Assignment Part-II

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?

Answer

Optimal value of lambda for Ridge Regression = 10

Optimal value of lambda for Lasso Regression = 0.001

If we double alpha below metrics are obtained

Ridge Regression Metrics

R2 score of train data set decreased from 0.93 to 0.92

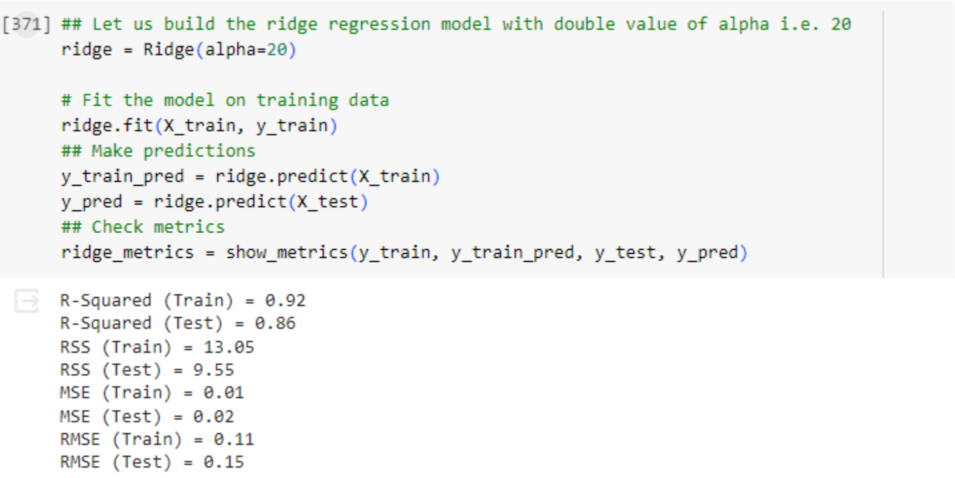
R2 score of test data set decreased from 0.86 to 0.85

Lasso Regression Metrics

R2 score of train data set decreased from 0.91 to 0.88

R2 score of test data set remain same at 0.82

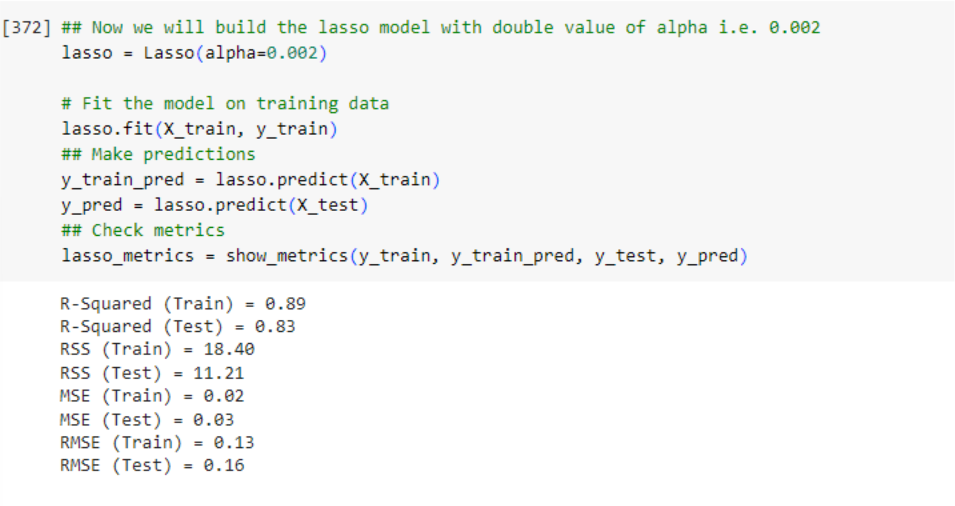
Refer below image which shows the whole metrics such as R2,RSS,MSE,RMSE



```
[371] ## Let us build the ridge regression model with double value of alpha i.e. 20
      ridge = Ridge(alpha=20)

      # Fit the model on training data
      ridge.fit(X_train, y_train)
      ## Make predictions
      y_train_pred = ridge.predict(X_train)
      y_pred = ridge.predict(X_test)
      ## Check metrics
      ridge_metrics = show_metrics(y_train, y_train_pred, y_test, y_pred)
```

```
R-Squared (Train) = 0.92
R-Squared (Test) = 0.86
RSS (Train) = 13.05
RSS (Test) = 9.55
MSE (Train) = 0.01
MSE (Test) = 0.02
RMSE (Train) = 0.11
RMSE (Test) = 0.15
```



```
[372] ## Now we will build the lasso model with double value of alpha i.e. 0.002
      lasso = Lasso(alpha=0.002)

      # Fit the model on training data
      lasso.fit(X_train, y_train)
      ## Make predictions
      y_train_pred = lasso.predict(X_train)
      y_pred = lasso.predict(X_test)
      ## Check metrics
      lasso_metrics = show_metrics(y_train, y_train_pred, y_test, y_pred)
```

```
R-Squared (Train) = 0.89
R-Squared (Test) = 0.83
RSS (Train) = 18.40
RSS (Test) = 11.21
MSE (Train) = 0.02
MSE (Test) = 0.03
RMSE (Train) = 0.13
RMSE (Test) = 0.16
```

