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?

The optimal value of

Ridge = 4

Lasso = 100

After doubling the values as well, the co-efficient remain the same

What will be the changes in the model if you choose double the value of alpha for both ridge a

```
alpha = 8
ridge2 = Ridge(alpha=alpha)
ridge2.fit(X_train, y_train)
```

36] ✓ 0.0s

```
Ridge(alpha=8)
```

```python
y_pred_train = ridge2.predict(X_train)
y_pred_test =  ridge2.predict(X_test)

rdige_metric2 = []

r2_train = r2_score(y_train, y_pred_train)
print(r2_train)
rdige_metric2.append(r2_train)


r2_test = r2_score(y_test, y_pred_test)
print(r2_test)
rdige_metric2.append(r2_test)

rss1 = np.sum(np.square(y_train - y_pred_train))
print(rss1)
rdige_metric2.append(rss1)

rss2_lr = np.sum(np.square(y_test - y_pred_test))
print(rss2_lr)
rdige_metric2.append(rss2_lr)

mse_train_lr = mean_squared_error(y_train, y_pred_train)
print(mse_train_lr)
rdige_metric2.append(mse_train_lr**0.5)

mse_test_lr = mean_squared_error(y_test, y_pred_test)
print(mse_test_lr)
rdige_metric2.append(mse_test_lr**0.5)
```

9] ✓ 0.0s

```
0.9396318482305696
0.90254194179256
275826847753.88196
172909770574.31647
312728852.32866436
457433255.4876097
```

```python
#Optimum Value of alpha is 200

alpha =200

lasso2 = Lasso(alpha=alpha)

lasso2.fit(X_train, y_train)


y_pred_train = lasso2.predict(X_train)
y_pred_test = lasso2.predict(X_test)

lasso_metric2 = []
r2_train_lr = r2_score(y_train, y_pred_train)
print(r2_train_lr)
lasso_metric2.append(r2_train_lr)

r2_test_lr = r2_score(y_test, y_pred_test)
print(r2_test_lr)
lasso_metric2.append(r2_test_lr)

rss1_lr = np.sum(np.square(y_train - y_pred_train))
print(rss1_lr)
lasso_metric2.append(rss1_lr)

rss2_lr = np.sum(np.square(y_test - y_pred_test))
print(rss2_lr)
lasso_metric2.append(rss2_lr)

mse_train_lr = mean_squared_error(y_train, y_pred_train)
print(mse_train_lr)
lasso_metric2.append(mse_train_lr**0.5)

mse_test_lr = mean_squared_error(y_test, y_pred_test)
print(mse_test_lr)
lasso_metric2.append(mse_test_lr**0.5)
```

✓ 0.0s

```
0.9321577293820309
0.9026620641667991
309976686390.72144
172696649847.79532
351447490.2389132
456869444.04178655
```

```
double_alpha_df = pd.DataFrame(index=X_train.columns)
double_alpha_df.rows = X_train.columns
double_alpha_df['Ridge2'] = ridge2.coef_
double_alpha_df['Ridge'] = ridge.coef_
double_alpha_df['Lasso'] = lasso.coef_
double_alpha_df['Lasso20'] = lasso2.coef_
pd.set_option('display.max_rows', None)
double_alpha_df.head(10)
```

✓ 0.0s

	Ridge2	Ridge	Lasso	Lasso20
MSSubClass	-12842.164103	-13285.773812	-16146.847489	-15254.800812
LotFrontage	4236.131536	3309.979863	0.000000	0.000000
LotArea	22123.278264	24547.651358	25675.309711	22294.656850
OverallQual	40122.010640	46087.725642	68214.857885	75192.519022
OverallCond	18337.082587	23591.893319	30409.838091	21601.595049
YearBuilt	12878.543253	19444.961233	33108.813179	23776.159062
YearRemodAdd	13913.631905	12433.945485	12302.680172	14794.908882
MasVnrArea	22179.762767	22807.443595	21262.567070	20094.899507
BsmtFinSF1	32665.175557	36990.151755	30126.565310	29906.492517
BsmtFinSF2	7409.439242	11140.659905	0.000000	0.000000

```
- OverallQual
- OverallCond
- YearBuilt
- Neighborhood_StoneBr
- Exterior1st_BrkFace
- TotalBsmtSF
- LotArea
```

Above are the important predictor variables.

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?

Both Lasso and Ridge has almost similar R2 score, But I do notice the test R2 score for Lasso is slightly higher than the Ridge. Hence I choose Lasso.

```
final_metric
```

✓ 0.0s

	Metric	Ridge Regression	Lasso Regression
0	R2 Score (Train)	9.464520e-01	9.414127e-01
1	R2 Score (Test)	9.029362e-01	9.061680e-01
2	RSS (Train)	2.446648e+11	2.676902e+11
3	RSS (Test)	1.722103e+11	1.664765e+11
4	MSE (Train)	1.665526e+04	1.742135e+04
5	MSE (Test)	2.134439e+04	2.098604e+04

Question 3

After building the model, you realized 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?

```python
X_train_new = X_train.drop(columns=['LotArea', 'OverallQual', 'OverallCond', 'YearBuilt
X_train_new.head(10)
```

