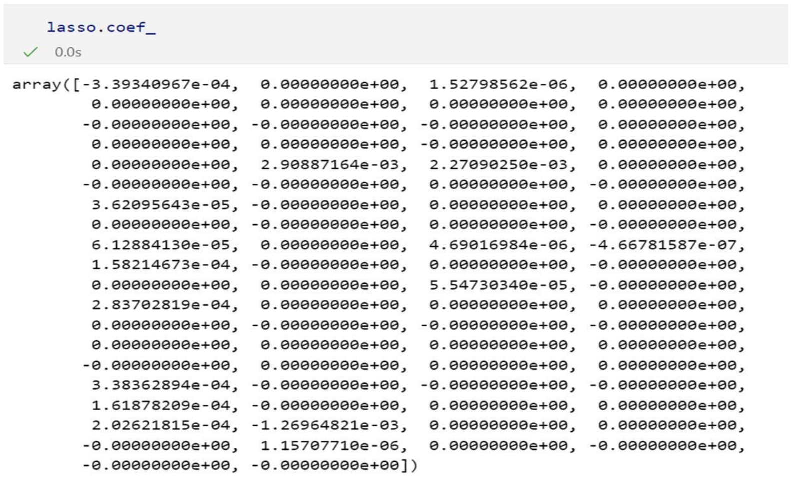
Question 1

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?

Ans: The optimal value of alpha for Ridge is 10 and for Lasso it is 0.5. With these alphas the R2 of the model was approximately 0.83.
After doubling the alpha values in the Ridge and Lasso, the prediction accuracy remains around 0.82 but there is a small change in the co-efficient values. The new model is created and demonstrated in the Jupiter notebook. Below are the changes in the co-efficient.

Lasso Regression Model:

Lasso Co-Efficient

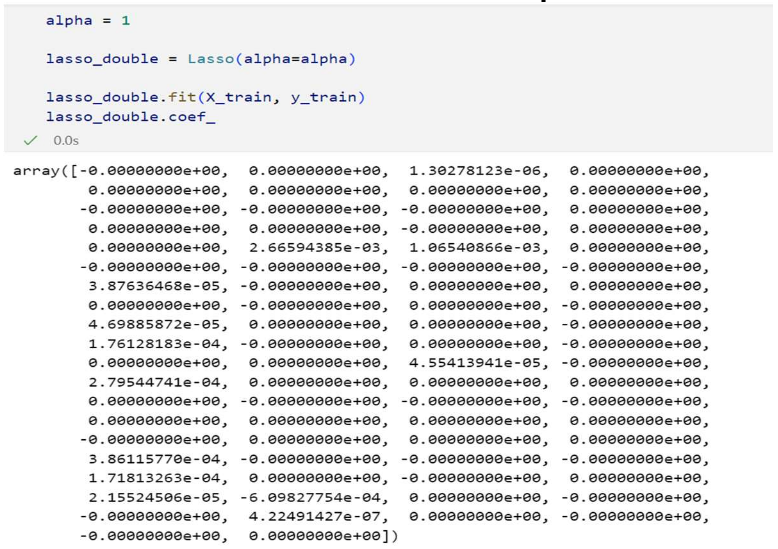


```
lasso.coef_
```

✓ 0.0s

```
array([-3.39340967e-04,  0.00000000e+00,  1.52798562e-06,  0.00000000e+00,
        0.00000000e+00,  0.00000000e+00,  0.00000000e+00,  0.00000000e+00,
       -0.00000000e+00, -0.00000000e+00, -0.00000000e+00,  0.00000000e+00,
        0.00000000e+00,  0.00000000e+00, -0.00000000e+00,  0.00000000e+00,
        0.00000000e+00,  2.90887164e-03,  2.27090250e-03,  0.00000000e+00,
       -0.00000000e+00, -0.00000000e+00,  0.00000000e+00, -0.00000000e+00,
        3.62095643e-05, -0.00000000e+00,  0.00000000e+00,  0.00000000e+00,
        0.00000000e+00, -0.00000000e+00,  0.00000000e+00, -0.00000000e+00,
        6.12884130e-05,  0.00000000e+00,  4.69016984e-06, -4.66781587e-07,
        1.58214673e-04, -0.00000000e+00,  0.00000000e+00, -0.00000000e+00,
        0.00000000e+00,  0.00000000e+00,  5.54730340e-05, -0.00000000e+00,
        2.83702819e-04,  0.00000000e+00,  0.00000000e+00,  0.00000000e+00,
        0.00000000e+00, -0.00000000e+00, -0.00000000e+00, -0.00000000e+00,
        0.00000000e+00,  0.00000000e+00,  0.00000000e+00,  0.00000000e+00,
       -0.00000000e+00,  0.00000000e+00,  0.00000000e+00,  0.00000000e+00,
        3.38362894e-04, -0.00000000e+00, -0.00000000e+00, -0.00000000e+00,
        1.61878209e-04, -0.00000000e+00,  0.00000000e+00,  0.00000000e+00,
        2.02621815e-04, -1.26964821e-03,  0.00000000e+00,  0.00000000e+00,
       -0.00000000e+00,  1.15707710e-06,  0.00000000e+00, -0.00000000e+00,
       -0.00000000e+00, -0.00000000e+00])
```

