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
Ridge Alpha. 2
Lasso. Alpha 0.01

Ridge Alpha. 4

```
# ridge regression
lm = Ridge(alpha=4)
lm.fit(X_train, y_train)

# predict
y_train_pred = lm.predict(X_train)
print(metrics.r2_score(y_true=y_train, y_pred=y_train_pred))
y_test_pred = lm.predict(X_test)
print(metrics.r2_score(y_true=y_test, y_pred=y_test_pred))
```

```
0.9293153913681106
0.8673123125036433
```

Lasso. Alpha 0.02

```
# lasso regression
la = Lasso(alpha=0.02)
la.fit(X_train, y_train)

# prediction on the test set(Using R2)
y_train_pred = la.predict(X_train)
print(metrics.r2_score(y_true=y_train, y_pred=y_train_pred))
y_test_pred = la.predict(X_test)
print(metrics.r2_score(y_true=y_test, y_pred=y_test_pred))
```

```
0.8274703555203093
0.8321493248301828
```

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:

The r2_score of ridge is slightly higher than lasso for the test dataset so we will choose ridge regression to solve this problem

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?

In [70]:
```
X_train.columns
```

Out[70]:
```
Index(['MSSubClass', 'LotArea', 'OverallQual', 'OverallCond', 'MasVnrArea',
       'BsmtFinSF1', 'BsmtFinSF2', 'BsmtUnfSF', 'TotalBsmtSF', '1stFlrSF',
       ...
       'SaleType_ConLI', 'SaleType_ConLw', 'SaleType_New', 'SaleType_Oth',
       'SaleType_WD', 'SaleCondition_AdjLand', 'SaleCondition_Alloca',
       'SaleCondition_Family', 'SaleCondition_Normal',
       'SaleCondition_Partial'],
      dtype='object', length=253)
```

In [74]:
```
X_train2 = X_train.drop(['MSSubClass', 'LotArea', 'OverallQual', 'OverallCond', 'MasVnrArea'],axis=1)
X_test2 = X_test.drop(['MSSubClass', 'LotArea', 'OverallQual', 'OverallCond', 'MasVnrArea'],axis=1)
```

In [75]:
```
X_train2.head()
```

Out[75]:

	BsmtFinSF1	BsmtFinSF2	BsmtUnfSF	TotalBsmtSF	1stFlrSF	2ndFlrSF	LowQualFinSF	GrLivArea	BsmtFullBath	BsmtHalfBath	FullBath	HalfBath	Be
984	-0.931228	-0.299629	-1.265589	-2.274081	0.320038	0.206085	-0.123208	0.399579	-0.826986	-0.241465	0.789648	-0.755902	
1416	-0.931228	-0.299629	0.464070	-0.607802	0.179679	1.610886	-0.123208	1.449374	-0.826986	-0.241465	0.789648	-0.755902	
390	-0.422211	1.997881	-0.755819	-0.427663	-0.534650	0.146404	-0.123208	-0.293361	1.087646	-0.241465	-1.022106	-0.755902	
877	1.539931	-0.299629	-0.615576	0.861183	0.721065	0.963575	-0.123208	1.324758	1.087646	-0.241465	0.789648	1.269756	
567	-0.927003	-0.299629	2.106912	0.979131	0.904033	-0.785539	-0.123208	0.023843	-0.826986	-0.241465	0.789648	-0.755902	

5 rows × 248 columns

In [76]:
```
X_test2.head()
```

Out[76]:

	BsmtFinSF1	BsmtFinSF2	BsmtUnfSF	TotalBsmtSF	1stFlrSF	2ndFlrSF	LowQualFinSF	GrLivArea	BsmtFullBath	BsmtHalfBath	FullBath	HalfBath	Bed
789	0.268446	-0.299629	-0.677906	-0.489854	-0.497053	1.764679	-0.123208	1.066085	-0.826986	-0.241465	0.789648	1.269756	
425	-0.931228	-0.299629	0.183585	-0.878009	-1.171279	0.779941	-0.123208	-0.251823	-0.826986	-0.241465	-1.022106	1.269756	
204	-0.272252	-0.299629	-0.357351	-0.730038	-1.138696	0.509081	-0.123208	-0.450075	-0.826986	-0.241465	-1.022106	1.269756	
118	2.173561	-0.299629	-0.813696	1.313673	1.315085	2.710395	-0.123208	3.209102	1.087646	-0.241465	2.601402	-0.755902	
244	0.536683	-0.299629	-0.702393	-0.241092	0.119525	1.252799	-0.123208	1.109512	1.087646	-0.241465	0.789648	1.269756	

5 rows × 248 columns

In [77]:
```
# lasso regression
la = Lasso(alpha=0.01)
la.fit(X_train2, y_train)

# prediction on the test set(Using R2)
y_train_pred = la.predict(X_train2)
print(metrics.r2_score(y_true=y_train, y_pred=y_train_pred))
y_test_pred = la.predict(X_test2)
print(metrics.r2_score(y_true=y_test, y_pred=y_test_pred))
```

```
0.8180356742698357
0.7803028122712371
```

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 should be generalized so that the test accuracy is not lesser than the training score. The model should be accurate for datasets other than the ones which were used during training. Too much importance should not given to the outliers so that the accuracy predicted by the model is high. To ensure that this is not the case, the outliers analysis needs to be done and only those which are relevant to the dataset need to be retained. Those outliers which it does not make sense to keep must be removed from the dataset. If the model is not robust, It cannot be trusted for predictive analysis.