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?

- **Optimal value calculated for both regularization models:**

Optimal value of alpha in Ridge:

8.0

Optimal value of alpha in Lasso:

0.00018

```
In [203]: print(model_cv_ridge.best_params_)
          print(model_cv_lasso_opt.best_params_)

{'alpha': 8.0}
{'alpha': 0.00018}
```

- **Analysis after doubling the alpha:**

For Lasso:

At alpha = 0.00018

```
R2 for train set:  0.9178563572706558
R2 for test set: 0.7762235732511631
```

At alpha = 0.00036

```
R2 for train set: 0.8756792410107785
R2 for test set: 0.8460443051326964
```

Model accuracy has been increased also r2 for test and train has come clos
er and have decent value, means model is better than earlier.

In [203]:
```python
print(model_cv_ridge.best_params_)
print(model_cv_lasso_opt.best_params_)
```

```
{'alpha': 8.0}
{'alpha': 0.00018}
```

In [204]:
```python
#Fitting Lasso model for alpha = 0.00036 and printing coefficients which have been penalised

alpha =0.00036

lasso3 = Lasso(alpha=alpha)

lasso3.fit(x_train, y_train)
```

Out[204]:
```
Lasso(alpha=0.00036)
```

In [205]:
```python
# Lets calculate some metrics such as R2 score, RSS and RMSE
y_pred_train = lasso3.predict(x_train)
y_pred_test = lasso3.predict(x_test_new)

metric2 = []
r2_train_lr = r2_score(y_train, y_pred_train)
print('r2 Train')
print(r2_train_lr)
metric2.append(r2_train_lr)

r2_test_lr = r2_score(y_test, y_pred_test)
print('r2 Test')
print(r2_test_lr)
metric2.append(r2_test_lr)
```

```
r2 Train
0.8756792410107785
r2 Test
0.8460443051326964
```

For Ridge:

At alpha = 8.0

R2 for train set: 0.8946506403409424
R2 for test set: 0.8587930428723037

At alpha = 16.0

R2 for train set: 0.8771528066322968

R2 for test set: 0.8505278962404824

Model accuracy has slightly decreased for both train and test.
Means our previous alpha 8.0 is performing better than its twice 16.0.

In [206]:
```python
#Fitting Ridge model for alpha = 16 and printing coefficients which have been penalis
alpha = 16.0
ridge2 = Ridge(alpha=alpha)

ridge2.fit(x_train, y_train)
```

Out[206]:
```
Ridge(alpha=16.0)
```

In [207]:
```python
# 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_new)

metric2 = []
r2_train_lr = r2_score(y_train, y_pred_train)
print('r2 Train')
print(r2_train_lr)
metric2.append(r2_train_lr)

r2_test_lr = r2_score(y_test, y_pred_test)
print('r2 Test')
print(r2_test_lr)
metric2.append(r2_test_lr)
```

```
r2 Train
0.8771528066322968
r2 Test
0.8505278962404824
```

- **Important predictor variables after the changes are implemented:**

Lasso:

GrLivArea, OverallQual, Neighborhood_NoRidge, GarageCars, PoolQC_Gd

In [211]:
```python
coef_df_lasso3_abs = coef_df_lasso3[0].abs()
```

In [216]:
```python
coef_df_lasso3_abs = pd.DataFrame(coef_df_lasso3_abs)
```

In [219]:
```python
coef_df_lasso3_abs.sort_values(0, axis = 0, ascending = False)
```

Out[219]:

	0
GrLivArea	0.299495
OverallQual	0.149536
PoolQC_Gd	0.093209
Neighborhood_NoRidge	0.067711
GarageCars	0.056264
RoofMatl_WdShngl	0.041293
Neighborhood_NridgHt	0.040054

Ridge:

OverallQual, Neighborhood_NoRidge, GrLivArea, 2ndFlrSF, TotRmsAbvGrd, FullBath

```
n [221]: coef_df_ridge2_abs = coef_df_ridge2[0].abs()
         coef_df_ridge2_abs = pd.DataFrame(coef_df_ridge2_abs)
         coef_df_ridge2_abs.sort_values(0, axis = 0, ascending = False)
```

ut[221]:

