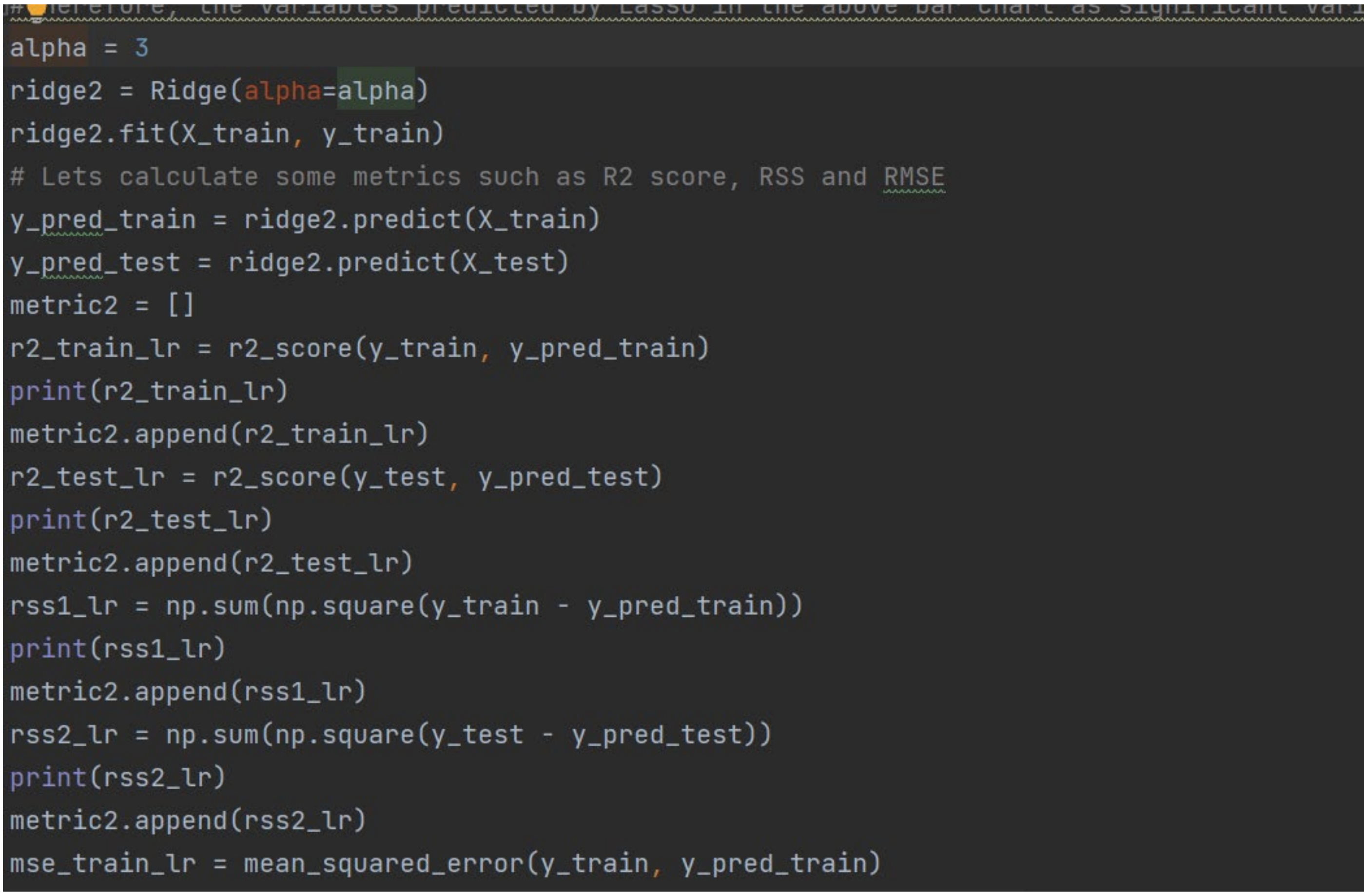
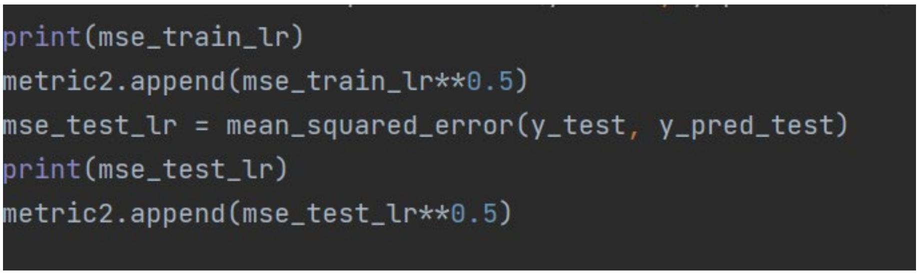
Q1: What is the optimal value of alpha for ridge and lasso regression? What will be the changes in the model if you choose double the value of alpha for both ridge and lasso? What will be the most important predictor variables after the change is implemented?

