DATA ANALYSIS OF IRIS DATASET

PROJECT REPORT FOR STAT 598Z (SPRING 2018)

Pradyuth Vangur, Upasana Angara

|

CONTENTS

PROJECT OVERVIEW

1.1 PROJECT OBJECTIVES

The objectives of the project are to gain a good grasp of R through the implementation of supervised learning algorithms as well as R packages for data visualization.

The dataset was borrowed from the data repository at www.kaggle.com and is called the Iris dataset.

This provides an excellent platform for building competence in R for data science in industrial applications.

1.2 DESCRIPTION OF THE DATASET

The Iris dataset The Iris flower data set or Fisher's Iris data set is a multivariate data set introduced by the British statistician and biologist Ronald Fisher. [1]

The data set contains 3 classes of 50 instances each, where each class refers to a type of iris plant.

Attribute Information:

1. Sepal length in cm: The length of the sepal of the Iris flower
2. Sepal width in cm: The width of the sepal of the Iris flower
3. Petal length in cm: : The length of the petal of the Iris flower
4. Petal width in cm: The width of the petal of the Iris flower
5. The type of species which are described as class are as follows:
   -- Iris Setosa
   -- Iris Versicolour
   -- Iris Virginica

1.3. INITIAL DATA ANALYSIS OBJECTIVES

Our objectives were aimed at implementing the following, the project outline therefore include:

a. Use multiple ‘ggplot’ and ‘ggvis’ tools to study the trends to establish the predictor and relations; simultaneously use non-trivial programming options like pipe operators and tibble in packages like ‘tidyverse’ to accomplish the same.

b. Perform cross-validation and determine the supervised machine learning k-nearest neighbours algorithm with the highest prediction accuracy.

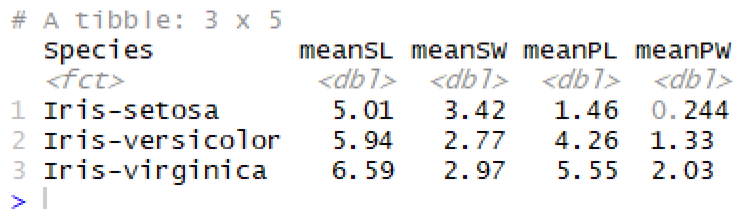
c. Based on the results obtained in part (b), use this pattern recognition algorithm to assign the output, a class membership. Classifying the object by a majority vote of its neighbors, with the object being assigned to the class most common among its *k* nearest neighbors

2. DATA ANALYSIS

2.1 PRELIMINARY GRAPHICAL ANALYSIS

Preliminary analysis of the effect of each of the indicators towards the Species of the Iris flower is done using several plots and correlations generated using 'tidyverse', 'ggplot' and 'ggvis'.

At first, we basically run a descriptive stats on the data set to find out the measures such as means of the numerical variables based on the species.



```
# A tibble: 3 x 5
  Species         meanSL meanSW meanPL meanPW
  <fct>            <dbl>  <dbl>  <dbl>  <dbl>
1 Iris-setosa       5.01   3.42   1.46  0.244
2 Iris-versicolor   5.94   2.77   4.26  1.33
3 Iris-virginica    6.59   2.97   5.55  2.03
> |
```