	0
OverallQual	0.058895
Neighborhood_NoRidge	0.050230
GrLivArea	0.048855
2ndFlrSF	0.047678
TotRmsAbvGrd	0.038989
FullBath	0.038116
1stFlrSF	0.036383
GarageCars	0.036046
BsmtQual_Gd	0.033574
KitchenQual_TA	0.030931
Neighborhood_NridgHt	0.030666
BsmtExposure_Gd	0.029341

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?

Ans:

After doubling the alpha in Lasso, we have observed major improvement in model accuracy, and it has come also near to scores for Ridge regression. Also, the diff between train and test scores are smaller for Lasso.

Lasso : At alpha = 0.00036
R2 train set: 0.8756792410107785   R2 test set: 0.8460443051326964

Ridge: At alpha = 8.0
R2 train set: 0.8946506403409424   R2 test set: 0.8587930428723037

With Lasso regression results in model parameters such that lesser important features coefficients beco me zero.

So, we will opt for Lasso regression in this case.

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?

Ans:

Top 5 predicating variables in lasso regression are:

GrLivArea, OverallQual, PoolQC_Gd, Neighborhood_NoRidge, GarageCars

After removing these features, we have rebuilt the model and below are the new top 5 important predict or variables:

1. **1stFlrSF**
2. **2ndFlrSF**
3. **Condition2_PosN**
4. **RoofMatl_WdShngl**
5. **GarageArea**

36]:

```
coef_df_lasso_rev5_abs = coef_df_lasso_rev5[0].abs()
coef_df_lasso_rev5_abs = pd.DataFrame(coef_df_lasso_rev5_abs)
coef_df_lasso_rev5_abs.sort_values(0, axis = 0, ascending = False)
```

36]:

	0
1stFlrSF	0.281099
2ndFlrSF	0.176466
Condition2_PosN	0.125862
RoofMatl_WdShngl	0.086365
GarageArea	0.060920
MSSubClass	0.044342
KitchenQual_TA	0.044260

Question 4

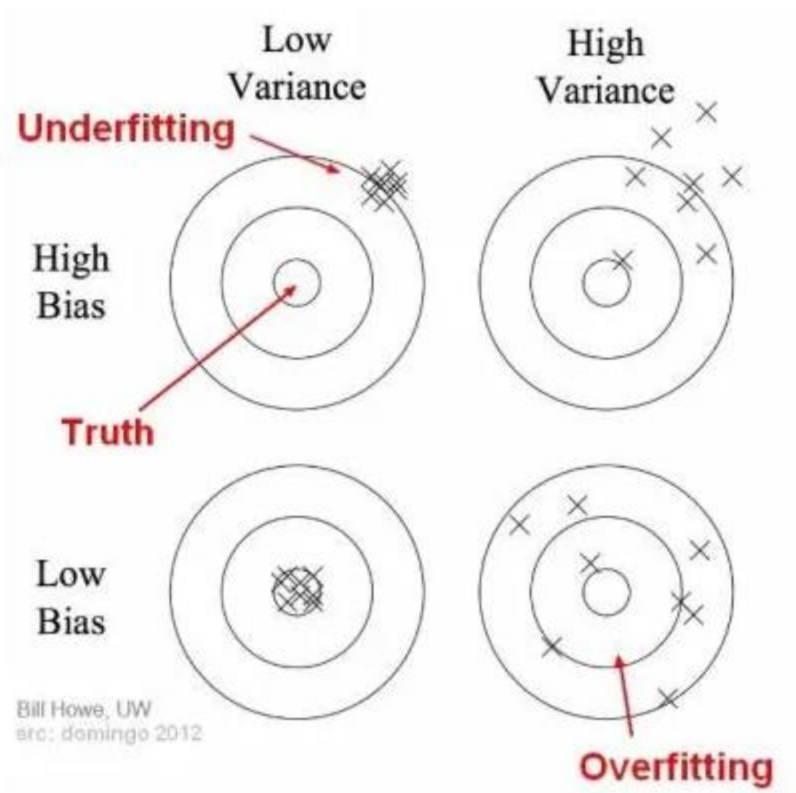
How can you make sure that a model is robust and generalizable? What are the impli cations of the same for the accuracy of the model and why?