Lasso Doubled Alpha Co-Efficient



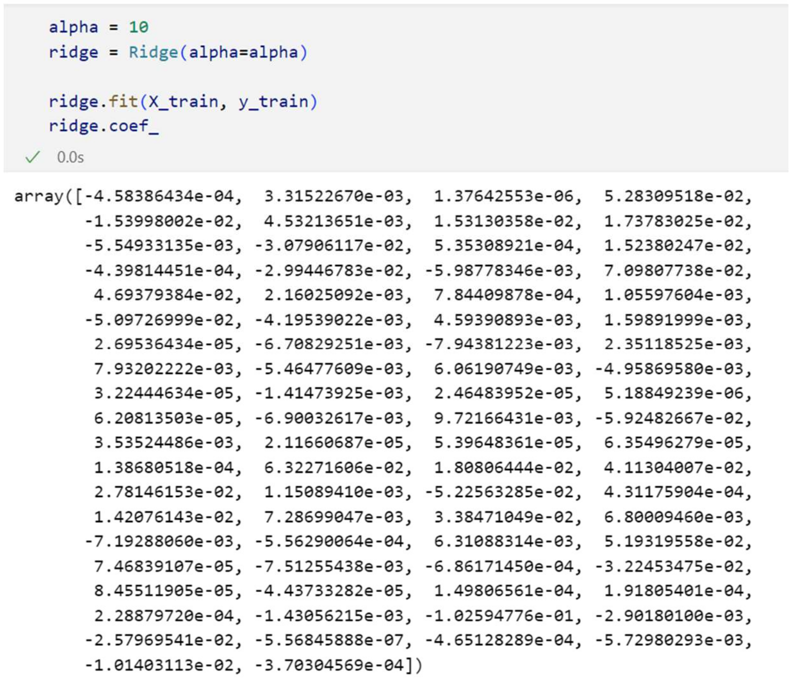
```
alpha = 1

lasso_double = Lasso(alpha=alpha)

lasso_double.fit(X_train, y_train)
lasso_double.coef_
```

✓ 0.0s

```
array([-0.00000000e+00,  0.00000000e+00,  1.30278123e-06,  0.00000000e+00,
        0.00000000e+00,  0.00000000e+00,  0.00000000e+00,  0.00000000e+00,
       -0.00000000e+00, -0.00000000e+00, -0.00000000e+00,  0.00000000e+00,
        0.00000000e+00,  0.00000000e+00, -0.00000000e+00,  0.00000000e+00,
        0.00000000e+00,  2.66594385e-03,  1.06540866e-03,  0.00000000e+00,
       -0.00000000e+00, -0.00000000e+00, -0.00000000e+00, -0.00000000e+00,
        3.87636468e-05, -0.00000000e+00,  0.00000000e+00,  0.00000000e+00,
        0.00000000e+00, -0.00000000e+00,  0.00000000e+00, -0.00000000e+00,
        4.69885872e-05,  0.00000000e+00,  0.00000000e+00, -0.00000000e+00,
        1.76128183e-04, -0.00000000e+00,  0.00000000e+00, -0.00000000e+00,
        0.00000000e+00,  0.00000000e+00,  4.55413941e-05, -0.00000000e+00,
        2.79544741e-04,  0.00000000e+00,  0.00000000e+00,  0.00000000e+00,
        0.00000000e+00, -0.00000000e+00, -0.00000000e+00, -0.00000000e+00,
        0.00000000e+00,  0.00000000e+00,  0.00000000e+00,  0.00000000e+00,
       -0.00000000e+00,  0.00000000e+00,  0.00000000e+00,  0.00000000e+00,
        3.86115770e-04, -0.00000000e+00, -0.00000000e+00, -0.00000000e+00,
        1.71813263e-04,  0.00000000e+00, -0.00000000e+00,  0.00000000e+00,
        2.15524506e-05, -6.09827754e-04,  0.00000000e+00, -0.00000000e+00,
       -0.00000000e+00,  4.22491427e-07,  0.00000000e+00, -0.00000000e+00,
       -0.00000000e+00,  0.00000000e+00])
```