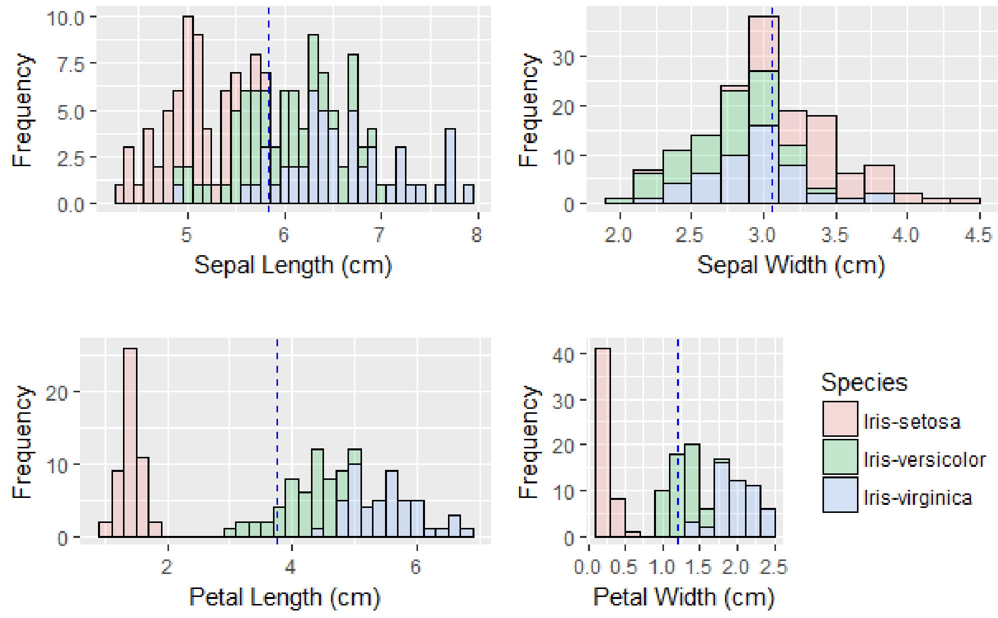
Data is visualized to understand if they all represent normal distributions and whether or not any data cleaning needs to be implemented.

Observing the histograms for each of the numerical variables, we find that each of the species exhibits a normal distribution based on explanatory variables.

There are no outliers that required to be treated in our dataset.

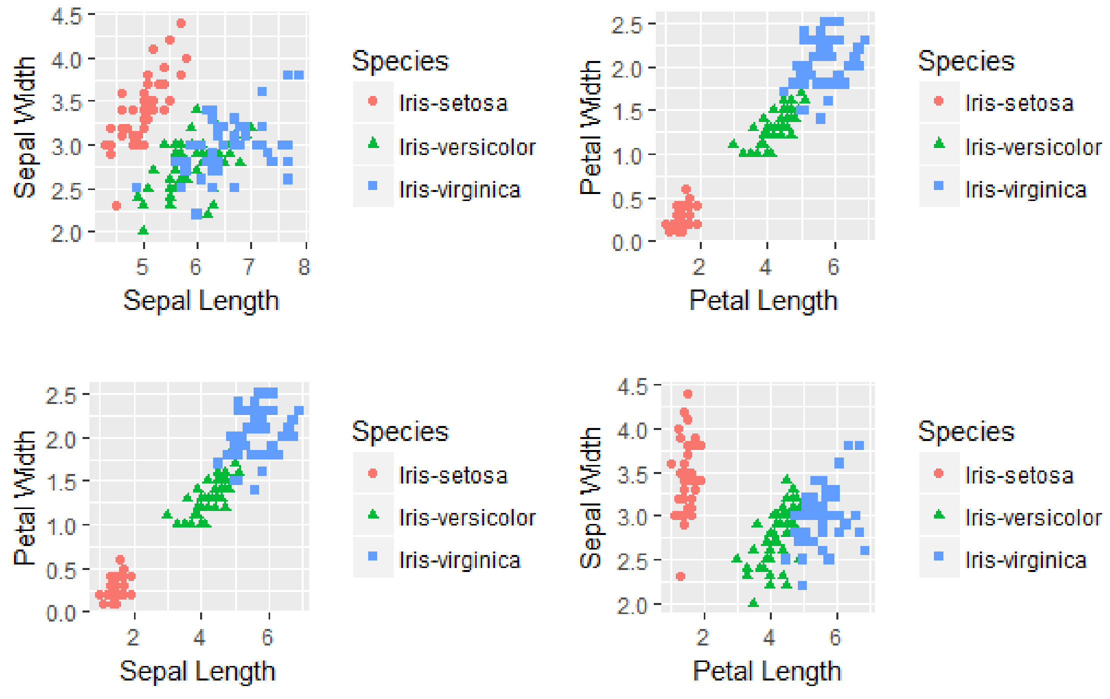
Several tidyverse commands such as mutate, subset, filter, group_by, summarize were also implemented in analyzing various aspects of the data.

Iris Frequency Histogram



We also analyze the correlations between each of these variables to understand if the species variable can be easily separated, so as to be able to build supervised learning algorithms over which we will build our algorithm and test the accuracy.

Scatter Plots



Given that there is sufficient variation such that we can observe the spread of species presented on the graph. This suggests that this current data set would be appropriate for running our classification algorithm. We can proceed to run our supervised learning algorithm that will basically classify the output by a majority vote of its neighbors, with the object being assigned to the class most common among its *k* nearest neighbors.