Ans:

The optimal value of alpha for ridge and lasso regression are Ridge Alpha 1 and lasso Alpha 10

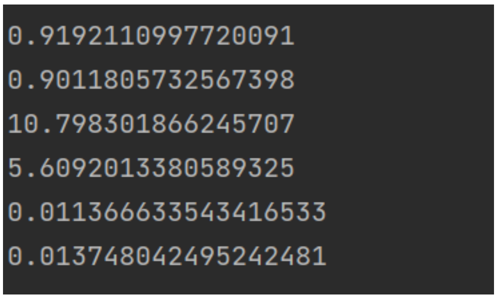


```
#Therefore, the variables predicted by Lasso in the above bar chart as significant vari
alpha = 3
ridge2 = Ridge(alpha=alpha)
ridge2.fit(X_train, y_train)
# Lets calculate some metrics such as R2 score, RSS and RMSE
y_pred_train = ridge2.predict(X_train)
y_pred_test = ridge2.predict(X_test)
metric2 = []
r2_train_lr = r2_score(y_train, y_pred_train)
print(r2_train_lr)
metric2.append(r2_train_lr)
r2_test_lr = r2_score(y_test, y_pred_test)
print(r2_test_lr)
metric2.append(r2_test_lr)
rss1_lr = np.sum(np.square(y_train - y_pred_train))
print(rss1_lr)
metric2.append(rss1_lr)
rss2_lr = np.sum(np.square(y_test - y_pred_test))
print(rss2_lr)
metric2.append(rss2_lr)
mse_train_lr = mean_squared_error(y_train, y_pred_train)
```



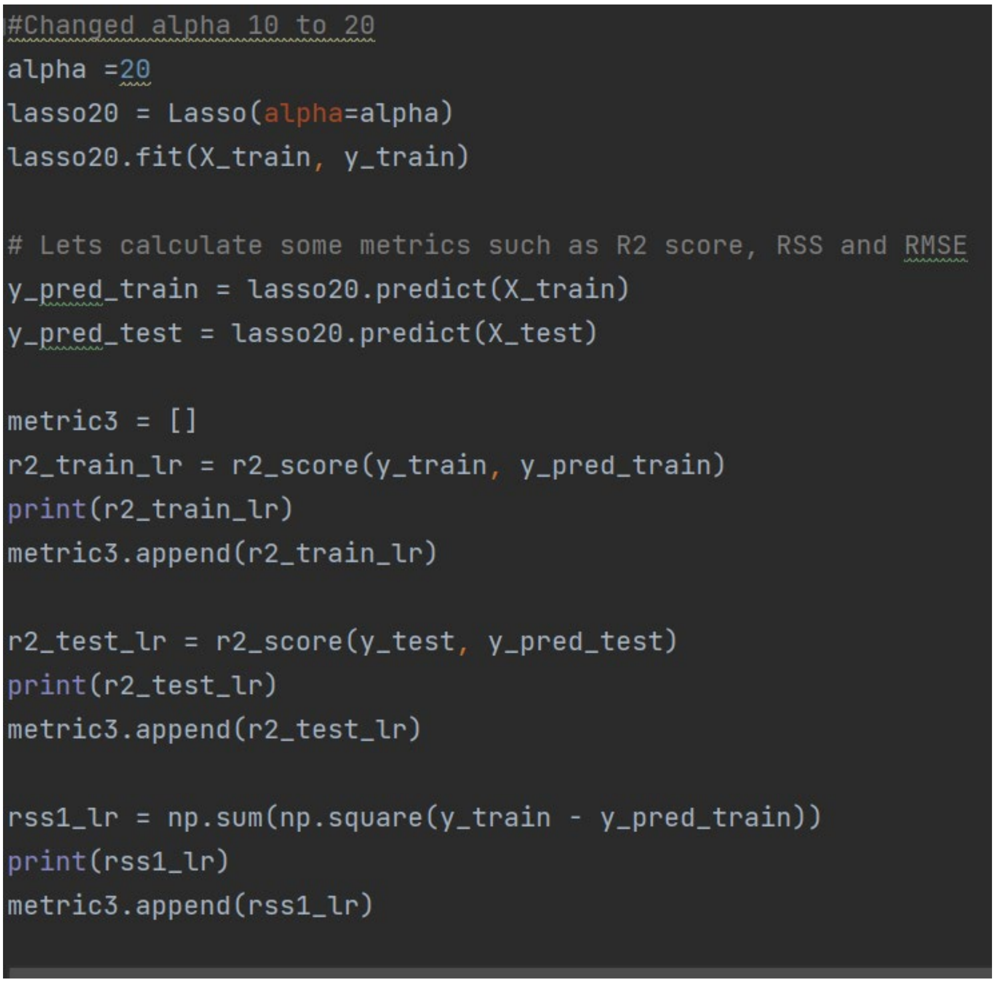
```
print(mse_train_lr)
metric2.append(mse_train_lr**0.5)
mse_test_lr = mean_squared_error(y_test, y_pred_test)
print(mse_test_lr)
metric2.append(mse_test_lr**0.5)
```

