

## Subjective Questions – Advanced Regression

### Question 1

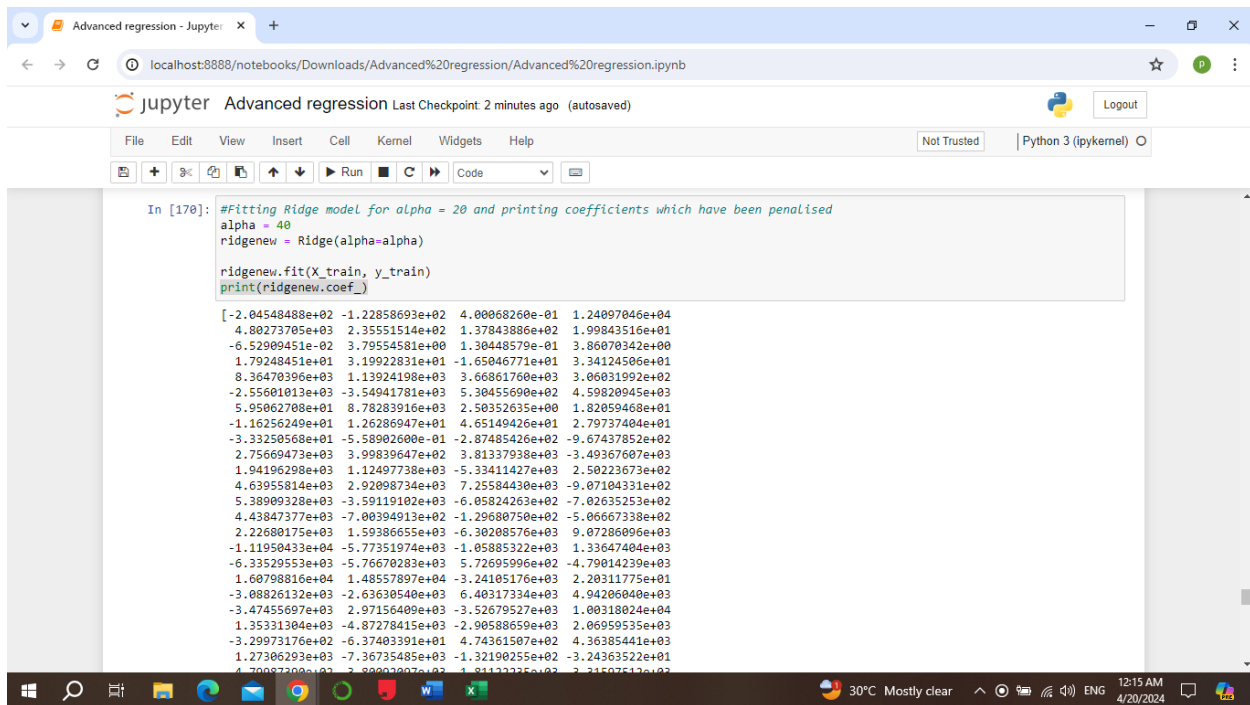
What is the optimal value of alpha for ridge and lasso regression? What will be the changes in the model if you choose double the value of alpha for both ridge and lasso? What will be the most important predictor variables after the change is implemented?

The optimal value of alpha are,

Ridge {'alpha': 20}

Alpha {'alpha': 20}

After choosing double the value of alpha for ridge and lasso, r2score values are dropped in both test and train data



The screenshot shows a Jupyter Notebook window titled "Advanced regression - Jupyter". The browser address bar indicates the notebook is running on localhost:8888. The notebook interface includes a menu bar (File, Edit, View, Insert, Cell, Kernel, Widgets, Help) and a toolbar with icons for file operations, running, and code execution. The code cell contains the following Python code:

```
In [170]: #Fitting Ridge model for alpha = 20 and printing coefficients which have been penalised
alpha = 40
ridgenew = Ridge(alpha=alpha)

ridgenew.fit(X_train, y_train)
print(ridgenew.coef_)
```