2.2 ESTABLISHING THE PREDICTOR

Based on the plots above, the categorical variable species would be our response variable that would be predicted based on the explanatory variables Sepal Length, Sepal Width, Petal Length and Petal Width. This the model based on which we continued to design the k nearest neighbors algorithm.

3. BUILDING THE ALGORITHM

3.1 **K nearest neighbors algorithm**

Splitting the data into test and training dataset :

- As an example, the data was split into two parts: ‘train’, consisting of 85% (or 127 rows) of the data and ‘test’, consisting of the remaining 15% (or 23 rows) of the data. The KNN algorithm models were then first trained with ‘train’ data and later tested on the ‘test’ data. The models were evaluated based on responses of the test data with the model and the actual responses.

Calculation of Euclidean distance of the test set with respect training dataset :

- A function to calculate the euclidean distance is used to calculate for each value of the test vector passed onto the training vector and stored in a matrix called dist (in this case, the matrix dimensionality is 127 X 23

Finding the k least distances for each row of the distance matrix:

- The function then sorts the distances based on ascending order of the magnitude of distances and returns the indices for ascending order of k least distances for each row.

Finding the majority of k nearest neighbours:

- It then calculates a majority vote of the k nearest neighbours and assigns the test vector to the majority class and returns this as the predicted class.

Calculating the accuracy of the response:

- The output of this classification algorithm was calculated as the class with the highest occurrences from the K-most similar instances. Each instance here, makes a majority vote for its class and the class with the most votes is taken as the prediction.
- The accuracy therefore is calculated as the outputs of the predicted classes / the actual classes of the test dataset

3.2 Cross validation

i) Test and Training data sets
A function was written to split the data into Test and Train.
It was run for several different cases
i) 95% and 5%
i) 85% and 15%
ii) 70% and 30%
iii) 50% and 50%

- Each of these data sets was run for several different values of k for example,
  k =4,7,8,9,11 etc.
- Once the best fit on the training data set was found, all the models mentioned above was fit on the training data set.

- The best model was picked to be evaluated on the test data set.

- Once the model is evaluated on the test data set, the output we get is the accuracy .

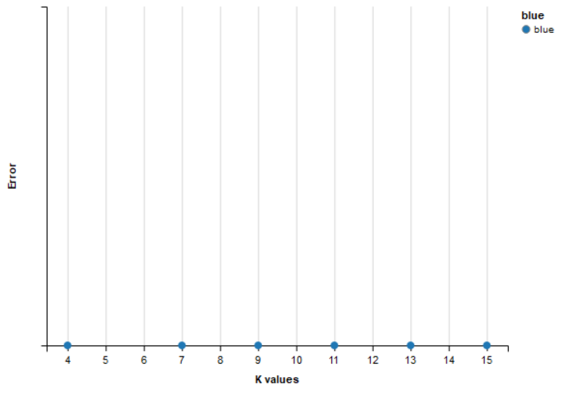
- Cross validation was performed on different values of K to determine the best fit model for our algorithm.The accuracy for each of these k values was compared to eventually find the best k.

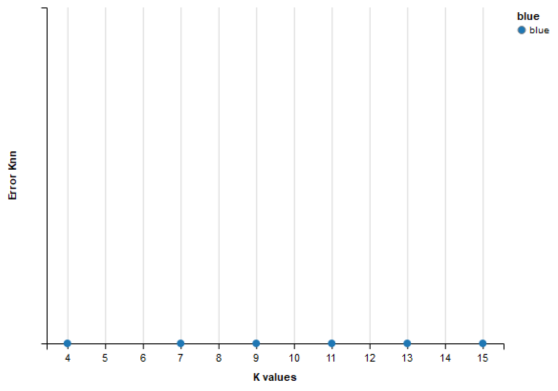
3.3 Response based upon the accuracy for different values of K

In the plots below, we can find the error(1- accuracy) versus different values of K for each of the 4 different splits of test and training data we have evaluated in split data function.[2]

