Datamining (CSCI-B 565) - Homework 2

Abhinandan Sampathkumar(asampath)

Question 1)

The mean and standard deviation are as follows: Run 1.py

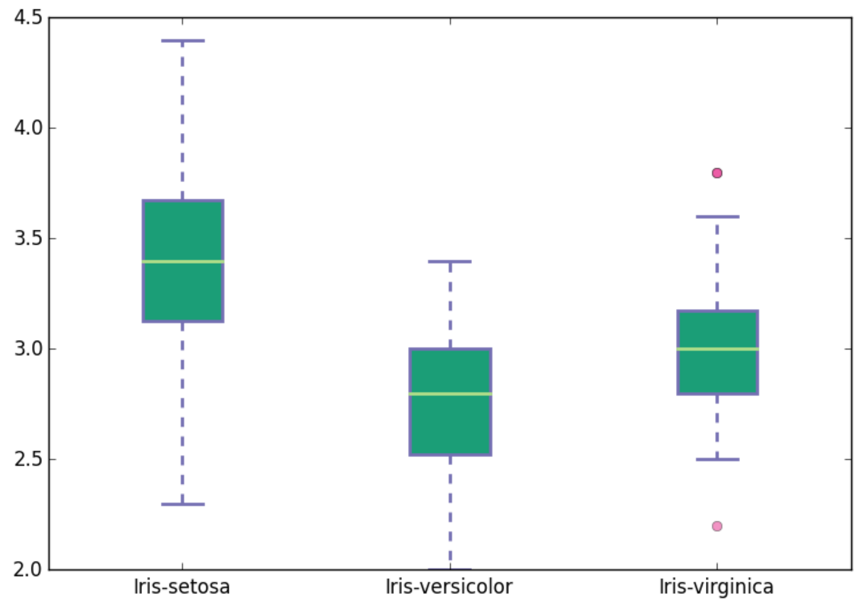
```
Mean of Sepal Length is 5.84
Std_Dev of Sepal Length is 0.83
Mean of Sepal Width is 3.05
Std_Dev of Sepal Width is 0.43
Mean of Petal Length is 3.76
Std_Dev of Petal Length is 1.76
Mean of Petal Width is 1.2
Std_Dev of Petal Width is 0.76
```

Mean, standard deviation for each flower are as follows: Run 1b.py

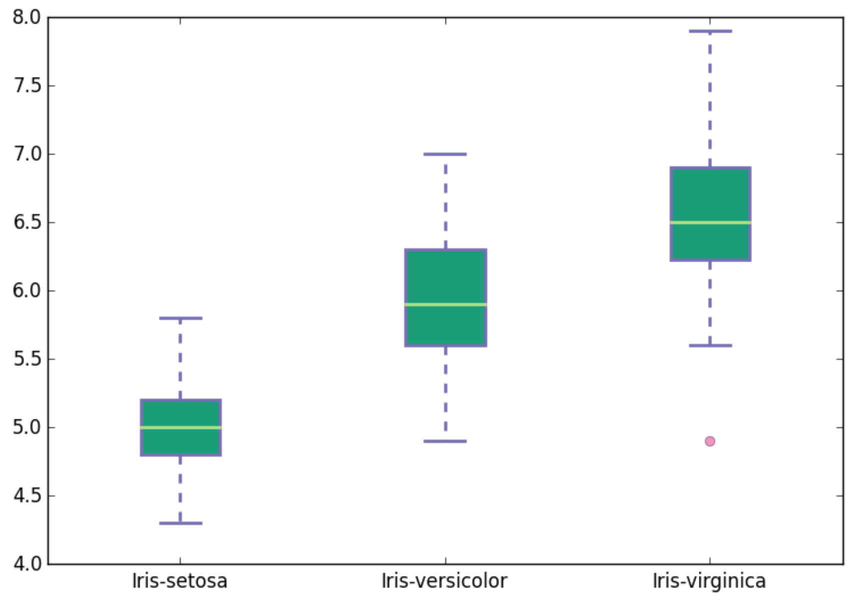
```
Iris-setosa
Mean of Sepal Length is 5.01
Std_Dev of Sepal Length is 0.35
Mean of Sepal Width is 3.42
Std_Dev of Sepal Width is 0.38
Mean of Petal Length is 1.46
Std_Dev of Petal Length is 0.17
Mean of Petal Width is 0.24
Std_Dev of Petal Width is 0.11
------------------------------
Iris-versicolor
Mean of Sepal Length is 5.94
Std_Dev of Sepal Length is 0.51
Mean of Sepal Width is 2.77
Std_Dev of Sepal Width is 0.31
Mean of Petal Length is 4.26
Std_Dev of Petal Length is 0.47
Mean of Petal Width is 1.33
Std_Dev of Petal Width is 0.2
------------------------------
Iris-virginica
Mean of Sepal Length is 6.59
Std_Dev of Sepal Length is 0.63
Mean of Sepal Width is 2.97
Std_Dev of Sepal Width is 0.32
Mean of Petal Length is 5.55
Std_Dev of Petal Length is 0.55
Mean of Petal Width is 2.03
Std_Dev of Petal Width is 0.27
```

