSS [Train data] ->  6.8873332918135866
RSS [Test data] ->  5.199204320146091
RMSE [Train data] ->  0.08213205303034271
RMSE [Test data] ->  0.10895104174834877
```

In [2379]:
```
## Lasso Regression model with double value of alpha = 2 * 0.001
lasso2 = Lasso(alpha=0.002)

# Fit the model on training data
lasso2.fit(X_train, y_train)
```
Out[2379]:
```
▸ Lasso
```

In [2387]:
```
metric_double_lasso = get_metrics_data(y_train, y_pred_train, y_test, y_pred_test)
```
```
R-Squared [Train data] ->  0.9472335637580859
R-Squared [Test data] ->  0.897932135861237
RSS [Train data] ->  6.8873332918135866
RSS [Test data] ->  5.199204320146091
RMSE [Train data] ->  0.08213205303034271
RMSE [Test data] ->  0.10895104174834877
```

In [2388]:
```
# Combining to table
draw_table('Ridge Regression', metric2, ['Ridge Regression (2X Alpha)', 'Lasso Regression', 'Lasso Regression (2X Alpha)'],
           [metric_double, metric3, metric_double_lasso])
```
Out[2388]:

	Metric	Ridge Regression	Ridge Regression (2X Alpha)	Lasso Regression	Lasso Regression (2X Alpha)
0	R2 Score (Train)	0.954961	0.947234	0.929503	0.947234
1	R2 Score (Test)	0.898857	0.897932	0.893315	0.897932
2	RSS (Train)	5.878695	6.887333	9.201675	6.887333
3	RSS (Test)	5.152112	5.199204	5.434409	5.199204
4	RMSE (Train)	0.075880	0.082132	0.094934	0.082132
5	RMSE (Test)	0.108457	0.108951	0.111388	0.108951

In [2389]:
```
## Lets observe the changes in the coefficients
betas_2 = pd.DataFrame(index=X.columns)
```

In [2390]:
```
betas_2.rows = X.columns
```

In [2391]:
```
betas_2['Ridge2'] = ridge2.coef_
betas_2['Lasso2'] = lasso2.coef_
```

Top 10 Ridge regression coefficients with 2x Alpha

In [2393]:
```
betas_2['Ridge2'].sort_values(ascending=False)[:10]
```
Out[2393]:
```
OverallQual_9           0.066585
Neighborhood_Crawfor    0.063865
OverallQual_8           0.059824
Functional_Typ          0.050927
OverallCond_7           0.043785
OverallCond_9           0.043653
Exterior1st_BrkFace     0.042566
CentralAir_Y            0.041252
Condition1_Norm         0.040695
OverallCond_8           0.039620
Name: Ridge2, dtype: float64
```

Top ten lasso regression coeffients with 2x Alpha

In [2394]:
```
betas_2['Lasso2'].sort_values(ascending=False)[:10]
```
Out[2394]:
```
OverallQual_9           0.085897
OverallQual_8           0.078406
KitchenAbvGr_1          0.073825
Neighborhood_Crawfor    0.068698
CentralAir_Y            0.064850
Functional_Typ          0.055678
OverallQual_7           0.050408
Condition1_Norm         0.042821
OverallCond_7           0.042014
Foundation_PConc        0.038271
Name: Lasso2, dtype: float64
```

Q2: You have determind the optimal value of lambda for ridge and lasso regression during the assignment. Now, which one will you choose to apply and why ?

A: As both regression method allow to use correlated predictors but multicollinearity is handled differently in both:

- For ridge regression coefficients of correlated predictors are similar and it works well if there are large parameters of same value
- For lasso one of the correlated predictors has larger coefficient and does it if there are small number of significant parameters and others are close to zero.

Lasso regression would be better option if we have to set some feature coefficients to zero, removing them from the model and we have very less number of significants to decide on the other hand ridge should be preferred if regularization parameter is too large as lass can set too many feature coefficients to 0.

So if we consider we have large number of significants to calculate ridge regression should be preferred.

Q3: 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 excluding the five most important predictor variables. Which are the five most important predictor variables now?

A: Top five after removing and creating another model is **Neighborhood_Somerst, CentralAir_Y, OverallCond_9, Condition1_Norm,Exterior1st_BrkFace**

Below is the screenshot and solution also included in the Jupyter notebook:

Changes for subjective questions - Q3 - [please refer pdf]

In [2490]:
```
top_5_lasso = ['OverallQual_9', 'OverallQual_8','KitchenAbvGr_1','Neighborhood_Crawfor','Functional_Typ']
```

In [2491]:
```
X_train_modified =  X_train.drop(top_5_lasso, axis=1)
X_test_modified = X_test.drop(top_5_lasso, axis=1)
```

In [2492]:
```
lassoModified = Lasso()

# cross validation
model_cv_lasso_modified = GridSearchCV(estimator = lassoModified,
                        param_grid = params,
                        scoring= 'neg_mean_absolute_error',
                        cv = folds,
                        return_train_score=True,
                        verbose = 1)

model_cv_lasso_modified.fit(X_train_modified, y_train)
```

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

Out[2492]:
```
GridSearchCV
estimator: Lasso
  Lasso
```

In [2493]:
```
model_cv_lasso_modified.best_params_
```

Out[2493]:
```
{'alpha': 0.001}
```

In [2494]:
```
lasso_modified = Lasso(alpha=0.001)
```

In [2495]:
```
lasso_modified.fit(X_train_modified, y_train)
```

Out[2495]:
```
Lasso
Lasso(alpha=0.001)
```

In [2496]:
```
y_train_pred_mod = lasso_modified.predict(X_train_modified)
y_test_pred_mod = lasso_modified.predict(X_test_modified)
```

In [2497]:
```
modified_metrics = get_metrics_data(y_train, y_train_pred_mod, y_test, y_test_pred_mod)
```

```
R-Squared [Train data] ->  0.9240504304079353
R-Squared [Test data] ->  0.8863117757962719
RSS [Train data] ->  9.91360607733844
RSS [Test data] ->  5.7913011926194224
RMSE [Train data] ->  0.09853782181701184
RMSE [Test data] ->  0.11498760299091748
```

In [2498]:
```
betas = pd.DataFrame(index=X_train_modified.columns)
betas.rows = X_train_modified.columns
betas['Lasso_modified'] = lasso_modified.coef_
```

In [2499]:
```
### arrange in sort order to get top 5
betas['Lasso_modified'].sort_values(ascending=False)[:5]
```

Out[2499]:
```
Neighborhood_Somerst    0.068430
CentralAir_Y            0.058466
OverallCond_9           0.057865
Condition1_Norm         0.051089
Exterior1st_BrkFace     0.049657
Name: Lasso_modified, dtype: float64
```

Q 4: 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?

A: To make a robust model it is important to ensure the model's stability and accuracy

- Use Data Preprocessing: Data Cleaning and applying EDA
- Feature Scaling: Preprocess data by identifying and handling outliers
- Model selection, evaluation and optimazation: Evaluate best fit model based on complexity of data
- Test model against train vs test data and evaluate model, it should not be underfit and it should not be overfit

If we try to train model based on data to make more accurate that can make our model complex and there might be a case where our model will lead to some biasness due to which it can be a overfitting or underfitting model.