Result

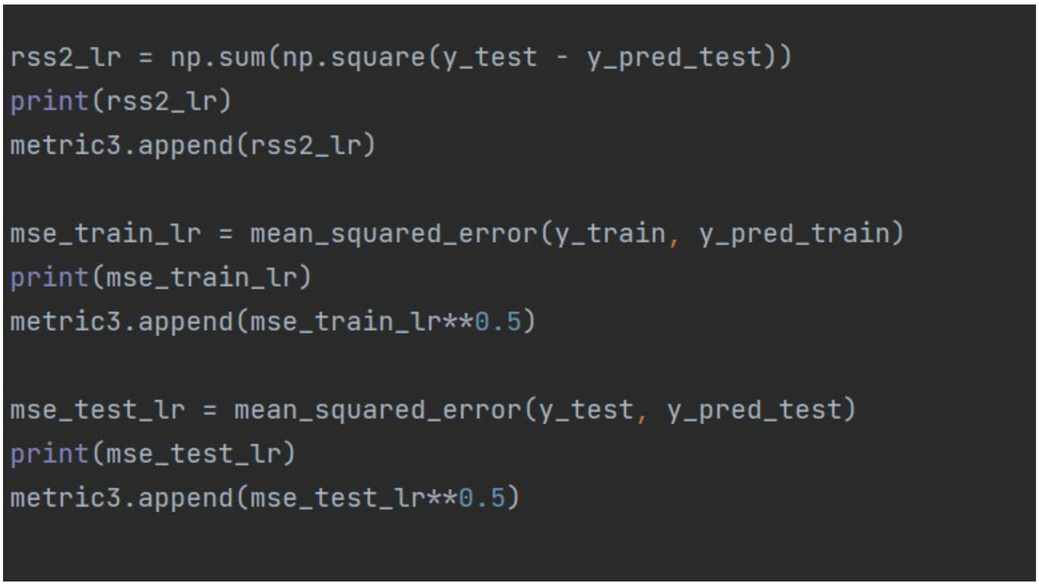


```
0.9192110997720091
0.9011805732567398
10.798301866245707
5.6092013380589325
0.011366633543416533
0.013748042495242481
```

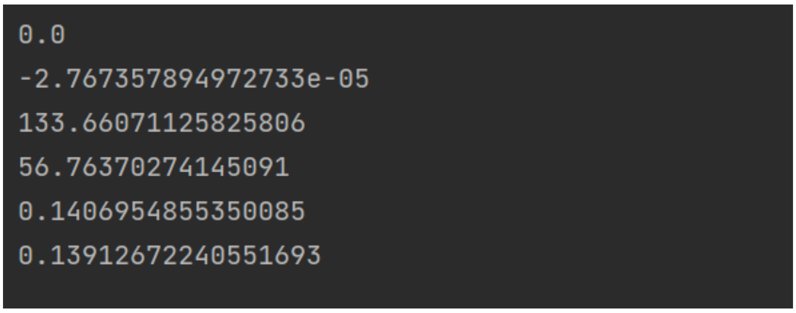
R2score on training data is decreased but it is increased on testing data



```python
#Changed alpha 10 to 20
alpha =20
lasso20 = Lasso(alpha=alpha)
lasso20.fit(X_train, y_train)

# Lets calculate some metrics such as R2 score, RSS and RMSE
y_pred_train = lasso20.predict(X_train)
y_pred_test = lasso20.predict(X_test)

metric3 = []
r2_train_lr = r2_score(y_train, y_pred_train)
print(r2_train_lr)
metric3.append(r2_train_lr)

r2_test_lr = r2_score(y_test, y_pred_test)
print(r2_test_lr)
metric3.append(r2_test_lr)

rss1_lr = np.sum(np.square(y_train - y_pred_train))
print(rss1_lr)
metric3.append(rss1_lr)
```

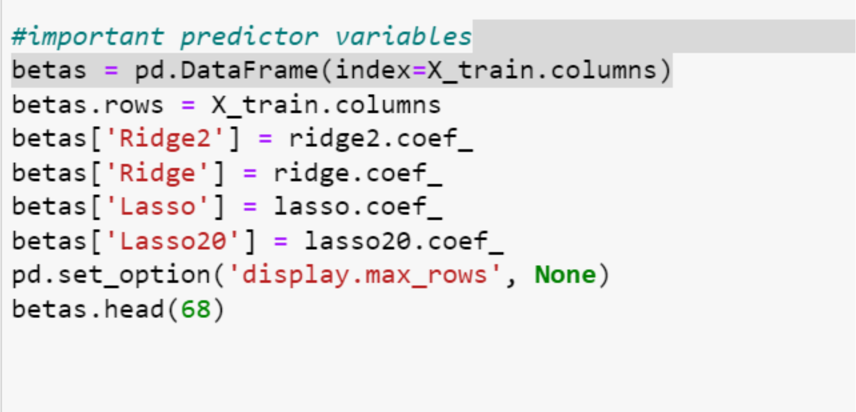


```
rss2_lr = np.sum(np.square(y_test - y_pred_test))
print(rss2_lr)
metric3.append(rss2_lr)

mse_train_lr = mean_squared_error(y_train, y_pred_train)
print(mse_train_lr)
metric3.append(mse_train_lr**0.5)

mse_test_lr = mean_squared_error(y_test, y_pred_test)
print(mse_test_lr)
metric3.append(mse_test_lr**0.5)
```

