Advanced Regression

ASSIGNMENT- FINAL SUBMISSION

[GitHub - subrahmanyeswaraokrv/Advanced-Regression-Assignment]

KRV SUBRAHMANYESWARAO | UPGRAD AND IIITB AI&ML | OCT-2023

QUESTION 1

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 of alpha for Ridge

- Ridge - 0.3
- Lasso - 0.0001

Double the values of alpha

- Ridge - 0.6
- Lasso - 0.0002

Lets analyze the model with these alpha values

In [122]:

```
# Ridge regression
ridge = Ridge(alpha=0.6)
ridge.fit(X_train, y_train)
print(ridge.coef_)
[ 0.09953354  0.05619525  0.07278613  0.13441889  0.05627912  0.29137694
  0.1774726   0.24827173  0.13289595 -0.05023351  0.0615185   0.06268812
  0.06224036 -0.09935608 -0.08113491 -0.05822534 -0.14280582  0.08082369
 -0.12792776 -0.12003386 -0.12095389 -0.09473805 -0.26225613  0.031477
 -0.04160321 -0.0376924   0.01143726 -0.04215807 -0.08500809  0.07335669
 -0.04160321  0.06271915  0.05740394  0.0463302   0.08707393 -0.11467856
  0.09710922  0.07150643  0.04693881  0.09908806  0.0642639   0.06318014
  0.04693881  0.06379198 -0.06213756 -0.05934919  0.0800697  -0.11101574
 -0.12232067  0.09177435]
```

In [124]:

```
#Printing the values of R2, RSS, MSE for train and test

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

metric4=[]
r2_train_lr = r2_score(y_train,y_pred_train )
print("r2_train:")
print(r2_train_lr)
metric4.append(r2_train_lr)

r2_test_lr = r2_score(y_test, y_pred_test)
print("r2_test:")
print(r2_test_lr)
metric4.append(r2_test_lr)
```

```
rss1_lr = np.sum(np.square(y_train-y_pred_train))
print("RSS_train:")
print(rss1_lr)
metric4.append(rss1_lr)

rss2_lr = np.sum(np.square(y_test - y_pred_test))
print("RSS_test:")
print(rss2_lr)
metric4.append(rss2_lr)

mse_train_lr = mean_squared_error(y_train,y_pred_train)
print("MSE_train:")
print(mse_train_lr)
metric4.append(mse_train_lr)

mse_test_lr = mean_squared_error(y_test , y_pred_test)
print("MSE_test:")
print(mse_test_lr)
metric4.append(mse_test_lr)
r2_train:
0.9063878384815918
r2_test:
0.8706105658030794
RSS_train:
13.010979044960635
RSS_test:
8.553367497353891
MSE_train:
0.013413380458722304
MSE_test:
0.020560979560946855
```

In [125]:

```
ridge_df = pd.DataFrame({'Features':X_train.columns,
'Coefficient':ridge.coef_.round(4)})
ridge_df.reset_index(drop=True, inplace=True)
ridge_df
```

Out[125]:

	Features	Coefficient
0	OverallQual	0.0995
1	OverallCond	0.0562

	Features	Coefficient
2	TotalBsmtSF	0.0728
3	GrLivArea	0.1344
4	GarageCars	0.0563
5	MSZoning_FV	0.2914
6	MSZoning_RH	0.1775
7	MSZoning_RL	0.2483
8	MSZoning_RM	0.1329
9	LotConfig_FR3	-0.0502
10	Condition1_Norm	0.0615
11	Condition1_RRAn	0.0627
12	Condition1_RRNn	0.0622
13	BldgType_Duplex	-0.0994
14	BldgType_Twnhs	-0.0811
15	HouseStyle_1.5Unf	-0.0582