Ridge Regression Model:

Ridge Co-Efficient

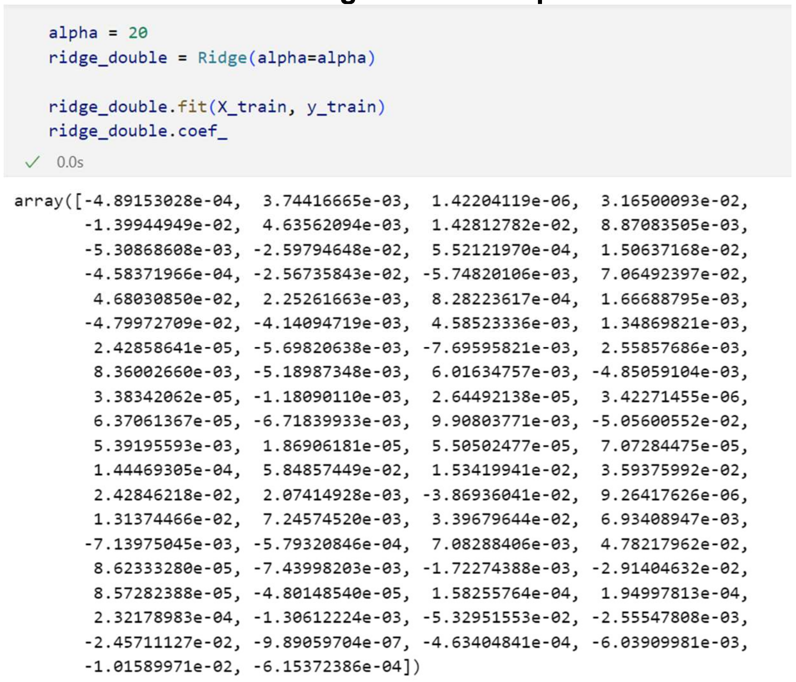


```
alpha = 10
ridge = Ridge(alpha=alpha)

ridge.fit(X_train, y_train)
ridge.coef_
```

✓ 0.0s

```
array([-4.58386434e-04,  3.31522670e-03,  1.37642553e-06,  5.28309518e-02,
       -1.53998002e-02,  4.53213651e-03,  1.53130358e-02,  1.73783025e-02,
       -5.54933135e-03, -3.07906117e-02,  5.35308921e-04,  1.52380247e-02,
       -4.39814451e-04, -2.99446783e-02, -5.98778346e-03,  7.09807738e-02,
        4.69379384e-02,  2.16025092e-03,  7.84409878e-04,  1.05597604e-03,
       -5.09726999e-02, -4.19539022e-03,  4.59390893e-03,  1.59891999e-03,
        2.69536434e-05, -6.70829251e-03, -7.94381223e-03,  2.35118525e-03,
        7.93202222e-03, -5.46477609e-03,  6.06190749e-03, -4.95869580e-03,
        3.22444634e-05, -1.41473925e-03,  2.46483952e-05,  5.18849239e-06,
        6.20813503e-05, -6.90032617e-03,  9.72166431e-03, -5.92482667e-02,
        3.53524486e-03,  2.11660687e-05,  5.39648361e-05,  6.35496279e-05,
        1.38680518e-04,  6.32271606e-02,  1.80806444e-02,  4.11304007e-02,
        2.78146153e-02,  1.15089410e-03, -5.22563285e-02,  4.31175904e-04,
        1.42076143e-02,  7.28699047e-03,  3.38471049e-02,  6.80009460e-03,
       -7.19288060e-03, -5.56290064e-04,  6.31088314e-03,  5.19319558e-02,
        7.46839107e-05, -7.51255438e-03, -6.86171450e-04, -3.22453475e-02,
        8.45511905e-05, -4.43733282e-05,  1.49806561e-04,  1.91805401e-04,
        2.28879720e-04, -1.43056215e-03, -1.02594776e-01, -2.90180100e-03,
       -2.57969541e-02, -5.56845888e-07, -4.65128289e-04, -5.72980293e-03,
       -1.01403113e-02, -3.70304569e-04])
```

Ridge Doubled Alpha Co-Efficient



```
alpha = 20
ridge_double = Ridge(alpha=alpha)

ridge_double.fit(X_train, y_train)
ridge_double.coef_
```