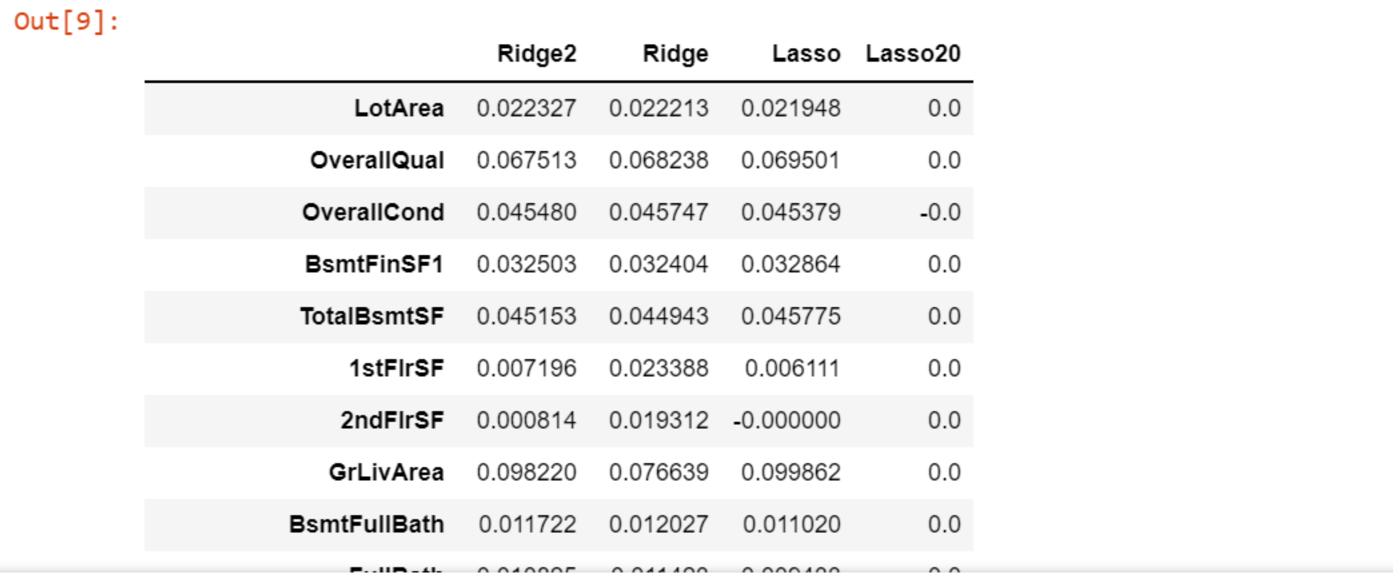


```
0.0
-2.767357894972733e-05
133.66071125825806
56.76370274145091
0.14069548553500085
0.13912672240551693
```

R2score of training data is decreased and it has increased on testing data



```
#important predictor variables
betas = pd.DataFrame(index=X_train.columns)
betas.rows = X_train.columns
betas['Ridge2'] = ridge2.coef_
betas['Ridge'] = ridge.coef_
betas['Lasso'] = lasso.coef_
betas['Lasso20'] = lasso20.coef_
pd.set_option('display.max_rows', None)
betas.head(68)
```



Out[9]:

	Ridge2	Ridge	Lasso	Lasso20
LotArea	0.022327	0.022213	0.021948	0.0
OverallQual	0.067513	0.068238	0.069501	0.0
OverallCond	0.045480	0.045747	0.045379	-0.0
BsmtFinSF1	0.032503	0.032404	0.032864	0.0
TotalBsmtSF	0.045153	0.044943	0.045775	0.0
1stFlrSF	0.007196	0.023388	0.006111	0.0
2ndFlrSF	0.000814	0.019312	-0.000000	0.0
GrLivArea	0.098220	0.076639	0.099862	0.0
BsmtFullBath	0.011722	0.012027	0.011020	0.0

LotArea :::::::::::::Lot size in square feet

OverallQual:::::::::::::Rates the overall material and finish of the house

OverallCond:::::::::::::Rates the overall condition of the house

YearBuilt::::::::::::Original construction date

BsmtFinSF1:::::::::::Type 1 finished square feet

TotalBsmtSF::::::::::::Total square feet of basement area

GrLivArea:::::::::::Above grade (ground) living area square feet

TotRmsAbvGrd:::::::::Total rooms above grade (does not include bathrooms)

Street_Pave::::::::::Pave road access to property

RoofMatl_Metal::::::::Roof material_Metal

Predictors are same but the coefficent of these predictor has changed

Q2:

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?

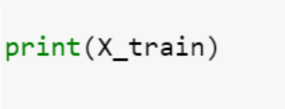
Ans: The r2_score of lasso has slightly more value than lasso for the test dataset so we will choose lasso regression to solve this problem.