Box plots for each feature : Run 1c.py

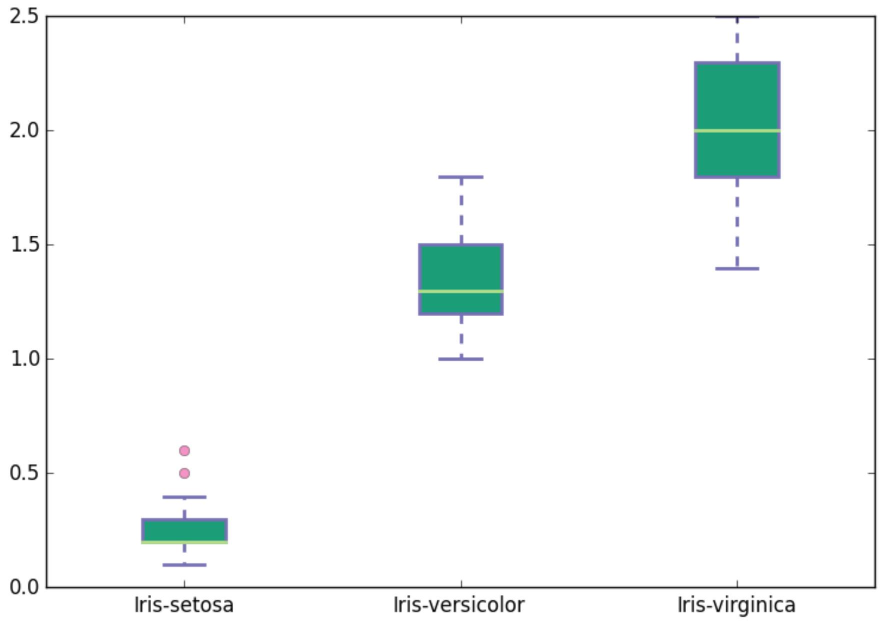
Sepal Width plot for all flowers



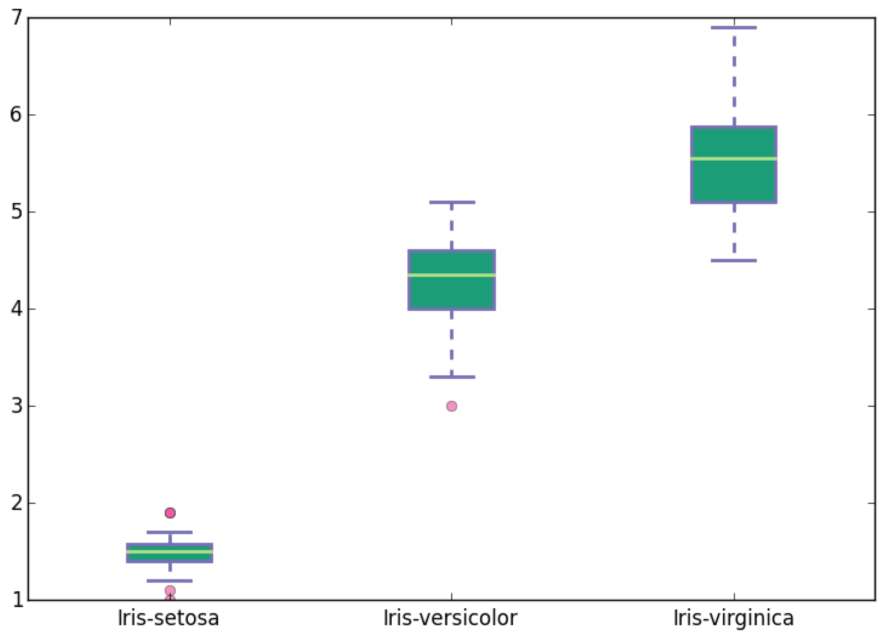
Sepal Length plot for each flower type



PetalWidth plot for each type of flower



Petal length plot for each type of flower



Question 2)

Scatter Plots for most and least 4 correlated pairs of features are attached in the submission with their column names.
Assuming the index of first column to be 0, which is the class variable. Below are the most and least 4 correlated columns and their values.