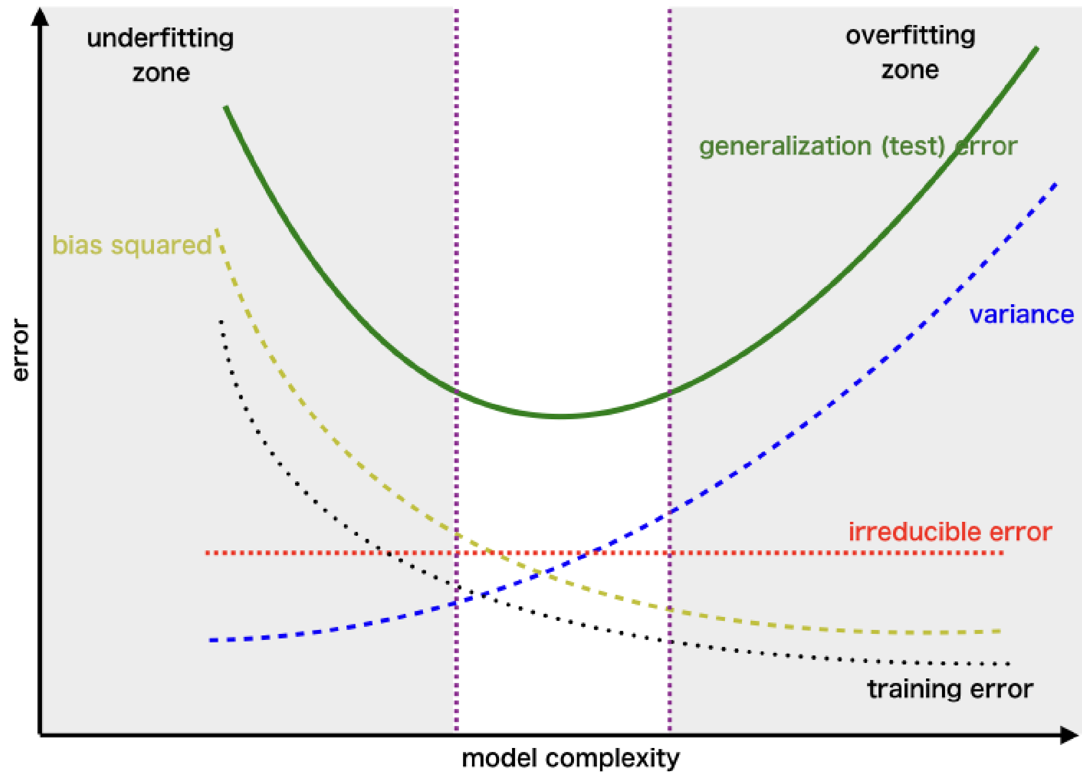
Ans:

The model should be as simple as possible, though its accuracy will decrease but it will be more robust a nd generalizable. It can be also understood using the Bias-Variance trade-off. The simpler the model the more the bias but less variance and more generalizable. Its implication in terms of accuracy is that a robu st and generalizable model will perform equally well on both training and test data i.e., the accuracy doe s not change much for training and test data.

Bias: Bias is error in model, when the model is weak to learn from the data. High bias means model is un able to learn details in the data. Model performs poor on training and testing data.

Variance: Variance is error in model, when model tries to over learn from the data. High variance means model performs exceptionally well on training data as it has very well trained on this of data but perform s very poor on testing data as it was unseen data for the model. It is important to have balance in Bias an d Variance to avoid overfitting and under-fitting of data.




underfitting zone
overfitting zone
generalization (test) error
bias squared
variance
error
irreducible error
training error
model complexity