Q3:

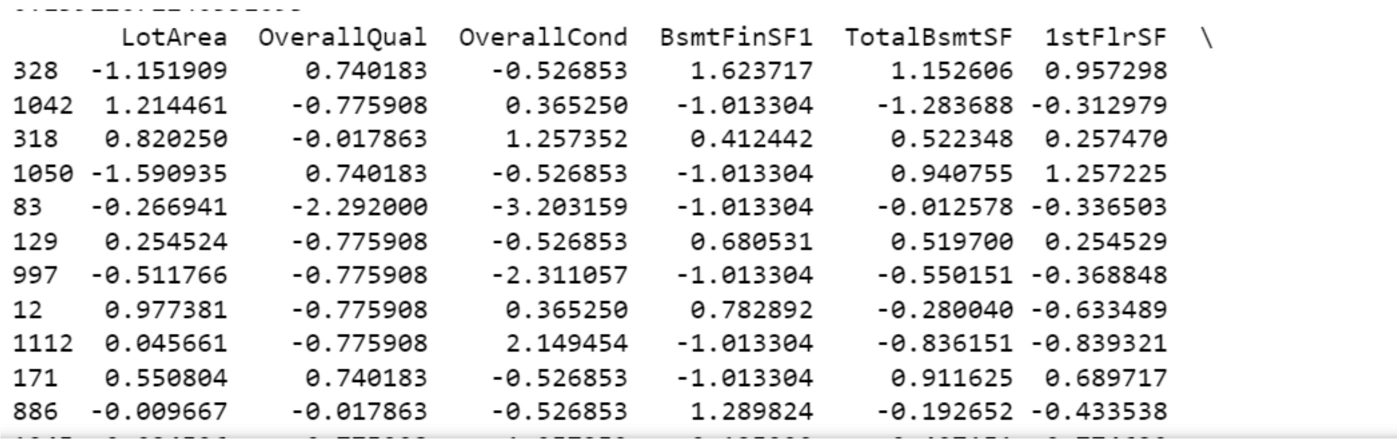
Q3:

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 model excluding the five most important predictor variables. Which are the five most important predictor variables now?

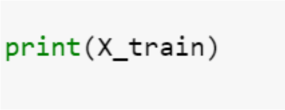
Ans:



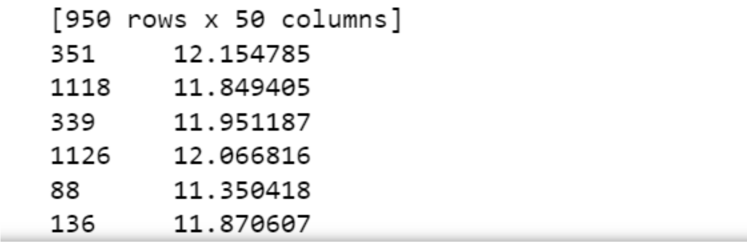
```
print(X_train)
```



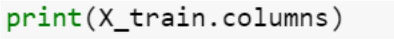
```
       LotArea  OverallQual  OverallCond  BsmtFinSF1  TotalBsmtSF  1stFlrSF  \
328  -1.151909     0.740183    -0.526853    1.623717     1.152606  0.957298
1042  1.214461    -0.775908     0.365250   -1.013304    -1.283688 -0.312979
318   0.820250    -0.017863     1.257352    0.412442     0.522348  0.257470
1050 -1.590935     0.740183    -0.526853   -1.013304     0.940755  1.257225
83   -0.266941    -2.292000    -3.203159   -1.013304    -0.012578 -0.336503
129   0.254524    -0.775908    -0.526853    0.680531     0.519700  0.254529
997  -0.511766    -0.775908    -2.311057   -1.013304    -0.550151 -0.368848
12    0.977381    -0.775908     0.365250    0.782892    -0.280040 -0.633489
1112  0.045661    -0.775908     2.149454   -1.013304    -0.836151 -0.839321
171   0.550804     0.740183    -0.526853   -1.013304     0.911625  0.689717
886  -0.009667    -0.017863    -0.526853    1.289824    -0.192652 -0.433538
```