Output: Run python file 2a.py for this output and plots.

```
Filepath is:wine1.data.txt
max co-efficients are [ 0.65269177  0.69994936  0.7871939   0.8645635 ]
their indexes[(8, 10), (10, 8)]
their indexes[(7, 13), (13, 7)]
their indexes[(8, 13), (13, 8)]
their indexes[(7, 8), (8, 7)]
minimum 4 co-efficients are [ 0.00391123  0.00965194  0.01873198  0.0553982 ]
their indexes[(4, 13), (13, 4)]
their indexes[(4, 10), (10, 4)]
their indexes[(5, 11), (11, 5)]
their indexes[(6, 12), (12, 6)]
```

Euclidean Distance:

Output: Run python file 2b.py for this output

```
In class 1, the % of points whose close neighbors belong to same class
88.13559322033898
In class 2, the % of points whose close neighbors belong to same class
76.05633802816901
In class 3, the % of points whose close neighbors belong to same class
64.58333333333334

at datalevel --- > 76.96629213483146
```

With normalization: Run python file 2b.py for this output

Change flag variable to 1 for z-score and 0 for 0-1 normalization

Z-Score Normalization:

Output :

```
In class 1, the % of points whose close neighbors belong to same class 100.0
In class 2, the % of points whose close neighbors belong to same class
90.14084507042254
In class 3, the % of points whose close neighbors belong to same class 100.0
at datalevel 96.06741573033707
```

0-1 Normalization:

Output :

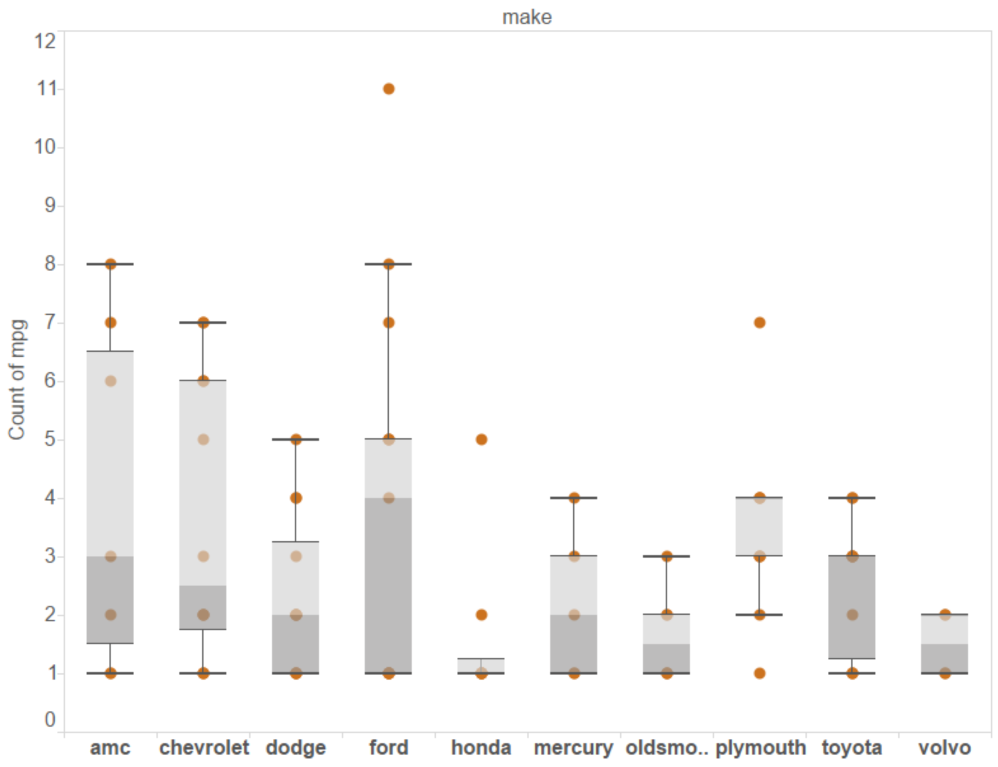
```
In class 1, the % of points whose close neighbors belong to same class 100.0
In class 2, the % of points whose close neighbors belong to same class 100.0
In class 3, the % of points whose close neighbors belong to same class 100.0
at datalevel 100.0
```

With Normalizing the data all the input variables would fall under the same scale in the model and results do not vary due to the units of the input. So with normalizing the data, discrepancy decreases. SAfter normalizing the data all the closest neighbors were in the same class for wine data in 0-1 normalization. Overall the results improved after normalizing the data.