✓ 0.0s

```
array([-4.89153028e-04,  3.74416665e-03,  1.42204119e-06,  3.16500093e-02,
       -1.39944949e-02,  4.63562094e-03,  1.42812782e-02,  8.87083505e-03,
       -5.30868608e-03, -2.59794648e-02,  5.52121970e-04,  1.50637168e-02,
       -4.58371966e-04, -2.56735843e-02, -5.74820106e-03,  7.06492397e-02,
        4.68030850e-02,  2.25261663e-03,  8.28223617e-04,  1.66688795e-03,
       -4.79972709e-02, -4.14094719e-03,  4.58523336e-03,  1.34869821e-03,
        2.42858641e-05, -5.69820638e-03, -7.69595821e-03,  2.55857686e-03,
        8.36002660e-03, -5.18987348e-03,  6.01634757e-03, -4.85059104e-03,
        3.38342062e-05, -1.18090110e-03,  2.64492138e-05,  3.42271455e-06,
        6.37061367e-05, -6.71839933e-03,  9.90803771e-03, -5.05600552e-02,
        5.39195593e-03,  1.86906181e-05,  5.50502477e-05,  7.07284475e-05,
        1.44469305e-04,  5.84857449e-02,  1.53419941e-02,  3.59375992e-02,
        2.42846218e-02,  2.07414928e-03, -3.86936041e-02,  9.26417626e-06,
        1.31374466e-02,  7.24574520e-03,  3.39679644e-02,  6.93408947e-03,
       -7.13975045e-03, -5.79320846e-04,  7.08288406e-03,  4.78217962e-02,
        8.62333280e-05, -7.43998203e-03, -1.72274388e-03, -2.91404632e-02,
        8.57282388e-05, -4.80148540e-05,  1.58255764e-04,  1.94997813e-04,
        2.32178983e-04, -1.30612224e-03, -5.32951553e-02, -2.55547808e-03,
       -2.45711127e-02, -9.89059704e-07, -4.63404841e-04, -6.03909981e-03,
       -1.01589971e-02, -6.15372386e-04])
```

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:

The optimum lambda value in case of Ridge and Lasso is as follows:-

- Ridge – 10
- Lasso – 0.5

The Mean Squared Error in case of Ridge and Lasso are:

- Ridge - 0.0018396090787924262
- Lasso - 0.0018634152629407766

The Mean Squared Error of both the models are almost same. Since Lasso helps in feature reduction (as the coefficient value of some of the features become zero), Lasso has a better edge over Ridge and should be used as the final model.

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?

Ans: The five most important predictor variables in the current lasso model is:-

1. Total_bmst_sf
2. 1stFlrSF
3. GrLiveArea
4. GarageCar
5. GarageArea

After removing these attributes from the dataset, the top new 5 predictors are:

1. GarageYearBlt
2. TotalRoomsAboveGrnd
3. FullBath
4. YearBuilt
5. YearRemodAdd

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?

Ans: As Per, Occam’s Razor— given two models that show similar ’performance’ in the finite training or test data, we should pick the one that makes fewer on the test data due to following reasons:-

- Simpler models are usually more ’generic’ and are more widely applicable
- Simpler models require fewer training samples for effective training than the more complex ones and hence are easier to train.
- Simpler models are more robust.
    - Complex models tend to change wildly with changes in the training data set
    - Simple models have low variance, high bias and complex models have low bias, high variance
    - Simpler models make more errors in the training set. Complex models lead to over fitting — they work very well for the training samples, fail miserably when applied to other test samples

Therefore, to make the model more robust and generalizable, make the model simple but not simpler which will not be of any use.

Regularization can be used to make the model simpler. Regularization helps to strike the delicate balance between keeping the model simple and not making it too naive to be of any use. For regression, regularization involves adding a regularization term to the cost that adds up the absolute values or the squares of the parameters of the model.

Also, Making a model simple leads to Bias-Variance Trade-off:

- A complex model will need to change for every little change in the dataset and hence is very unstable and extremely sensitive to any changes in the training data.
- A simpler model that abstracts out some pattern followed by the data points given is unlikely to change wildly even if more points are added or removed.

Bias quantifies how accurate is the model likely to be on test data. A complex model can do an accurate job prediction provided there is enough training data. Models that are too naïve, for e.g. one that gives same answer to all test inputs and makes no discrimination whatsoever has a very large bias as its expected error across all test inputs are very high.
Variance refers to the degree of changes in the model itself with respect to changes in the training data.