	Features	Coefficient
16	HouseStyle_2.5Unf	-0.1428
17	HouseStyle_SFoyer	0.0808
18	RoofStyle_Gable	-0.1279
19	RoofStyle_Gambrel	-0.1200
20	RoofStyle_Hip	-0.1210
21	RoofStyle_Mansard	-0.0947
22	Exterior1st_BrkComm	-0.2623
23	Exterior1st_BrkFace	0.0315
24	Exterior1st_CBlock	-0.0416
25	Exterior1st_CemntBd	-0.0377
26	Exterior1st_ImStucc	0.0114
27	Exterior1st_Stone	-0.0422
28	Exterior1st_Wd Sdng	-0.0850
29	Exterior2nd_AsphShn	0.0734

	Features	Coefficient
30	Exterior2nd_CBlock	-0.0416
31	Exterior2nd_CmentBd	0.0627
32	Exterior2nd_Other	0.0574
33	Exterior2nd_Wd Sdng	0.0463
34	Foundation_PConc	0.0871
35	Foundation_Wood	-0.1147
36	SaleType_CWD	0.0971
37	SaleType_Con	0.0715
38	SaleType_New	0.0469
39	SaleType_Oth	0.0991
40	SaleCondition_AdjLand	0.0643
41	SaleCondition_Normal	0.0632
42	SaleCondition_Partial	0.0469
43	Neighborhood_Crawfor	0.0638

	Features	Coefficient
44	Neighborhood_Edwards	-0.0621
45	Neighborhood_MeadowV	-0.0593
46	Neighborhood_NridgHt	0.0801
47	Neighborhood_OldTown	-0.1110
48	Neighborhood_SWISU	-0.1223
49	Neighborhood_Veenker	0.0918

In [126]:

```
#feature reduction - taking top 10 features from ridge
model_param = list(ridge.coef_)
model_param.insert(0,ridge.intercept_)
cols =  X_train[ridge_df.Features]

ridge_coef = pd.DataFrame(list(zip(cols,model_param)))
ridge_coef.columns = ['Featuere','Coef']
ridge_coef.sort_values(by='Coef',ascending=False).head(10)
```

Out[126]:

	Featuere	Coef
0	OverallQual	11.770827
6	MSZoning_RH	0.291377
8	MSZoning_RM	0.248272

	Featuere	Coef
7	MSZoning_RL	0.177473
4	GarageCars	0.134419
9	LotConfig_FR3	0.132896
1	OverallCond	0.099534
40	SaleCondition_AdjLand	0.099088
37	SaleType_Con	0.097109
35	Foundation_Wood	0.087074

- Here we have got Zoning, Condition1, Saletype condition, Exterios

In [127]:

```
# Lasso Regression:

lm = Lasso(alpha=0.002)
lm.fit(X_train,y_train)

#r2 train
y_train_pred = lm.predict(X_train)
print(r2_score(y_true=y_train,y_pred=y_train_pred))

#r2 test
y_test_pred  = lm.predict(X_test)
print(r2_score(y_true=y_test,y_pred=y_test_pred))
0.8879357191712136
0.8605144673838798
```

In [128]:

```
# prnitng R2, RSS, MSE of test train when we double the alpha value for Lasso
y_pred_train = lm.predict(X_train)
y_pred_test = lm.predict(X_test)
```

```
metric5=[]
r2_train_lr = r2_score(y_train,y_pred_train )
print("r2_train:")
print(r2_train_lr)
metric5.append(r2_train_lr)

r2_test_lr = r2_score(y_test, y_pred_test)
print("r2_test:")
print(r2_test_lr)
metric5.append(r2_test_lr)

rss1_lr = np.sum(np.square(y_train-y_pred_train))
print("RSS_train:")
print(rss1_lr)
metric5.append(rss1_lr)

rss2_lr = np.sum(np.square(y_test - y_pred_test))
print("RSS_test:")
print(rss2_lr)
metric5.append(rss2_lr)

mse_train_lr = mean_squared_error(y_train,y_pred_train)
print("MSE_train:")
print(mse_train_lr)
metric5.append(mse_train_lr)

mse_test_lr = mean_squared_error(y_test , y_pred_test)
print("MSE_test:")
print(mse_test_lr)
metric5.append(mse_test_lr)
```

```
r2_train:
0.8879357191712136
r2_test:
0.8605144673838798
RSS_train:
15.575604557162212
RSS_test:
9.220776243708254
MSE_train:
0.01605732428573424
MSE_test:
0.022165327508914073
```