Question 3)

Auto MPG dataset

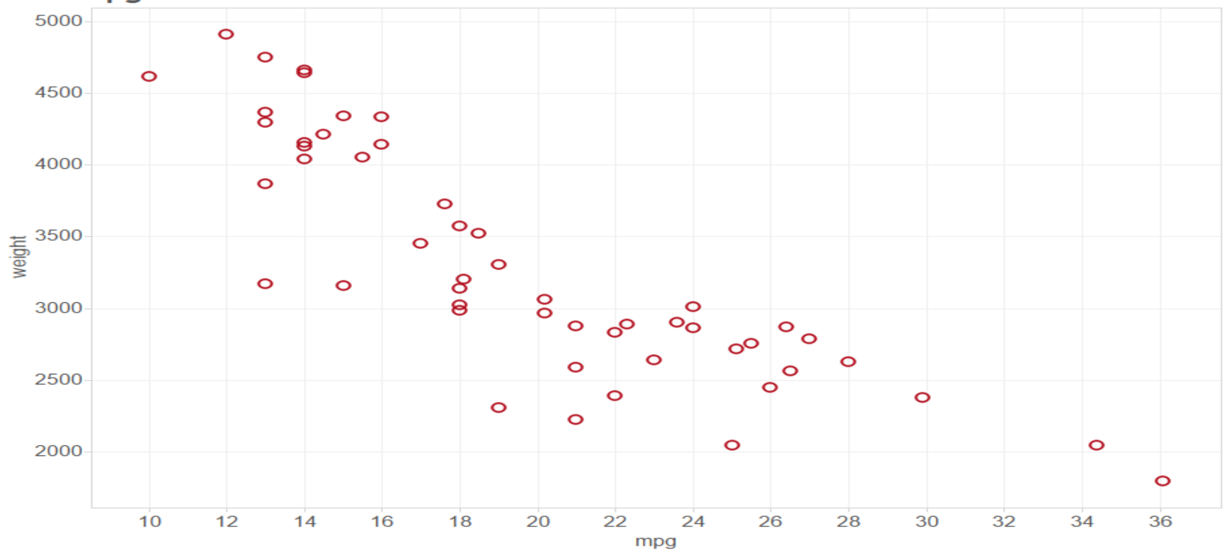
make_mpg_BoxPlot



Count of mpg for each make. Details are shown for mpg (bin). The view is filtered on make, which keeps 10 of 37 members.

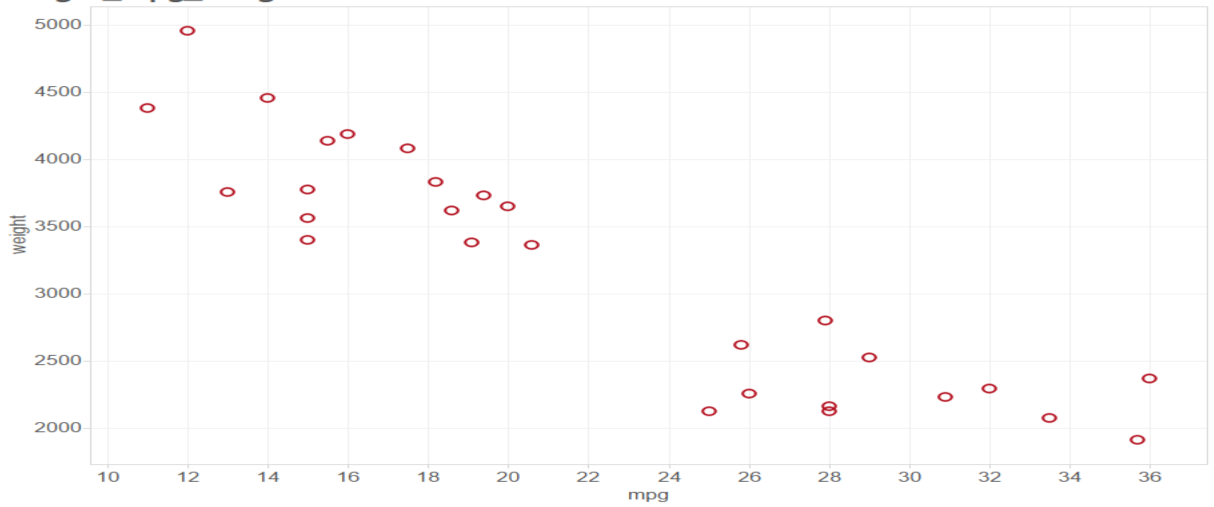
Scatter plots of 5 makes of weight vs mpg