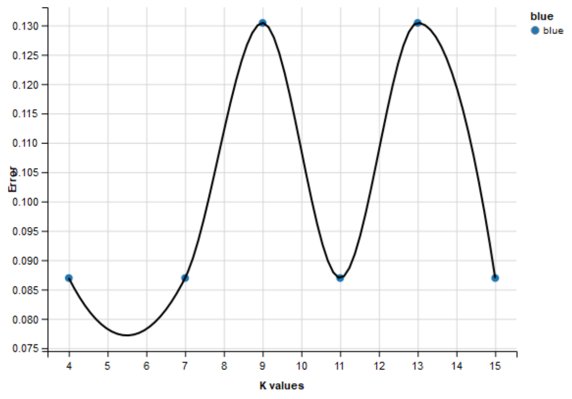
L: Plot for our algorithm and R: Plot from Knn function

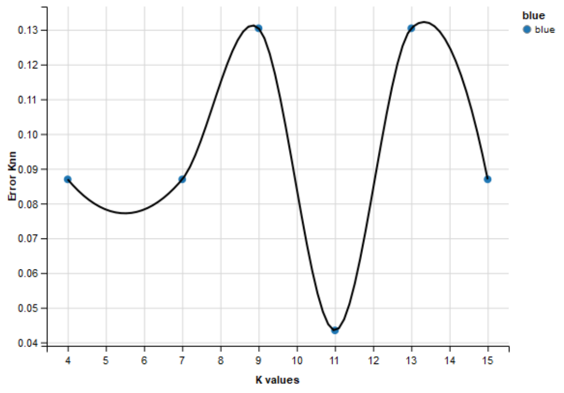
1) **Split data for Training:0.95 and Test 0.5**



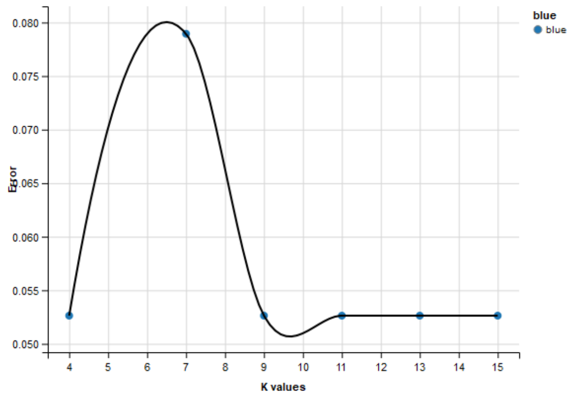


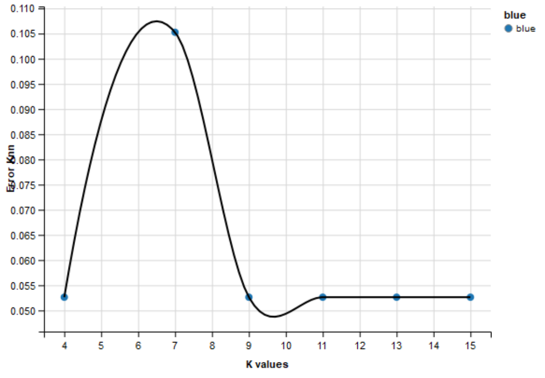
2) **Split data for Training:0.85 and Test 0.15**



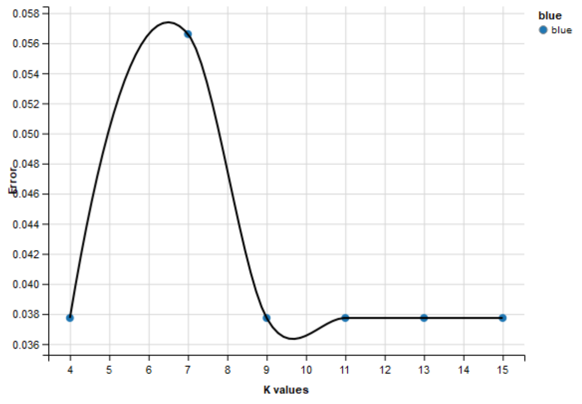


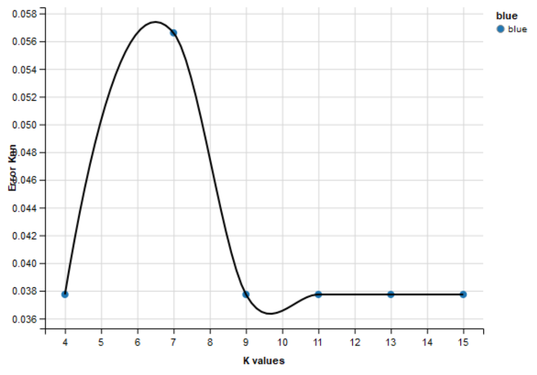
3) **Split data for Training:0.75 and Test 0.25**