Some of the important predictor variables after we double the alpha values are: -

- GrLivArea
- OverallQual_8
- OverallQual_9
- Functional_Typ
- Neighborhood_Crawfor
- Neighborhood_NridgHt
- Neighborhood_Somerst
- CentralAir_Y

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?

Answer

We will choose lasso regression as it also acts as method of feature selection. Among 316 variables 206 features have been removed by Lasso. But Ridge will not reduce columns, it will keep all 306 variables with the reducing the coefficient of variables. Even though the model performance by Ridge Regression was better in terms of R2 values of Train and Test, it is better to use Lasso, since it brings and assigns a zero value to insignificant features, enabling us to choose the predictive variables. It is always advisable to use simple yet robust model.

Question 3

After building the model, you realized 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?

Answer

Let’s drop the top 5 features in Lasso model and build the model again.
Top 5 predictors were.

OverallQual_9,GrLivArea,Neighborhood_Crawfor,
CentralAir_Y,OverallQual_8

Below are the five most important predictor variables after building new Lasso regression model excluding the five most important predictors:

- 2ndFlrSF
- Neighborhood_NridgHt
- 1stFlrSF
- Exterior1st_BrkFace
- Neighborhood_Somerst

Question 4

How can you make sure that a model is robust and generalizable? What are the implications of the same for the accuracy of the model and why?

Answer:

Ensuring that a machine learning model is robust and generalizable is crucial to its performance and applicability. Robustness refers to the model's ability to perform consistently in various real-world conditions, while generalizability refers to its ability to perform well on unseen data. Here are some strategies to make sure a model is robust and generalizable:

Quality Data Collection: Start with high-quality and diverse data. Ensure that the training dataset is representative of the real-world scenarios the model will encounter.

Data Preprocessing: Carefully preprocess and clean the data. This may involve handling missing values, removing outliers, and scaling the features.

Feature Engineering: Select or engineer features that are most relevant to the problem at hand.

Cross-Validation: Use techniques like k-fold cross-validation to evaluate the model's performance on multiple subsets of the data. This helps in estimating the model's generalization performance and identifying potential overfitting.

Regularization: Apply Ridge/Lasso regression to prevent overfitting. Regularization encourages the model to have simpler, more generalizable representations.

Model Selection: Experiment with different algorithms and model architectures.

Hyperparameter Tuning: Tune the model's hyperparameters to find the right balance between underfitting and overfitting. Grid search or random search can help in this process.

Data Augmentation: For image or text data, data augmentation techniques can be used to increase the diversity of the training data, making the model more robust.

Transfer Learning: If applicable, leverage pre-trained models and fine-tune them for your specific task. Transfer learning can significantly improve generalization.

Testing on Unseen Data: Finally, evaluate the model's performance on a separate test set that it has not seen during training. This is a critical step to assess how well the model generalizes to new data.

The implications of these strategies for the accuracy of the model are as follows:

Balancing Robustness and Accuracy: Improving robustness may come at the cost of a slight decrease in accuracy on the training data. This trade-off is necessary to ensure the model doesn't overfit and can handle unseen data effectively.

Generalization Accuracy: The goal is not just high accuracy on the training data but also strong performance on unseen data. If a model is too optimized for the training data, it may not generalize well to new data, resulting in poor accuracy in real-world scenarios.

Validation and Testing: The model's accuracy on validation and test data is a better measure of its true performance, as opposed to the training data. These metrics provide insights into how well the model generalizes, which is often more important than achieving high accuracy on the training set.

In summary, the key to building a robust and generalizable model is striking a balance between accuracy on the training data and the ability to perform well in real-world situations. It's crucial to evaluate a model's performance on unseen data to ensure that it can handle the variability and noise present in the real world.