w vs mpg ford



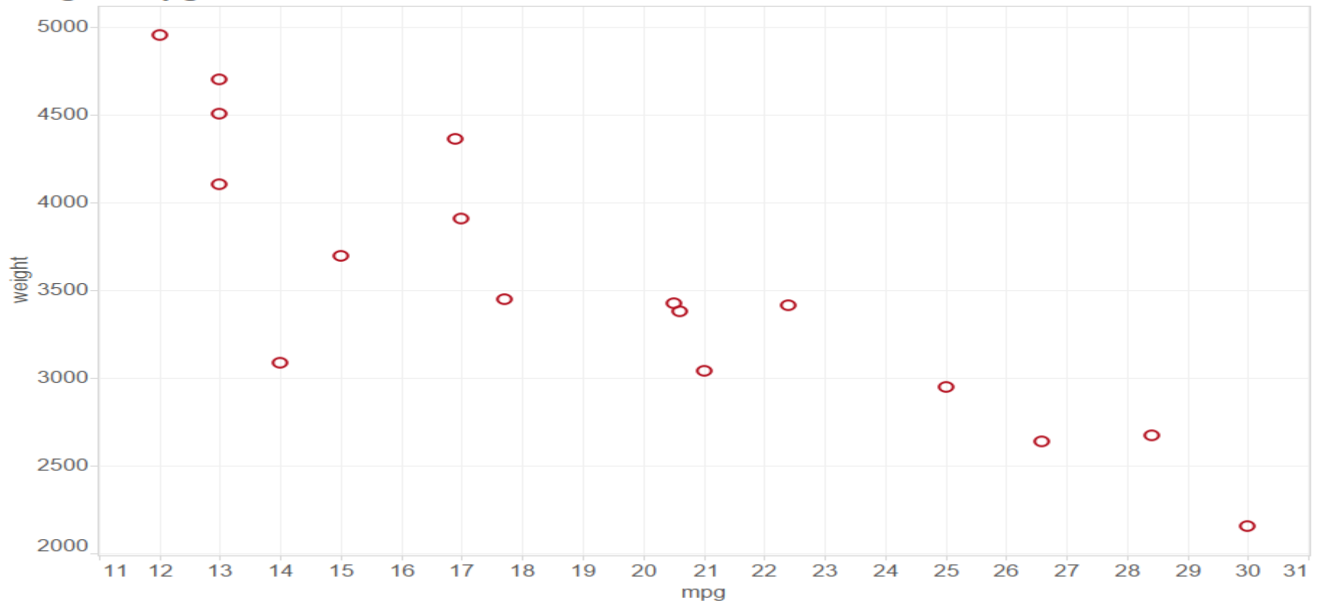
Mpg vs. weight. Details are shown for make. The view is filtered on make, which keeps ford.

weight_mpg_dodge



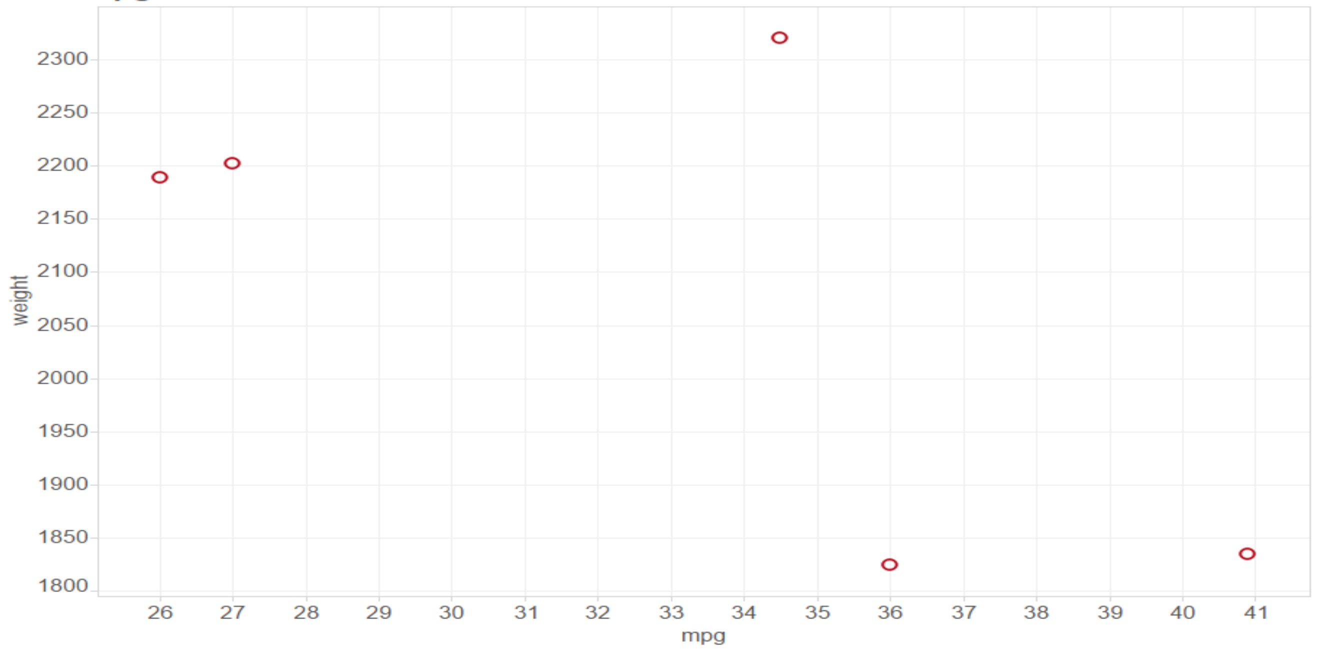
Mpg vs. weight. Details are shown for make. The view is filtered on make, which keeps dodge.