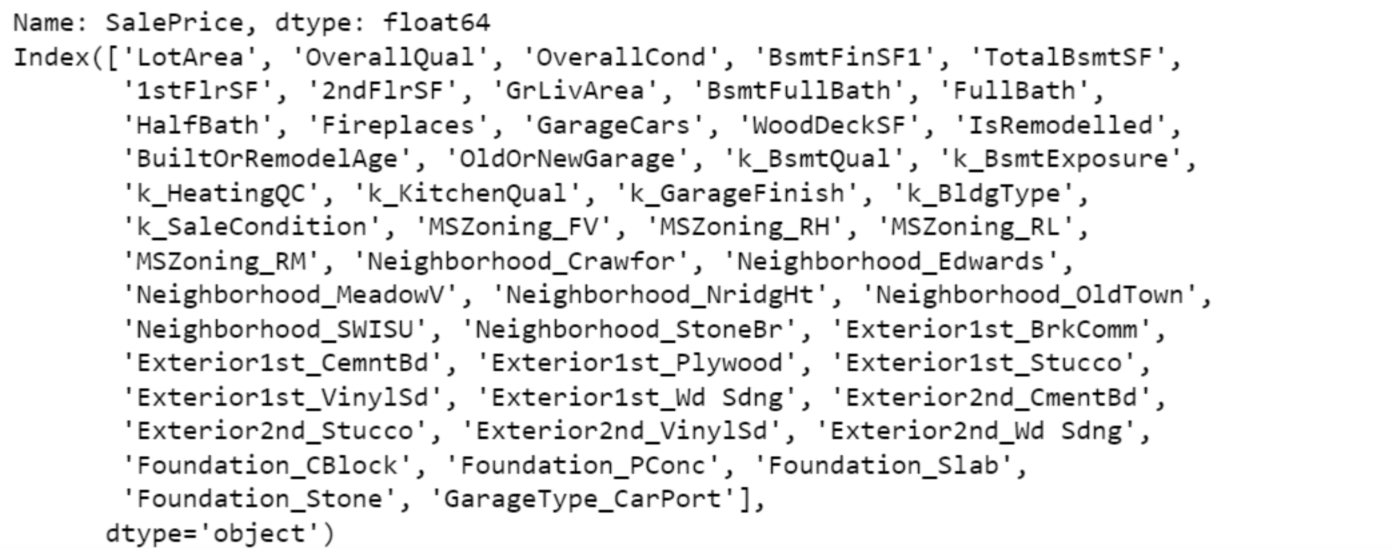
```
print(X_train)
```



```
[950 rows x 50 columns]
351     12.154785
1118    11.849405
339     11.951187
1126    12.066816
88      11.350418
136     11.870607
```

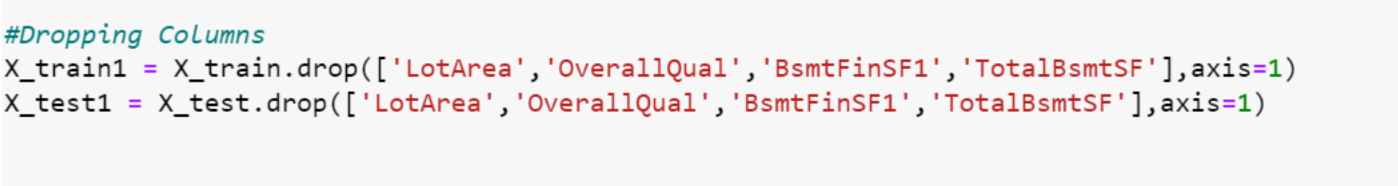


```
print(X_train.columns)
```

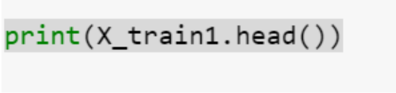


```
Name: SalePrice, dtype: float64
Index(['LotArea', 'OverallQual', 'OverallCond', 'BsmtFinSF1', 'TotalBsmtSF',
       '1stFlrSF', '2ndFlrSF', 'GrLivArea', 'BsmtFullBath', 'FullBath',
       'HalfBath', 'Fireplaces', 'GarageCars', 'WoodDeckSF', 'IsRemodelled',
       'BuiltOrRemodelAge', 'OldOrNewGarage', 'k_BsmtQual', 'k_BsmtExposure',
       'k_HeatingQC', 'k_KitchenQual', 'k_GarageFinish', 'k_BldgType',
       'k_SaleCondition', 'MSZoning_FV', 'MSZoning_RH', 'MSZoning_RL',
       'MSZoning_RM', 'Neighborhood_Crawfor', 'Neighborhood_Edwards',
       'Neighborhood_MeadowV', 'Neighborhood_NridgHt', 'Neighborhood_OldTown',
       'Neighborhood_SWISU', 'Neighborhood_StoneBr', 'Exterior1st_BrkComm',
       'Exterior1st_CemntBd', 'Exterior1st_Plywood', 'Exterior1st_Stucco',
       'Exterior1st_VinylSd', 'Exterior1st_Wd Sdng', 'Exterior2nd_CmentBd',
       'Exterior2nd_Stucco', 'Exterior2nd_VinylSd', 'Exterior2nd_Wd Sdng',
       'Foundation_CBlock', 'Foundation_PConc', 'Foundation_Slab',
       'Foundation_Stone', 'GarageType_CarPort'],
      dtype='object')
```