In [129]:

```
# Put the shortlisted Features and coefficienst in a dataframe

lasso_df = pd.DataFrame({'Features':X_train.columns,
'Coefficient':lm.coef_.round(4)})
lasso_df = lasso_df[lasso_df['Coefficient'] != 0.00]
lasso_df.reset_index(drop=True, inplace=True)
```

```
lasso_df
```

Out[129]:

	Features	Coefficient
0	OverallQual	0.1193
1	OverallCond	0.0575
2	TotalBsmtSF	0.0772
3	GrLivArea	0.1236
4	GarageCars	0.0616
5	MSZoning_FV	0.0738
6	MSZoning_RL	0.0851
7	MSZoning_RM	-0.0423
8	Condition1_Norm	0.0378
9	BldgType_Duplex	-0.0150
10	Exterior1st_Wd Sdng	-0.0317
11	Foundation_PConc	0.0842
12	SaleType_New	0.0478

	Features	Coefficient
13	SaleCondition_Normal	0.0268
14	SaleCondition_Partial	0.0031
15	Neighborhood_Crawfor	0.0111
16	Neighborhood_Edwards	-0.0275
17	Neighborhood_NridgHt	0.0136
18	Neighborhood_OldTown	-0.0770

- here we have got 19 features

In [132]:

```
# Do an RFE to minimise the features to 15

X_train_lasso = X_train[lasso_df.Features]

lm = LinearRegression()
lm.fit(X_train_lasso, y_train)

# running RFE

rfe = RFE(lm, n_features_to_select=15)
rfe = rfe.fit(X_train_lasso, y_train)
```

In [133]:

```
# Method to get the coefficient values
lasso_coeff_dict = dict(pd.Series(lm.coef_, index = X_train_lasso.columns))

# Assign top 10 features to a temp dataframe for further display in the bar
plot
df = pd.DataFrame(list(zip( X_train_lasso.columns, rfe.support_,
rfe.ranking_)), columns=['Features', 'rfe_support', 'rfe_ranking'])
df = df.loc[df['rfe_support'] == True]
df.reset_index(drop=True, inplace=True)
```

```
df['Coefficient'] = df['Features'].apply(find)
df = df.sort_values(by=['Coefficient'], ascending=False)
df = df.head(10)
df
```

Out[133]:

	Features	rfe_support	rfe_ranking	Coefficient
4	MSZoning_FV	True	1	0.222373
5	MSZoning_RL	True	1	0.183704
2	GrLivArea	True	1	0.129387
0	OverallQual	True	1	0.102871
8	Foundation_PConc	True	1	0.084897
1	TotalBsmtSF	True	1	0.077818
11	Neighborhood_Crawfor	True	1	0.074205
13	Neighborhood_NridgHt	True	1	0.065437
9	SaleCondition_Normal	True	1	0.059463
3	GarageCars	True	1	0.059166

COMPARING THE RIDGE AND LASSO AFTER DOUBLE THE VLAUES OF ALPHA

In [134]:

```
#Comparing results of Ridge and Lasso
```

```
resultTable = {'Metric':["R2 Score Train", "R2Score Test", "RSS Train", "RSS
Test", "MSE Train", "MSE Test"],
              'Ridge regression':metric4}
rg_metric = pd.DataFrame(resultTable, columns=["Metric", 'Ridge regression'])
ls_metric = pd.Series(metric5, name ='Lasso regression')
final = pd.concat([rg_metric,ls_metric],axis=1)
final
```

Out[134]:

	Metric	Ridge regression	Lasso regression
0	R2 Score Train	0.906388	0.887936
1	R2Score Test	0.870611	0.860514
2	RSS Train	13.010979	15.575605
3	RSS Test	8.553367	9.220776
4	MSE Train	0.013413	0.016057
5	MSE Test	0.020561	0.022165