weight_mpg_buick



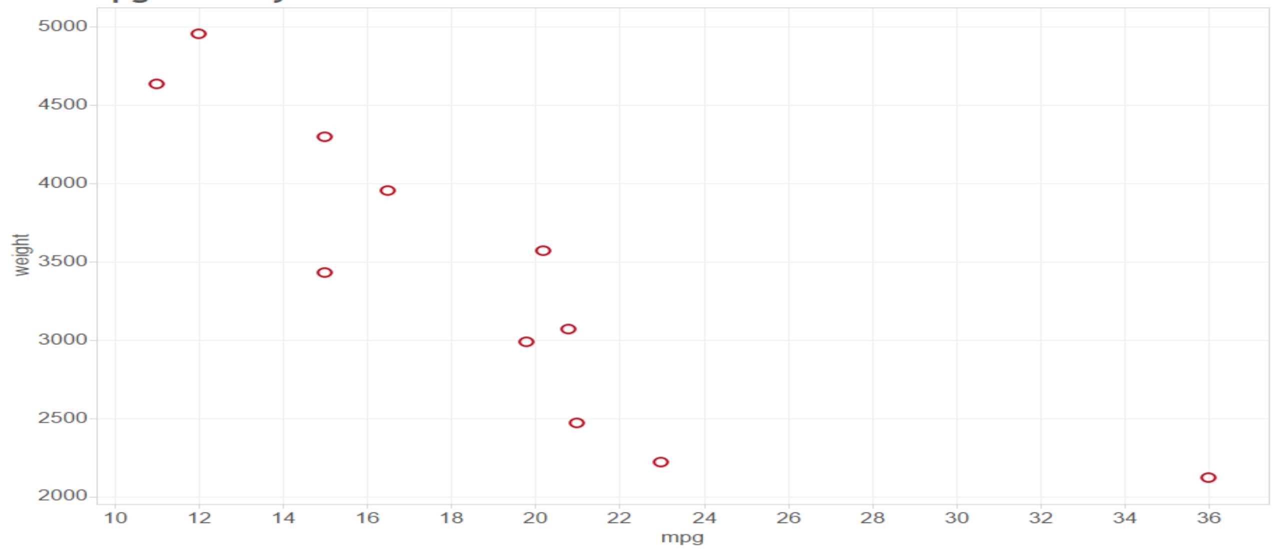
Mpg vs. weight. Details are shown for make. The view is filtered on make, which keeps buick.

w vs mpg renault



Mpg vs. weight. Details are shown for make. The view is filtered on make, which keeps renault.