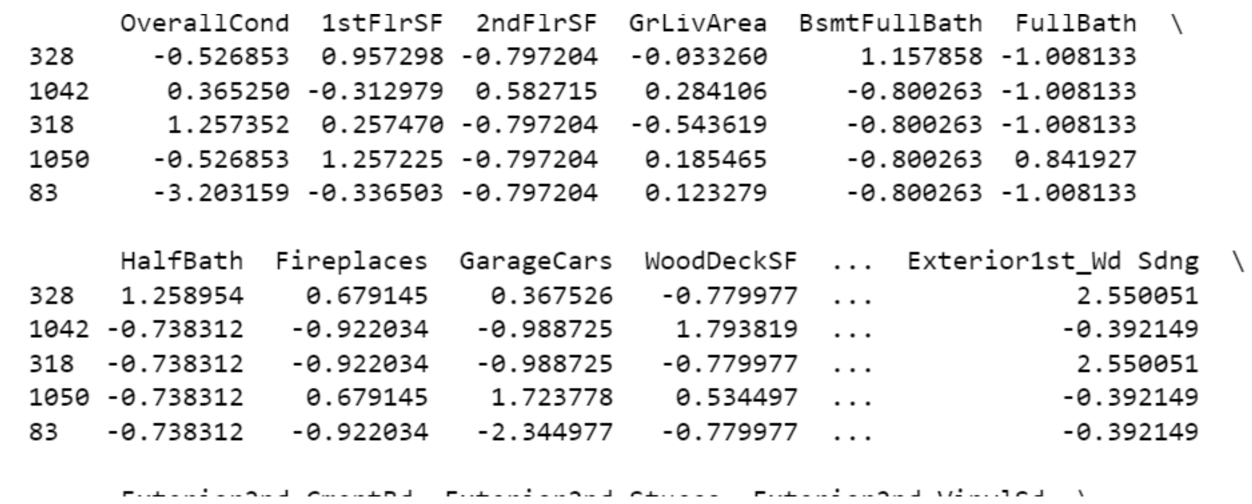
LotArea,OverallQual,YearBuilt,BsmtFinSF1,TotalBsmtSF are the top 5 important predictors



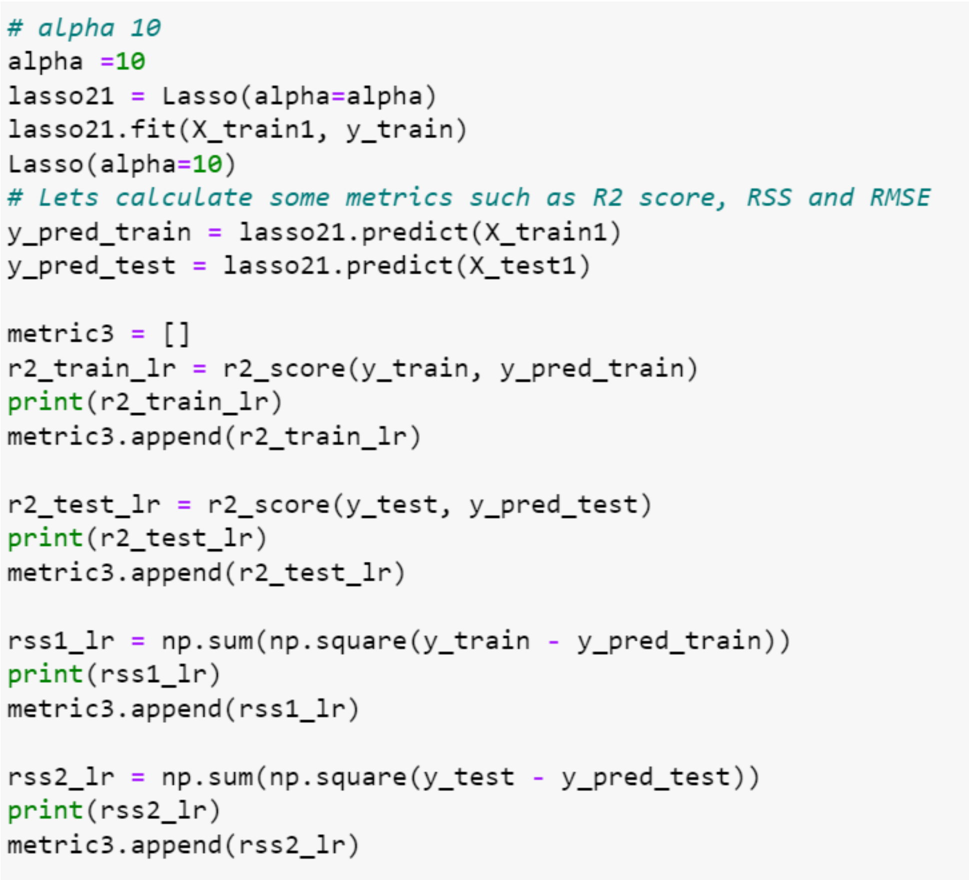
```
#Dropping Columns
X_train1 = X_train.drop(['LotArea','OverallQual','BsmtFinSF1','TotalBsmtSF'],axis=1)
X_test1 = X_test.drop(['LotArea','OverallQual','BsmtFinSF1','TotalBsmtSF'],axis=1)
```