Here Lasso given the very close result of R2 score for both test and train. The most important feature after double the value of alpha is

- MSZoning_FV
- MSZoning_RL
- GrLivArea
- OverallQual
- TotalBsmtSF
- Neighborhood_Crawfor
- Foundation_PConc
- Neighborhood_NridgHt
- SaleCondition_Normal
- GarageCars

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: Based on the alpha/Lambda values I have got, Ridge regression does not zero any of the co efficient, Lasso zeroed one or two coefficients in the selected features, Lasso is better option and it also helps in the some of the feature elimination.

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 [136]:

```
houseLasso = houseNew
houseLasso = houseLasso.drop(["MSZoning_FV", "GrLivArea", "MSZoning_RL",
"OverallQual", "Foundation_PConc"], axis=1)
```

In [137]:

```
df_train, df_test = train_test_split(houseLasso, train_size=0.7,test_size =
0.3, random_state=100)
```

In [138]:

```
num_col =['MSSubClass','LotArea','OverallCond',
          'TotalBsmtSF','1stFlrSF','2ndFlrSF',
         'BsmtFullBath','FullBath','HalfBath','BedroomAbvGr',
         'Fireplaces','GarageCars',
          'WoodDeckSF','OpenPorchSF','EnclosedPorch',
        ]

scaler = StandardScaler()
df_train[num_col] = scaler.fit_transform(df_train[num_col])
df_test[num_col] = scaler.transform(df_test[num_col])
```

In [139]:

```
X_trainLS = df_train
y_trainLS = df_train.pop('SalePrice')

X_testLS = df_test
y_testLS = df_test.pop('SalePrice')
```

In [141]:

```
# Linear regression
lm = LinearRegression()
lm.fit(X_train, y_train)

# running RFE
```

```python
rfe = RFE(lm, n_features_to_select=50)
rfe = rfe.fit(X_trainLS, y_trainLS)
```

In [142]:

```python
# Assign the columns selected by RFE to cols

col = X_trainLS.columns[rfe.support_]

# assign the 50 features selected using RFE to a dataframe and view them

temp1 = pd.DataFrame(list(zip(X_trainLS.columns,rfe.support_,rfe.ranking_)),
columns=['Variable', 'rfe_support', 'rfe_ranking'])
temp1 = temp1.loc[temp1['rfe_support'] == True]
temp1.reset_index(drop=True, inplace=True)

temp1
```

Out[142]:

	Variable	rfe_support	rfe_ranking
0	TotalBsmtSF	True	1
1	1stFlrSF	True	1
2	2ndFlrSF	True	1
3	KitchenQual	True	1
4	BldgType_Duplex	True	1
5	HouseStyle_1.5Unf	True	1
6	HouseStyle_1Story	True	1
7	HouseStyle_2.5Fin	True	1

	Variable	rfe_support	rfe_ranking
8	RoofStyle_Gable	True	1
9	RoofStyle_Gambrel	True	1
10	RoofStyle_Hip	True	1
11	RoofStyle_Mansard	True	1
12	Exterior1st_BrkComm	True	1
13	Exterior1st_BrkFace	True	1
14	Exterior1st_CemntBd	True	1
15	Exterior2nd_AsphShn	True	1
16	Exterior2nd_CmentBd	True	1
17	Exterior2nd_HdBoard	True	1
18	Exterior2nd_ImStucc	True	1
19	Exterior2nd_MetalSd	True	1
20	Exterior2nd_Other	True	1
21	Exterior2nd_Plywood	True	1