The output of the code is a large array of 40 coefficients, each represented in scientific notation. The coefficients are arranged in a single row, separated by spaces. The values range from approximately  $-2.04548488 \times 10^2$  to  $2.31507513 \times 10^2$ .

```
Advanced regression - Jupyter | +
localhost:8888/notebooks/Downloads/Advanced%20regression/Advanced%20regression.ipynb
jupyter Advanced regression Last Checkpoint 2 minutes ago (autosaved)
Python 3 (ipykernel)
File Edit View Insert Cell Kernel Widgets Help
Run
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)

0.8724130948333536
0.8633499180060508
814093612625.8152
385177298389.7433
797349277.7921795
879400224.6341171

In [173]: #Fitting Ridge model for alpha = 20 and printing coefficients which have been penalised
alpha = 40
```

```
Advanced regression - Jupyter | +
localhost:8888/notebooks/Downloads/Advanced%20regression/Advanced%20regression.ipynb
jupyter Advanced regression Last Checkpoint 2 minutes ago (autosaved)
Python 3 (ipykernel)
File Edit View Insert Cell Kernel Widgets Help
Run
797349277.7921795
879400224.6341171

In [173]: #Fitting Ridge model for alpha = 20 and printing coefficients which have been penalised
alpha = 40

lassonew = Lasso(alpha=alpha)
lassonew.fit(X_train, y_train)
print(lassonew.coef_)

[ -1.39444508e+02 -8.62318804e+01  4.60551133e-01  8.94436498e+03
  4.84175280e+03  1.74597804e+02  1.15612288e+02  1.29803608e+01
  1.11785604e+00  6.55988576e+00  1.70581368e+00  1.12567349e+00
  4.55758888e+01  6.55649632e+01  1.01015682e+01  7.88976806e+00
  8.33778447e+03  1.84960658e+03  5.65309479e+03  -7.27853191e+02
 -2.22388789e+03  -1.26999968e+04  -0.00000000e+00  2.24190389e+03
  4.31128001e+01  1.22103803e+04  -1.69091308e+01  9.88752685e+00
  6.50488965e+00  7.60446296e+00  4.07380510e+01  2.10109657e+01
 -1.33553453e+01  -3.77439432e-01  -1.90133963e+02  -6.65936104e+02
  0.00000000e+00  2.04833477e+03  5.44757159e+03  0.00000000e+00
  1.45991952e+04  0.00000000e+00  -2.33065902e+04  9.22050216e+02
  1.09212175e+04  9.78988234e+03  1.17896559e+04  -0.00000000e+00
  1.00259449e+04  -9.69311954e+03  -0.00000000e+00  -4.28834917e+02
  1.38352166e+03  0.00000000e+00  0.00000000e+00  0.00000000e+00
  2.35600073e+02  2.91359588e+03  0.00000000e+00  2.30354109e+04
 -1.38981287e+04  -1.40078194e+03  -3.34517109e+03  -0.00000000e+00
 -1.29678623e+04  -5.92024740e+03  0.00000000e+00  -4.14971097e+03
  4.39719272e+04  2.89349903e+04  -5.53841727e+03  2.07307761e+02
 -3.92683807e+03  1.36388777e+03  2.31002540e+04  2.66301056e+04]
```

```
Advanced regression - Jupyter | x +
localhost:8888/notebooks/Downloads/Advanced%20regression/Advanced%20regression.ipynb
jupyter Advanced regression Last Checkpoint: 3 minutes ago (autosaved)
File Edit View Insert Cell Kernel Widgets Help
Run Not Trusted Python 3 (ipykernel)

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)

0.9133655934819271
0.8491814288336623
552788054173.7316
425114196356.08765
541418270.4933708
970580356.9773691

In [182]: #important predictor variables
betas = pd.DataFrame(index=X_train.columns)
betas.rows = X_train.columns
betas['Ridge'] = ridge.coef_
betas['Ridge_new'] = ridge_new.coef_
```

Most important predictor variables after the change is implemented are,

### Lasso new

Neighborhood_Crawfor	23035.41995
Street_Pave	14599.19525
GarageCars	12210.38031
LandContour_Lvl	11789.65589
LandContour_HLS	10921.21748