✓ 0.0s

	MSSubClass	LotFrontage	YearRemodAdd	MasVnrArea	BsmtFinSF1	BsmtFinSF2	BsmtUn
546	0.176471	0.167808	0.000000	0.000000	0.157563	0.0	0.355
274	0.000000	0.188356	0.533333	0.000000	0.286765	0.0	0.125
1216	0.411765	0.160959	0.466667	0.000000	0.000000	0.0	0.000
793	0.000000	0.188356	0.950000	0.181347	0.000000	0.0	0.694
169	0.000000	0.164384	0.516667	0.845855	0.000000	0.0	0.783
1285	0.176471	0.099315	0.000000	0.000000	0.000000	0.0	0.362
518	0.235294	0.164384	0.800000	0.000000	0.370798	0.0	0.040
282	0.588235	0.044521	0.966667	0.215026	0.474790	0.0	0.190
994	0.000000	0.256849	0.950000	0.297927	0.615546	0.0	0.245
375	0.058824	0.164384	0.000000	0.000000	0.183824	0.0	0.154

```python
X_test_new = X_test.drop(columns=['LotArea', 'OverallQual', 'OverallCond', 'YearBuilt',
X_test_new.head(10)
```

✓ 0.0s

	MSSubClass	LotFrontage	YearRemodAdd	MasVnrArea	BsmtFinSF1	BsmtFinSF2	BsmtUn
299	0.000000	0.202055	0.900000	0.000000	0.000000	0.0	0.507
294	0.000000	0.202055	0.050000	0.308290	0.674895	0.0	0.060

```python
alpha =100

lasso3 = Lasso(alpha=alpha)

lasso3.fit(X_train_new, y_train)


y_pred_train = lasso3.predict(X_train_new)
y_pred_test = lasso3.predict(X_test_new)

lasso_metric3 = []
r2_train_lr = r2_score(y_train, y_pred_train)
print(r2_train_lr)
lasso_metric3.append(r2_train_lr)

r2_test_lr = r2_score(y_test, y_pred_test)
print(r2_test_lr)
lasso_metric3.append(r2_test_lr)

rss1_lr = np.sum(np.square(y_train - y_pred_train))
print(rss1_lr)
lasso_metric3.append(rss1_lr)

rss2_lr = np.sum(np.square(y_test - y_pred_test))
print(rss2_lr)
lasso_metric3.append(rss2_lr)

mse_train_lr = mean_squared_error(y_train, y_pred_train)
print(mse_train_lr)
lasso_metric3.append(mse_train_lr**0.5)

mse_test_lr = mean_squared_error(y_test, y_pred_test)
print(mse_test_lr)
lasso_metric3.append(mse_test_lr**0.5)
```

] ✓ 0.0s

```
0.9329845031703407
0.8876962657717308
306199091729.4502
```

```python
# R2 calculation
alpha = 4
ridge3 = Ridge(alpha=alpha)
ridge3.fit(X_train_new, y_train)
y_pred_train = ridge3.predict(X_train_new)
y_pred_test =  ridge3.predict(X_test_new)

rdige_metric3 = []

r2_train = r2_score(y_train, y_pred_train)
print(r2_train)
rdige_metric3.append(r2_train)


r2_test = r2_score(y_test, y_pred_test)
print(r2_test)
rdige_metric3.append(r2_test)

rss1 = np.sum(np.square(y_train - y_pred_train))
print(rss1)
rdige_metric3.append(rss1)

rss2_lr = np.sum(np.square(y_test - y_pred_test))
print(rss2_lr)
rdige_metric3.append(rss2_lr)

mse_train_lr = mean_squared_error(y_train, y_pred_train)
print(mse_train_lr)
rdige_metric3.append(mse_train_lr**0.5)

mse_test_lr = mean_squared_error(y_test, y_pred_test)
print(mse_test_lr)
rdige_metric3.append(mse_test_lr**0.5)
```

✓ 0.0s

```
0.9392498120919069
```

```python
lr_table = {'Metric': ['R2 Score (Train)','R2 Score (Test)','RSS (Train)','RSS (Test)',
                       'MSE (Train)','MSE (Test)']
           }

lr_metric = pd.DataFrame(lr_table ,columns = ['Metric'] )

rg_metric = pd.Series(rdige_metric3, name = 'Ridge Regression')
ls_metric = pd.Series(lasso_metric2, name = 'Lasso Regression')

final_metric2 = pd.concat([lr_metric, rg_metric, ls_metric], axis = 1)

final_metric2
```

2] ✓ 0.0s

	Metric	Ridge Regression	Lasso Regression
0	R2 Score (Train)	9.392498e-01	9.321577e-01
1	R2 Score (Test)	8.912240e-01	9.026621e-01
2	RSS (Train)	2.775724e+11	3.099767e+11
3	RSS (Test)	1.929900e+11	1.726966e+11
4	MSE (Train)	1.774001e+04	1.874693e+04
5	MSE (Test)	2.259548e+04	2.137450e+04

```python
#important predictor variables
betas = pd.DataFrame(index=X_train_new.columns)
betas.rows = X_train_new.columns
betas['Lasso3'] = lasso3.coef_
pd.set_option('display.max_rows', None)
betas.head(15)
```

3] ✓ 0.0s

[103] ✓ 0.0s

	Lasso3
MSSubClass	-20379.893153
LotFrontage	0.000000
YearRemodAdd	18741.477607
MasVnrArea	24037.883483
BsmtFinSF1	46525.377176
BsmtFinSF2	6949.930108
BsmtUnfSF	18356.706123
1stFlrSF	17094.458522
2ndFlrSF	0.000000
LowQualFinSF	-17370.049670
GrLivArea	175167.702352
BsmtFullBath	4578.330253
BsmtHalfBath	0.000000
FullBath	0.000000
HalfBath	2970.450981

```
The next five important predictors are


-LotFrontage
-MasVnrArea
-BsmtFinSF1
-GrLivArea
-YearRemodAdd
```

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?

The Regression model calculated should not decrease the value of the test data significantly. We should make sure the outliers are not provided high importance during building a model.
It is also important to avoid overfitting the training data, due to which the model might not work well on the unseen test data.
A generalized model will have a good trade-off between bias and variance.