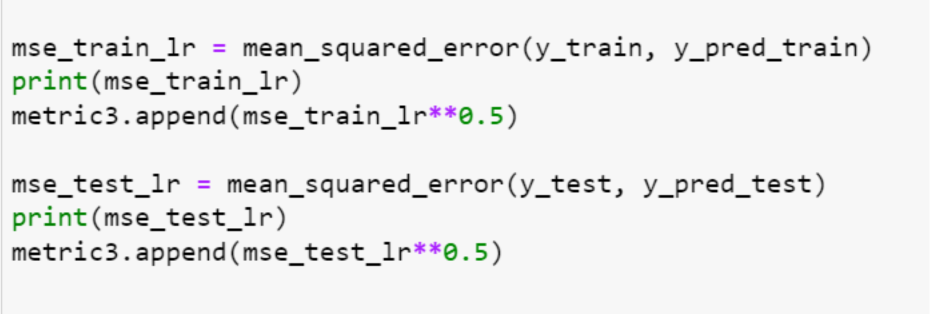
```
print(X_train1.head())
```



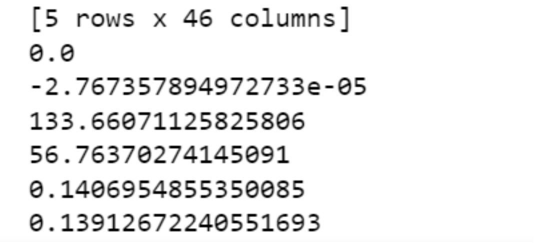
```
      OverallCond  1stFlrSF  2ndFlrSF  GrLivArea  BsmtFullBath  FullBath  \
328     -0.526853  0.957298 -0.797204  -0.033260      1.157858 -1.008133
1042     0.365250 -0.312979  0.582715   0.284106     -0.800263 -1.008133
318      1.257352  0.257470 -0.797204  -0.543619     -0.800263 -1.008133
1050    -0.526853  1.257225 -0.797204   0.185465     -0.800263  0.841927
83      -3.203159 -0.336503 -0.797204   0.123279     -0.800263 -1.008133

      HalfBath  Fireplaces  GarageCars  WoodDeckSF  ...  Exterior1st_Wd Sdng  \
328   1.258954    0.679145    0.367526   -0.779977  ...             2.550051
1042 -0.738312   -0.922034   -0.988725    1.793819  ...            -0.392149
318  -0.738312   -0.922034   -0.988725   -0.779977  ...             2.550051
1050 -0.738312    0.679145    1.723778    0.534497  ...            -0.392149
83   -0.738312   -0.922034   -2.344977   -0.779977  ...            -0.392149

      Exterior2nd_CmentBd  Exterior2nd_Stucco  Exterior2nd_VinylSd  \
```



```
# alpha 10
alpha =10
lasso21 = Lasso(alpha=alpha)
lasso21.fit(X_train1, y_train)
Lasso(alpha=10)
# Lets calculate some metrics such as R2 score, RSS and RMSE
y_pred_train = lasso21.predict(X_train1)
y_pred_test = lasso21.predict(X_test1)

metric3 = []
r2_train_lr = r2_score(y_train, y_pred_train)
print(r2_train_lr)
metric3.append(r2_train_lr)

r2_test_lr = r2_score(y_test, y_pred_test)
print(r2_test_lr)
metric3.append(r2_test_lr)

rss1_lr = np.sum(np.square(y_train - y_pred_train))
print(rss1_lr)
metric3.append(rss1_lr)

rss2_lr = np.sum(np.square(y_test - y_pred_test))
print(rss2_lr)
metric3.append(rss2_lr)
```

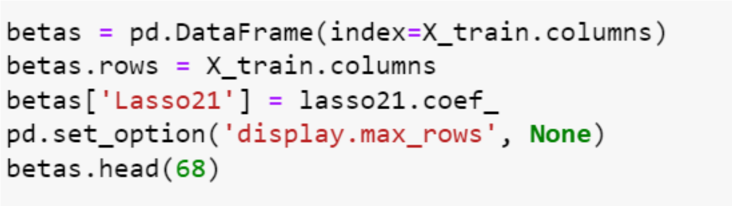


```
mse_train_lr = mean_squared_error(y_train, y_pred_train)
print(mse_train_lr)
metric3.append(mse_train_lr**0.5)

mse_test_lr = mean_squared_error(y_test, y_pred_test)
print(mse_test_lr)
metric3.append(mse_test_lr**0.5)
```



```
[5 rows x 46 columns]
0.0
-2.767357894972733e-05
133.66071125825806
56.76370274145091
0.1406954855350085
0.13912672240551693
```

R2score of training and testing data has decreased



```
betas = pd.DataFrame(index=X_train.columns)
betas.rows = X_train.columns
betas['Lasso21'] = lasso21.coef_
pd.set_option('display.max_rows', None)
betas.head(68)
```

11stFlrSF---------→>>>>>>>>First Floor square feet

GrLivArea---------→>>>>>>>>>>>Above grade (ground) living area square feet

Street_Pave-------→>>>>>>>>>>>Pave road access to property

RoofMatl_Metal---→>>>>>>>-Roof material_Metal

RoofStyle_Shed----→>>>>>>>Type of roof(Shed)

Q4:

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?

Ans:

1.The model should be generalized so that the test accuracy is not lesser than the training score.

2.The model should be accurate for datasets other than the ones which were used during training.

3.More importance should not given to the outliers so that the accuracy predicted by the model is high.

4.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.

5.Those outliers which it does not make sense to keep must be removed from the dataset. If the model is not robust,

6.It cannot be trusted for predictive analysis.