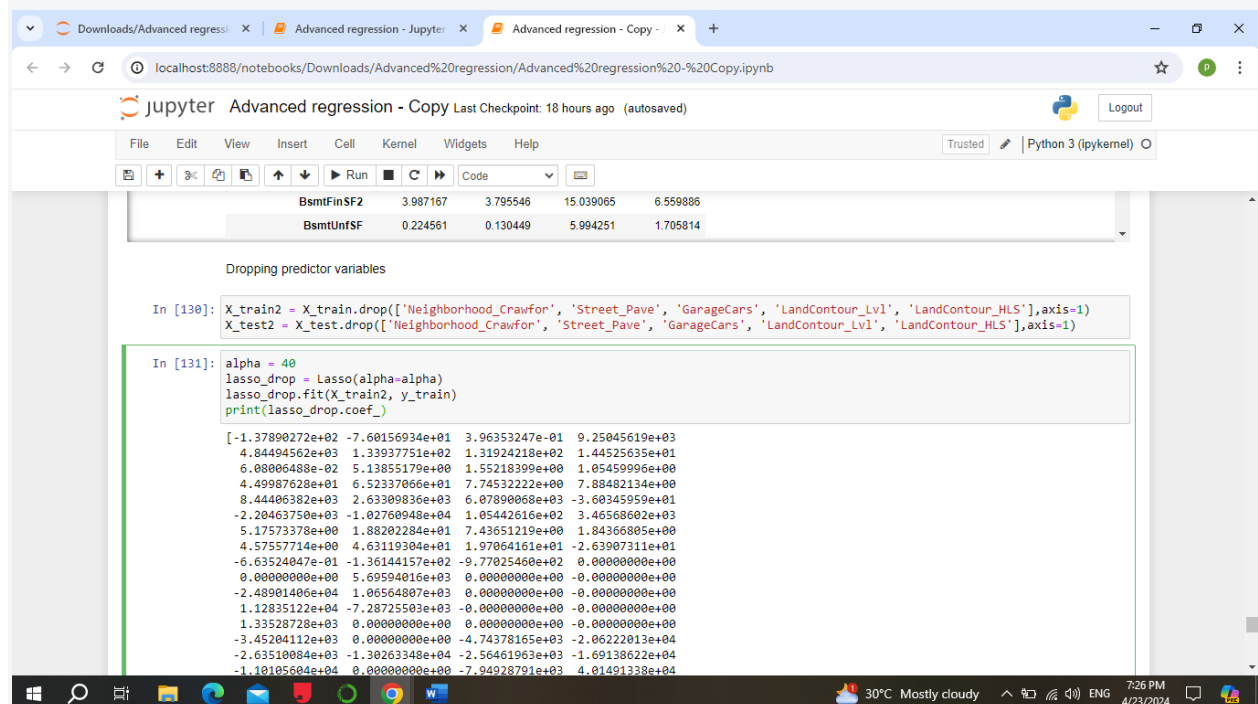
## Question 2

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?

The  $r^2_{\text{train}}$  value of lasso increases whereas the  $r^2_{\text{test}}$  data of lasso decreases. So, we choose  $r^2_{\text{train}}$  value.

## 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?



```
File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel) Logout
```

BsmFinSF2	3.987167	3.795546	15.039065	6.559886
BsmUnfSF	0.224561	0.130449	5.994251	1.705814

Dropping predictor variables

```
In [130]: X_train2 = X_train.drop(['Neighborhood_Crawfor', 'Street_Pave', 'GarageCars', 'LandContour_Lvl', 'LandContour_HLS'],axis=1)
X_test2 = X_test.drop(['Neighborhood_Crawfor', 'Street_Pave', 'GarageCars', 'LandContour_Lvl', 'LandContour_HLS'],axis=1)
```

```
In [131]: alpha = 40
lasso_drop = Lasso(alpha=alpha)
lasso_drop.fit(X_train2, y_train)
print(lasso_drop.coef_)
```

```
[ -1.37890272e+02 -7.60156934e+01  3.96353247e-01  9.25045619e+03
  4.84494562e+03  1.33937751e+02  1.31924218e+02  1.44525635e+01
  6.0806488e-02  5.13855179e+00  1.55218399e+00  1.05459996e+00
  4.49987628e+01  6.52337066e+01  7.74532222e+00  7.88482134e+00
  8.44406382e+03  2.63309836e+03  6.07890068e+03  -3.60345959e+01
 -2.20463750e+03 -1.02760948e+04  1.05442616e+02  3.46568602e+03
  5.17573378e+00  1.88202284e+01  7.43651219e+00  1.84366805e+00
  4.57557714e+00  4.63119304e+01  1.97064161e+01  -2.63907311e+01
 -6.63524047e-01 -1.36144157e+02 -9.77025460e+02  0.00000000e+00
  0.00000000e+00  5.69594016e+03  0.00000000e+00 -0.00000000e+00
 -2.48901406e+04  1.06564807e+03  0.00000000e+00 -0.00000000e+00
  1.12835122e+04 -7.28725503e+03 -0.00000000e+00 -0.00000000e+00
  1.33528728e+03  0.00000000e+00  0.00000000e+00 -0.00000000e+00
 -3.45204112e+03  0.00000000e+00 -4.74378165e+03 -2.0622013e+04
 -2.63510084e+03 -1.30263348e+04 -2.56461963e+03 -1.69138622e+04
 -1.10105604e+04  0.00000000e+00 -7.94928791e+03  4.01491338e+04]
```

30°C Mostly cloudy 7:26 PM 4/23/2024

```

: #important predictor variables
betas = pd.DataFrame(index=X_train2.columns)
betas.rows = X_train2.columns
betas['Lasso_drop'] = lasso_drop.coef_
pd.set_option('display.max_rows', None)
betas.sort_values(by=['Lasso_drop'], ascending=False)

```

```

:

```

	Lasso_drop
RoofMatl_WdShngl	135314.884998
RoofMatl_CompShg	71679.918309
RoofMatl_WdShake	47642.114719
Neighborhood_NoRidge	40149.133817
RoofMatl_Tar&Grv	35804.161375

#### 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?

To ensure the robustness and generalizability of predictive models, we must make sure that the quality of the data is clean. Data must be validated to remove errors, outliers, missing values, and biases. Then using techniques like normalization, scaling, aggregation we refine the variables. Based on the nature and complexity of data available, we have to choose the appropriate model. Also, ensure that overfitting and underfitting is taken care. Regularization techniques are used to reduce the complexity and variance of the model.

If the model is not robust, it cannot be trusted for future analysis.