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 optimum values for ridge and lasso are 50 and 0.001 respectively.

Below is a comparison of top features with original alphas and after doubling the values:

Based on Ridge Regression

```
a = pd.DataFrame(X_train.columns)
b = pd.DataFrame(ridge.coef_)
```

```
coefficients = pd.concat([a,b], axis=1)
coefficients.columns = ['Features','Coefficient']
```

```
coefficients.sort_values(by='Coefficient', ascending=False).head(10)
```

	Features	Coefficient
2	OverallQual	0.196706
80	Neighborhood_NoRidge	0.192311
81	Neighborhood_NridgHt	0.174011
18	GrLivArea	0.165329
91	HouseStyle_1Story	0.141040
16	2ndFlrSF	0.120117
30	GarageCars	0.103119
8	BsmtExposure	0.095484
15	1stFlrSF	0.091253
71	Neighborhood_Crawfor	0.089360

Ridge with doubled alpha

```
a = pd.DataFrame(X_train.columns)
b = pd.DataFrame(ridge2.coef_)
```

```
coefficients = pd.concat([a,b], axis=1)
coefficients.columns = ['Features','Coefficient']
```

```
coefficients.sort_values(by='Coefficient', ascending=False).head(10)
```

	Features	Coefficient
2	OverallQual	0.184294
18	GrLivArea	0.147662
80	Neighborhood_NoRidge	0.123972
81	Neighborhood_NridgHt	0.106785
16	2ndFlrSF	0.097768
91	HouseStyle_1Story	0.093152
30	GarageCars	0.092249
15	1stFlrSF	0.091881
8	BsmtExposure	0.087993
25	KitchenQual	0.081235

As seen from above results screenshots, OverallQuall remains top features even with doubled alpha but with reduced values of co efficient. With double Alpha, GrLivArea moved up to 2nd place and Neighborhood_NoRidge and Neighborhood_NridgHt moved down to 3rd and 4th place .

Based on Lasso Regression

```
a = pd.DataFrame(X_train.columns)
b = pd.DataFrame(lasso.coef_)
```

```
coefficients = pd.concat([a,b], axis=1)
coefficients.columns = ['Features','Coefficient']
```

```
coefficients.sort_values(by='Coefficient', ascending=False).head(10)
```

	Features	Coefficient
81	Neighborhood_NridgHt	0.602054
80	Neighborhood_NoRidge	0.595412
87	Neighborhood_StoneBr	0.330668
91	HouseStyle_1Story	0.311616
86	Neighborhood_Somerst	0.309355
71	Neighborhood_Crawfor	0.299277
18	GrLivArea	0.287606
122	Exterior2nd_ImStucc	0.223743
2	OverallQual	0.196574
104	Exterior1st_BrkFace	0.183175

Lasso with doubled alpha

```
a = pd.DataFrame(X_train.columns)
b = pd.DataFrame(lasso2.coef_)
```

```
coefficients = pd.concat([a,b], axis=1)
coefficients.columns = ['Features','Coefficient']
```

```
coefficients.sort_values(by='Coefficient', ascending=False).head(10)
```

	Features	Coefficient
80	Neighborhood_NoRidge	0.562109
81	Neighborhood_NridgHt	0.520797
18	GrLivArea	0.298478
91	HouseStyle_1Story	0.262482
71	Neighborhood_Crawfor	0.247066
86	Neighborhood_Somerst	0.242775
2	OverallQual	0.204301
87	Neighborhood_StoneBr	0.190948
104	Exterior1st_BrkFace	0.149129
30	GarageCars	0.112274

As seen from above results of Lasso with doubled alpha, Neighborhood_NoRidge moved up to top and Neighborhood_NridHt moved to second position. GrLivArea moved up to 3rd spot and House_Style_1Story remained at 5th spot.

After the change in implemented, below are the overall top features based on ridge and lasso respectively :

Top features based on changed Ridge aplha

```
a = pd.DataFrame(X_train.columns)
b = pd.DataFrame(ridge2.coef_)
```

```
coefficients = pd.concat([a,b], axis=1)
coefficients.columns = ['Features','Coefficient']
```

```
coefficients.sort_values(by='Coefficient', ascending=False).head(10)
```

	Features	Coefficient
2	OverallQual	0.184294
18	GrLivArea	0.147662
80	Neighborhood_NoRidge	0.123972
81	Neighborhood_NridgHt	0.106785
16	2ndFlrSF	0.097768
91	HouseStyle_1Story	0.093152
30	GarageCars	0.092249
15	1stFlrSF	0.091881
8	BsmtExposure	0.087993
25	KitchenQual	0.081235

Top features based on Lasso changed alpha

```
a = pd.DataFrame(X_train.columns)
b = pd.DataFrame(lasso2.coef_)
```

```
coefficients = pd.concat([a,b], axis=1)
coefficients.columns = ['Features','Coefficient']
```

```
coefficients.sort_values(by='Coefficient', ascending=False).head(10)
```

	Features	Coefficient
80	Neighborhood_NoRidge	0.562109
81	Neighborhood_NridgHt	0.520797
18	GrLivArea	0.298478
91	HouseStyle_1Story	0.262482
71	Neighborhood_Crawfor	0.247066
86	Neighborhood_Somerst	0.242775
2	OverallQual	0.204301
87	Neighborhood_StoneBr	0.190948
104	Exterior1st_BrkFace	0.149129
30	GarageCars	0.112274

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 2:

Based on the R squared and RMSE values(as shown below), I will select Lasso regression as its R squared and RMSE in test are better.

Out[2627]:

	R2 score(Train)	R2 score(Test)	RMSE
Ridge Regression	0.851363	0.854741	0.381128
Lasso Regression	0.869804	0.858460	0.376218
Ridge Regression 2	0.841478	0.851953	0.384769
Lasso Regression 2	0.864228	0.858475	0.376198

With doubled alpha, also Lasso (Lasso Regression 2) is giving better results in test.

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 3:

Below are the most important predictors after removing top 5 from lasso:

	Features	Coefficient
18	GrLivArea	0.314908
117	Exterior2nd_ImStucc	0.294312
2	OverallQual	0.204423
99	Exterior1st_BrkFace	0.190671
131	Foundation_Slab	0.165208
30	GarageCars	0.127561
57	MSZoning_RL	0.109285
127	MasVnrType_None	0.100094
8	BsmtExposure	0.092186
62	LotConfig_CulDSac	0.090043
71	Neighborhood_Crawfor	0.084298

GrLivArea, Exterior2nd_ImStucc, OverallQual, Exterior1st_BrkFace and Foundation_Slab will be the top 5 new predictors.

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?

To make a model robust and generalisable, following steps can be taken:

1. Data should be split into test and train. This will ensure that model is trained and tested on separate datasets and hence reduce chances of overfitting.
2. Cross validation can also be used to test data. This would reduce variance in the model.
3. Technique of regularization can be used. This helps in preventing overfitting by penalizing the coefficient of features.
4. Methods like RFE can be used for feature selection.

Following are the implications of above on accuracy:

1. Splitting the data in train and test and comparing the R squared(R2) values can give a good idea if the model is giving good results or overfitting. If the difference between R2 of train and test is very high, then model must be overfitting.
2. Cross validation helps improve model performance.
3. Regularization can have impact on accuracy as the coefficients can be sometimes over compensated.