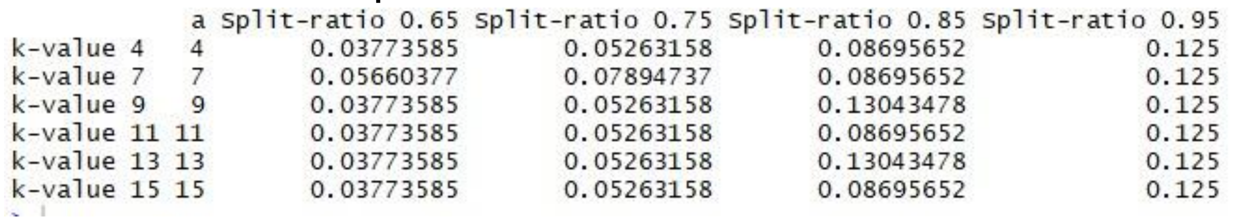


4) **Split data for Training:0.65 and Test 0.35**

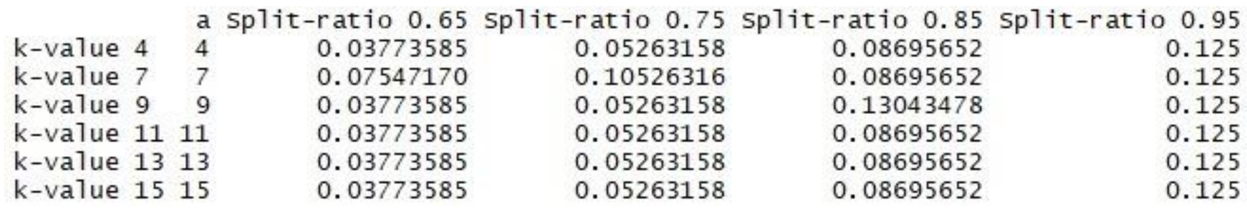




Tabulated values for the Accuracy of our algorithm based on the cross validation k values and different data splits


```
              a Split-ratio 0.65 Split-ratio 0.75 Split-ratio 0.85 Split-ratio 0.95
k-value 4     4       0.03773585       0.05263158       0.08695652            0.125
k-value 7     7       0.05660377       0.07894737       0.08695652            0.125
k-value 9     9       0.03773585       0.05263158       0.13043478            0.125
k-value 11   11       0.03773585       0.05263158       0.08695652            0.125
k-value 13   13       0.03773585       0.05263158       0.13043478            0.125
k-value 15   15       0.03773585       0.05263158       0.08695652            0.125
```

Tabulated values for the Accuracy of KNN algorithm(package) based on the cross validation k values and different data splits


```
              a Split-ratio 0.65 Split-ratio 0.75 Split-ratio 0.85 Split-ratio 0.95
k-value 4     4       0.03773585       0.05263158       0.08695652            0.125
k-value 7     7       0.07547170       0.10526316       0.08695652            0.125
k-value 9     9       0.03773585       0.05263158       0.13043478            0.125
k-value 11   11       0.03773585       0.05263158       0.08695652            0.125
k-value 13   13       0.03773585       0.05263158       0.08695652            0.125
k-value 15   15       0.03773585       0.05263158       0.08695652            0.125
```

4. CONCLUSIONS

Based on the plots, above we can find that best set to pick is the 0.65 data set split between training and test data i.e 65% of the data is training and 35% is test .
The value for k at which the graph starts converging is 9.

5. SCOPE FOR IMPROVEMENT

- The algorithm threw errors for a few values of K and it was diagnosed that the number of rows returned by the split function was causing both training and test sets to round off to the next nearest integer. We found that the split function is returning 1 row extra within the accuracy function only for particular values of split data and K.

- We may be able to vectorize some parts of our code and make it comparable to the speed of the KNN algorithm, although at this point the speed of our algorithm is quite comparable to the knn function.

6. CITATIONS

[1] https://archive.ics.uci.edu/ml/datasets/iris

[2]https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/

______________________________END OF REPORT_____________________________