w vs mpg mercury



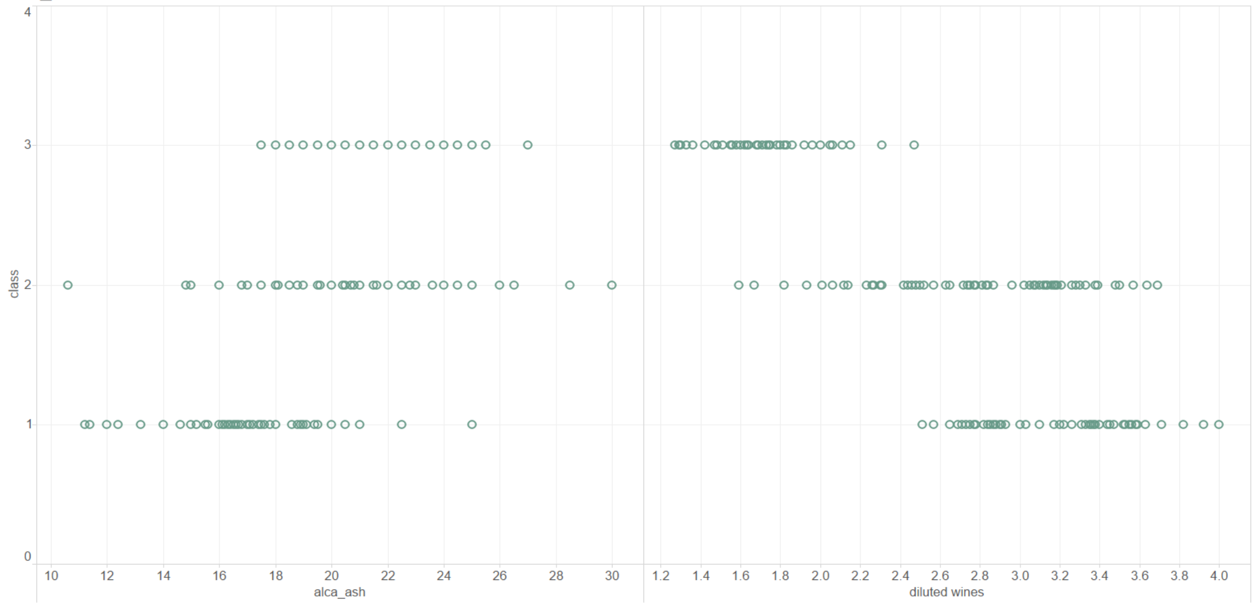
Mpg vs. weight. Details are shown for make. The view is filtered on make, which keeps mercury.

3 Patterns from the DataSet

Wine data set –

I plotted a chart to explore the automatic plot option of Tableau. I gave the features for the plot and the automatic chart option plotted the below graph. Amount of diluted wine decreases with class 1 to class 3. But alkaline ash content increases with Class 1 to class 3.

alca_ash vs class and dilutedWines vs class



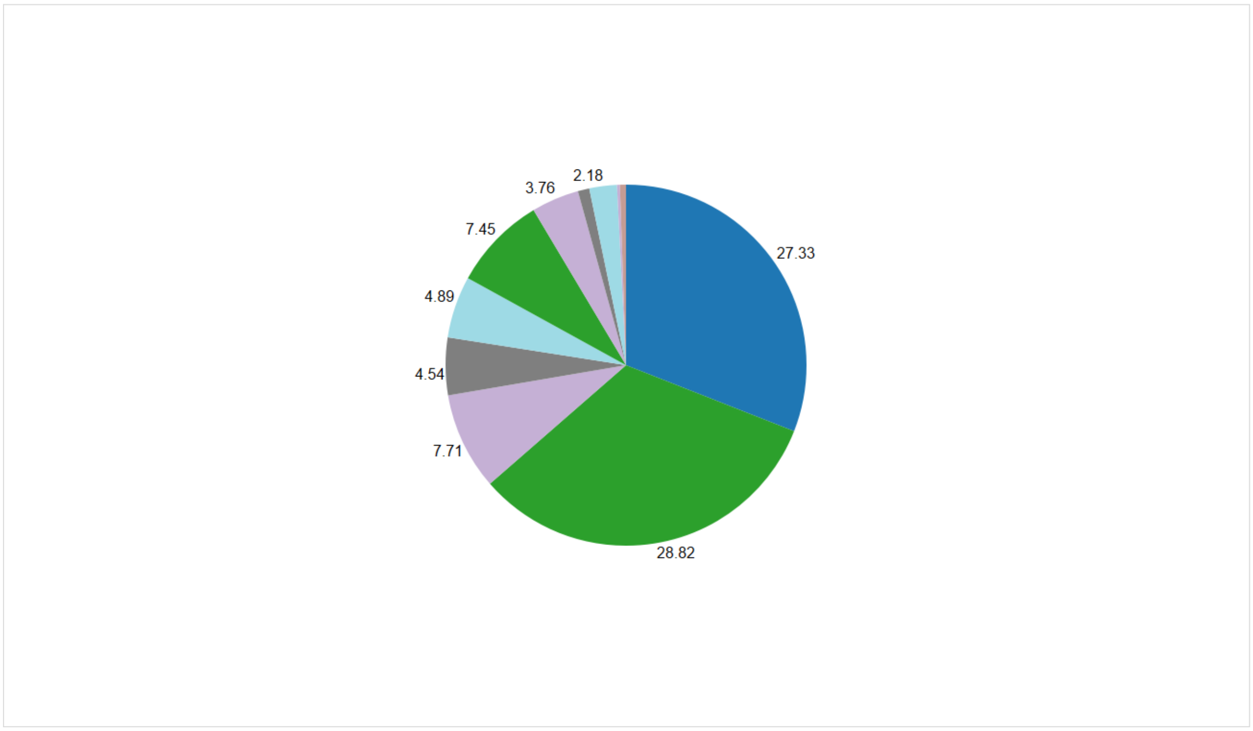
Alca_ash and diluted wines vs. class.

