PDF Report (Group 14)
PRELIM PROJECT

Singular Linear Regression

In performing Linear regression, we used the libraries scikit, specifically sklearn for machine learning, matplotlib for the data visualization, and numpy for its various libraries. We performed the singular linear regression on the dataset “iris.csv” which contained 151 rows of data regarding the sepal length and width and petal length and width of 3 different varieties of flowers.

Before fitting the model with our data, we prepared it by splitting it into two sets, one for training and one for testing. After fitting the model with our training data, we used the testing data to develop a prediction. We got a coefficient value of 0.4053 and a mean squared error of 0.1291 which suggests that the model we trained got values which are close the the actual values indicating that our model is accurate.

Multiple Linear Regression

Once again, to perform linear regression we used the same libraries as we did in performing the singular linear regression. Sklearn, numpy, and matplotlib.

The procedures we did were also the same however there was an added step this time because we needed to use multiple variables hence the name “multiple linear regression”. We prepared our data by first dropping the columns which weren’t needed. These columns were “variety” and “sepal.length”. After doing so, we fitted the model with our training data and made predictions using the test data. Instead of using the coefficients and the mean squared error, we used the predicted R2 scores as our metric in determining if our model was successful. Our training data got an R2 score of 0.857 while our testing data got an R2 score of 0.913. By analyzing our predicted R2 scores, we can determine that the model fits our data.

Polynomial Linear Regression

As for the Polynomial Linear Regression, we still used the iris.csv dataset to model the relationship between the predictor variable (Petal Length) and the response variable (Sepal Length). Moreover, we are still going to use the previous linear regression by transforming the input features into polynomial features and fitting the model accordingly.

To start, we used a degree of 2 to predict the sepal length based on petal length but since the values from the dataset is random, the visualization doesn’t illustrate the fitted curve but rather a “connect-the-dots” representation of the data points. The mean squared error of 0.097 indicated that the model performed very well.

Logistic Regression

In performing the logistic regression on the ‘iris.csv’ dataset, we made use of different libraries, namely numpy, scikit, and pandas. We also used the libraries matplotlib and seaborn in order to properly visualize the data that we got.

To start, we prepared our data by turning all the non-numerical values into numerical values. In our case, the only values that were non-numerical were the values in the ‘variety’ column. After doing so, we then split our dataset into two, one for training and one for testing. After preparing the data, we fitted the model with our training data and proceeded to use the model to make predictions using the testing data. In order to measure the fit of the model, we used a confusion matrix. The confusion matrix we got was [29, 0, 0], [0, 23, 0], [0, 0, 23]. This simply indicates that our model was able to successfully identify each class without misclassifications. This implies that our model performed very well as it did not misclassify any data.

Decision Tree

For this model, we used iris.csv to classify different types of iris flowers. This dataset is a great example of using a decision tree classification since we are able to distinguish between the different types of iris flowers based on their features such as sepal length, sepal width, petal length and petal width.

We simply loaded the iris data set from the sklearn, and identified the target labels which is the type of iris flower, and the features which are the petal width and length. Then, we started to create the decision tree classifier using the function from sklearn.tree library. After doing so, we set the criterion parameter to ‘entropy’ meaning that the algorithm should use information gain to make decisions at each node. Additionally, we set the maximum depth of the tree to 2 which helps prevent overfitting the model and keeping it simple. Finally, we trained the model using the fit method and displaying the graph itself. The results show that each node represents a decision based on the characteristics of the flowers, specifically their petal length and width which are our features.

Random Forest

This model is defined as multiple decision trees working together to provide a single output. Each tree is constructed using a random subset of the data set to measure a random subset of features in each partition. In this model, we made use of the iris.csv data set and numerous libraries such as pandas, numpy, and scikit.

Initially, we loaded the iris.csv data set and converted it to a data frame using the pandas function. Then, we added a new column for species that indicates the type of flower for every row. Then, we split the dataset into train and test data which will be used for training the model and testing the prediction. We then proceeded to create a list called ‘features’ that contains the names of the first 4 columns and converted the species names into numerical values using the factorize function from pandas. Next, we created the random forest classifier, trained it with the features and their corresponding species names, and made predictions using the test dataset. Finally, we generated a confusion matrix to evaluate the performance of the model. The confusion matrix that we got was [13, 0, 0],[0, 5, 2], [0, 0, 12] indicating that our model is 93% accurate in predicting the species of iris flowers based on their characteristics.