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 7.0 and the optimal value of Alpha for lasso regression is 0.001. The value of R2 score almost remains constant varying by 0.001. Below is the screenshot of the statistics calculated

	Metric	Ridge Regression	Ridge Regression (2X Alpha)	Lasso Regression	Lasso Regression (2X Alpha)
0	R2 Score (Train)	0.916407	0.906849	0.897606	0.906849
1	R2 Score (Test)	0.856979	0.852263	0.846129	0.852263
2	RSS (Train)	13.943327	15.537663	17.079402	15.537663
3	RSS (Test)	9.428391	9.739296	10.143683	9.739296
4	MSE (Train)	0.116861	0.123362	0.129337	0.123362
5	MSE (Test)	0.146717	0.149117	0.152181	0.149117

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

Top ten coefficients of Ridge regression

```
betas['Ridge'].sort_values(ascending=False)[:10]
```

```
FullBath_3              0.158822
OverallQual_9           0.154709
TotRmsAbvGrd_10         0.120687
OverallQual_10          0.111178
Neighborhood_NoRidge    0.111132
Neighborhood_Crawfor    0.105763
CentralAir_Y            0.100462
Neighborhood_StoneBr    0.098619
OverallQual_8           0.093166
Neighborhood_NridgHt    0.091525
Name: Ridge, dtype: float64
```

Top 10 Ridge regression coefficients with 2x Alpha

```
betas_2['Ridge2'].sort_values(ascending=False)[:10]
```

```
FullBath_3              0.129386
OverallQual_9           0.127257
TotRmsAbvGrd_10         0.108140
CentralAir_Y            0.096613
Neighborhood_NoRidge    0.090643
Neighborhood_Crawfor    0.088717
OverallQual_10          0.087615
Neighborhood_NridgHt    0.084298
BsmtExposure_Gd         0.083449
OverallQual_8           0.082174
Name: Ridge2, dtype: float64
```

Top ten coefficients of Lasso regression

```
betas['Lasso'].sort_values(ascending=False)[:10]
```

```
OverallQual_9           0.257618
OverallQual_10          0.191193
FullBath_3              0.176308
OverallQual_8           0.143808
Neighborhood_Crawfor    0.122150
CentralAir_Y            0.117224
TotRmsAbvGrd_10         0.114791
Neighborhood_NoRidge    0.102620
BsmtExposure_Gd         0.100626
Neighborhood_NridgHt    0.093432
Name: Lasso, dtype: float64
```

Top ten lasso regression coeffients with 2x Alpha

```
betas_2['Lasso2'].sort_values(ascending=False)[:10]
```

```
OverallQual_9           0.230837
OverallQual_10          0.154209
OverallQual_8           0.138221
FullBath_3              0.137802
CentralAir_Y            0.132348
GarageCars_3            0.107385
BsmtExposure_Gd         0.104751
TotRmsAbvGrd_10         0.096818
Neighborhood_Crawfor    0.094004
Neighborhood_NridgHt    0.086667
Name: Lasso2, dtype: float64
```

Screenshot for jupyter notebook solution for first question:

CHANGES FOR SUBJECTIVE QUESTIONS - Q1 [please refer pdf]

In [1815]:
```
## Ridge regression model with double value of alpha = 2*7
ridge2 = Ridge(alpha=14)

# Fit the model on training data
ridge2.fit(X_train, y_train)
```

Out[1815]:
```
Ridge
Ridge(alpha=14)
```

In [1816]:
```
# 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] ->  0.9068491399788395
R-Squared [Test data] ->  0.8522629194786264
RSS [Train data] ->  15.537663020069498
RSS [Test data] ->  9.739296113823944
RMSE [Train data] ->  0.12336159560969821
RMSE [Test data] ->  0.1491168530928358
```

In [1817]:
```
## 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[1817]:
```
Lasso
Lasso(alpha=0.002)
```

In [1818]:
```
metric_double_lasso = get_metrics_data(y_train, y_pred_train, y_test, y_pred_test)
```

```
R-Squared [Train data] ->  0.9068491399788395
R-Squared [Test data] ->  0.8522629194786264
RSS [Train data] ->  15.537663020069498
RSS [Test data] ->  9.739296113823944
RMSE [Train data] ->  0.12336159560969821
RMSE [Test data] ->  0.1491168530928358
```

In [1819]:
```
# 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[1819]:

	Metric	Ridge Regression	Ridge Regression (2X Alpha)	Lasso Regression	Lasso Regression (2X Alpha)
0	R2 Score (Train)	0.916407	0.906849	0.897606	0.906849
1	R2 Score (Test)	0.856979	0.852263	0.846129	0.852263
2	RSS (Train)	13.943327	15.537663	17.079402	15.537663
3	RSS (Test)	9.428391	9.739296	10.143683	9.739296
4	RMSE (Train)	0.116861	0.123362	0.129337	0.123362
5	RMSE (Test)	0.146717	0.149117	0.152181	0.149117

In [1820]:
```
## Lets observe the changes in the coefficients
betas_2 = pd.DataFrame(index=X.columns)
```

In [1821]:
```
betas_2.rows = X.columns
```

In [1822]:
```
betas_2['Ridge2'] = ridge2.coef_
betas_2['Lasso2'] = lasso2.coef_
```

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 **TotRmsAbvGrd_10, CentralAir_Y, Neighborhood_NridgHt, Neighborhood_NoRidge, BsmtExposure_Gd**

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

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

In [1661]:
```
top_5_lasso = ['OverallQual_9', 'OverallQual_10','FullBath_3','OverallQual_8','Neighborhood_Crawfor']
```

In [1662]:
```
X_train_modified =  X_train.drop(top_5_lasso, axis=1)
X_test_modified = X_test.drop(top_5_lasso, axis=1)
```

In [1663]:
```
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[1663]:
```
▸ GridSearchCV
▸ estimator: Lasso
    ▸ Lasso
```

In [1664]:
```
model_cv_lasso_modified.best_params_
```

Out[1664]:
```
{'alpha': 0.001}
```

In [1740]:
```
lasso_modified = Lasso(alpha=0.001)
```

In [1741]:
```
lasso_modified.fit(X_train_modified, y_train)
```

Out[1741]:
```
▾       Lasso
Lasso(alpha=0.001)
```

In [1742]:
```
y_train_pred_mod = lasso_modified.predict(X_train_modified)
y_test_pred_mod = lasso_modified.predict(X_test_modified)
```

In [1749]:
```
modified_metrics = get_metrics_data(y_train, y_train_pred_mod, y_test, y_test_pred_mod)
```

```
R-Squared [Train data] ->  0.8904112976971699
R-Squared [Test data] ->  0.8342593753742075
RSS [Train data] ->  18.279512682988493
RSS [Test data] ->  10.926146744095183
RMSE [Train data] ->  0.1338041044851391
RMSE [Test data] ->  0.15794157309471385
```

In [1750]:
```
betas = pd.DataFrame(index=X_train_modified.columns)
betas.rows = X_train_modified.columns
betas['Lasso_modified'] = lasso_modified.coef_
```

In [1751]:
```
### arrange in sort order to get top 5
betas['Lasso_modified'].sort_values(ascending=False)[:5]
```

Out[1751]:
```
TotRmsAbvGrd_10         0.131094
CentralAir_Y            0.124631
Neighborhood_NridgHt    0.121757
Neighborhood_NoRidge    0.121068
BsmtExposure_Gd         0.111180
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.