	Variable	rfe_support	rfe_ranking
22	Exterior2nd_Stucco	True	1
23	Exterior2nd_VinylSd	True	1
24	Exterior2nd_Wd Sdng	True	1
25	GarageType_Attchd	True	1
26	GarageType_Basment	True	1
27	GarageType_BuiltIn	True	1
28	GarageType_Detchd	True	1
29	SaleType_CWD	True	1
30	SaleType_Con	True	1
31	SaleType_ConLI	True	1
32	SaleType_New	True	1
33	SaleType_Oth	True	1
34	SaleCondition_AdjLand	True	1
35	SaleCondition_Normal	True	1

	Variable	rfe_support	rfe_ranking
36	SaleCondition_Partial	True	1
37	Neighborhood_Blueste	True	1
38	Neighborhood_BrDale	True	1
39	Neighborhood_BrkSide	True	1
40	Neighborhood_ClearCr	True	1
41	Neighborhood_Edwards	True	1
42	Neighborhood_IDOTRR	True	1
43	Neighborhood_MeadowV	True	1
44	Neighborhood_NAmes	True	1
45	Neighborhood_NoRidge	True	1
46	Neighborhood_NridgHt	True	1
47	Neighborhood_OldTown	True	1
48	Neighborhood_SWISU	True	1
49	Neighborhood_Sawyer	True	1

In [143]:

```
# Assign the 50 columns to X_train_rfe

X_trainLS_rfe = X_trainLS[col]
# Associate the new 50 columns to X_train and X_test for further analysis

X_trainLS = X_trainLS_rfe[X_trainLS_rfe.columns]
X_testLS =  X_testLS[X_trainLS.columns]
```

PERFORM LASSO TO NEW MODEL AFTER DROPPING THE FIVE IMP FEATURES

In [144]:

```
# Lasso Regression:

lm = Lasso(alpha=0.001)
lm.fit(X_trainLS,y_trainLS)

y_train_predLS = lm.predict(X_trainLS)
print(r2_score(y_true=y_trainLS,y_pred=y_train_predLS))

y_test_predLS  = lm.predict(X_testLS)
print(r2_score(y_true=y_testLS,y_pred=y_test_predLS))
0.8508069159169048
0.831327471195756
```

In [145]:

```
#printing R2, RSS, MSE results
r2_train_lr = r2_score(y_trainLS ,y_train_predLS )
print(r2_train_lr)


r2_test_lr = r2_score(y_testLS, y_test_predLS)
print(r2_test_lr)


rss1_lr = np.sum(np.square(y_trainLS-y_train_predLS))
print(rss1_lr)


rss2_lr = np.sum(np.square(y_testLS - y_test_predLS))
print(rss2_lr)


mse_train_lr = mean_squared_error(y_trainLS,y_train_predLS)
print(mse_train_lr)


mse_test_lr = mean_squared_error(y_testLS , y_test_predLS)
print(mse_test_lr)
0.8508069159169048
```

```
0.831327471195756
20.736067399495816
11.150200435802233
0.021377389071645173
0.026803366432216907
```

In [146]:

```
model_param = list(lm.coef_)
model_param.insert(0,lm.intercept_)
cols = df_train.columns
cols.insert(0,'const')
lasso_coef = pd.DataFrame(list(zip(cols,model_param)))
lasso_coef.columns = ['Featuere','Coef']
lasso_coef.sort_values(by='Coef',ascending=False).head(10)
#(["MSZoning_FV", "GrLivArea", "MSZoning_RL", "OverallQual",
"Foundation_PConc"]#
```

Out[146]:

	Featuere	Coef
0	MSSubClass	11.576299
4	OverallCond	0.139090
3	LotShape	0.134275
2	LotArea	0.125079
26	MoSold	0.122311
28	BuiltOrRemodelAge	0.114759
1	LotFrontage	0.085939
36	LotConfig_FR2	0.067728

	Featuere	Coef
47	BldgType_2fmCon	0.063315
37	LotConfig_FR3	0.060795

After removing the five most important fetaure that we have got prior "MSZoning_FV", "GrLivArea", "MSZoning_RL", "OverallQual", "Foundation_PConc" I have got the other important fetaures to predict the sales price with Overall condition, Lot area, shape, Condition1, IsRemodeled.