Subjective Questions Solutions

Assignment-based Subjective Questions

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?**

Ans : The optimal value of alpha for ridge and lasso regression

Ridge Alpha 9

lasso Alpha 100

```
# Changing the alpha for ridge from alpha = 9 to alpha = 18

#Fitting Ridge model for alpha = 9 and printing coefficients which have been penalised
alpha = 18
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)
```

```
0.8847713615420342
0.8811984259075518
844636864756.3575
223025521468.16208
723148000.6475664
763786032.4252126
```

For alpha = 9

R2(Train) 0.8951172
R2(Test) 0.8887555

for alpha = 18

R2(Train) 0.8847

R2(Test) 0.8811

Both training and test R2 score got decreased but a very very small change. Not significant

```
# Lasso alpha from 100 to 200
alpha = 200
lasso2 = Lasso(alpha=alpha)
lasso2.fit(X_train, y_train)

# Lets calculate some metrics such as R2 score, RSS and RMSE
y_pred_train = lasso2.predict(X_train)
y_pred_test = lasso2.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)
```

```
0.8837527882732236
0.8827874180144964
852103103564.7473
220042515594.9878
729540328.3944755
753570258.8869445
```

For alpha = 100 R2(Train) 0.8995731 R2(Test) 0.891154

for alpha = 200 R2(Train) 0.88375 R2(Test) 0.88278

Both R2 training and test score got decreased.

```
#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['Lasso2'] = lasso2.coef_
pd.set_option('display.max_rows', None)
betas.head(68)
```

	Ridge2	Ridge	Lasso	Lasso2
mssub_class	-3138.910216	-3534.551486	-4848.146909	-3102.454648
lot_frontage	5475.035385	3584.447095	0.000000	0.000000
lot_area	18585.341279	20351.519556	22762.140297	21135.253768
mas_vnr_area	-5886.873814	-6569.290727	-4144.321920	-1746.105123
bsmt_fin_sf1	22673.087965	26220.868597	29029.413402	29154.912790
bsmt_unf_sf	6195.756657	6790.633321	2654.568174	0.000000
total_bsmt_sf	14434.211799	13163.918096	3984.054286	5150.985020
1st_flr_sf	27897.824541	32442.700062	42795.058807	36807.154576
2nd_flr_sf	8058.722767	7895.258131	6771.706515	0.000000
gr_liv_area	24827.807458	27984.534233	39240.613726	46265.665763
bedroom_abv_gr	5329.006894	3917.370301	0.000000	0.000000

Predictors are same but the coefficent of these predictor has changed

Predictors are same only the coefficients of these predictors have changed.

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 :

r2_score of lasso is slightly higher than lasso for the test dataset so we will choose lasso regression\to solve this problem . Lasso provides features selections as well . Better to use lasso with respect to that and to make the model more robust.

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?**

Ans :

```
X_train.columns
```

```
Index(['mssub_class', 'lot_frontage', 'lot_area', 'mas_vnr_area',
       'bsmt_fin_sf1', 'bsmt_unf_sf', 'total_bsmt_sf', '1st_flr_sf',
       '2nd_flr_sf', 'gr_liv_area',
       ...
       'bsmt_half_bath_1', 'bsmt_half_bath_2', 'full_bath_1', 'full_bath_2',
       'full_bath_3', 'half_bath_1', 'half_bath_2', 'kitchen_abv_gr_1',
       'kitchen_abv_gr_2', 'kitchen_abv_gr_3'],
      dtype='object', length=253)
```

X_train1 = X_train.drop(['overall_qual_10', 'overall_qual_9', 'full_bath_3', 'roof_matl_WdShngl', '1st_flr_sf'], axis=1)

X_test1 = X_test.drop(['overall_qual_10', 'overall_qual_9', 'full_bath_3', 'roof_matl_WdShngl', '1st_flr_sf'], axis=1)

```
lasso = Lasso()

# cross validation
model_cv = GridSearchCV(estimator = lasso,
                        param_grid = params,
                        scoring= 'neg_mean_absolute_error',
                        cv = folds,
                        return_train_score=True,
                        verbose = 1)

model_cv.fit(X_train1, y_train)
#Fitting Lasso model for alpha = 100 and printing coefficients which have been penalised

alpha = model_cv.best_params_['alpha']

lasso3 = Lasso(alpha=alpha)

lasso3.fit(X_train1, y_train)

# Lets calculate some metrics such as R2 score, RSS and RMSE

y_pred_train = lasso3.predict(X_train1)
y_pred_test = lasso3.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)
```

```python
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)

# lasso model parameters
model_parameters = list(lasso3.coef_ )
model_parameters.insert(0, lasso3.intercept_)
model_parameters = [round(x, 3) for x in model_parameters]
cols = X_train1.columns
cols = cols.insert(0, "constant")
list(zip(cols, model_parameters))

mod = list(zip(cols, model_parameters))
para = pd.DataFrame(mod)
para.columns = ['Variable', 'Coeff']
# sort the coefficients in ascending order
para = para.sort_values((['Coeff']), axis = 0, ascending = False)
# Chose variables whose coefficients are non-zero
pred = pd.DataFrame(para[(para['Coeff'] != 0)])
pred
```

```
Fitting 5 folds for each of 28 candidates, totalling 140 fits
0.8837098915832098
0.8678723343188257
852417540377.0416
248042517652.0287
729809537.9940425
849460676.8905092
```

9]:

	Variable	Coeff
0	constant	192036.208
48	neighborhood_NoRidge	49840.829
9	gr_liv_area	45395.315
11	tot_rms_abv_grd	38860.147
5	bsmt_fin_sf1	34558.836
55	neighborhood_StoneBr	30657.817

Now , the 5 most important predictors are :

1.neighborhood_NoRidge

2.gr_liv_area

3.tot_rms_abv_grd

4.bsmt_fin_sf1

5.neighborhood_StoneBr

Q.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?

Ans : The model should be generalized so that the test accuracy should not be much lesser compared to the training score. The model should be accurate for test datasets and so it should be generic in nature . Too much importance should not given to the outliers so that the accuracy predicted by the model is high. To ensure that this should not happen , the outliers analysis needs to be done and only those which are relevant to the dataset need to be retained. Those outliers which it does not make sense to keep must be handled or removed from the dataset.

If the model is not robust, It cannot be trusted for predictive analysis.