Diabetes data Set

I plotted a pie chart with the Diabetes data. I took age vs Diabetes pedigree to plot a pie chart.

The diabetes pedigree decreases as the age increases. After age 30 the pedigree quotient decreases gradually.

Age vs Diabetes Pedigree



Age (color) and sum of Diabetes pedigree function (size). The view is filtered on Age, which keeps 11 of 52 members.

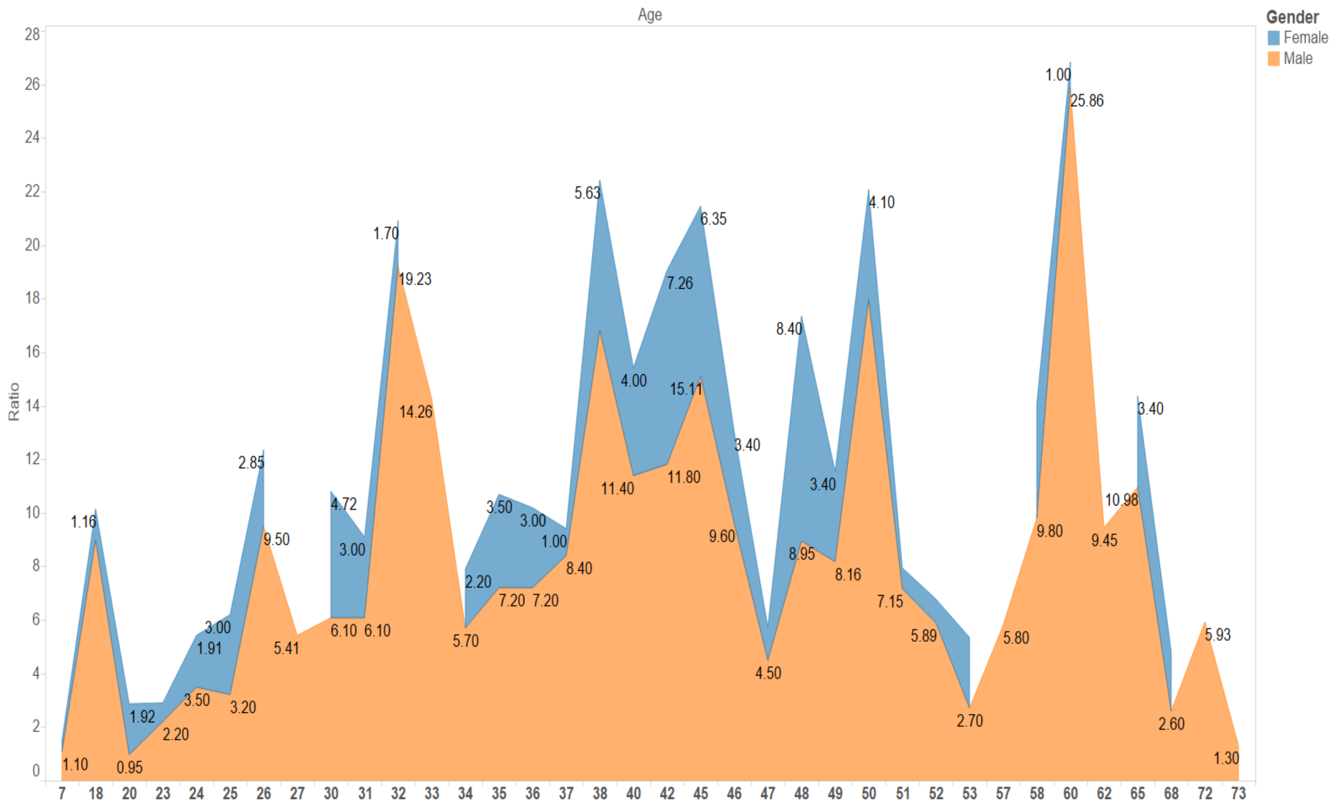
Diabetes pedigree function 88.29

Age
🟦 21
🟩 25
🟪 30
⬛ 35
🟦 40
🟩 45
🟪 50
⬛ 55
🟦 60
🟪 70
🟫 81

Indian Liver Patient Dataset

Albumin and Globulin which is the key factor for a diabetes patient can be visualized in this graph. The pattern here is that ratio is more in male in ages from 7-30 but the female ratio is also significantly visible in the range of 30 – 50 but then it decreases as the age increases beyond 51. You can observe that in the area graph below.

A/G RatioAlbumin and Globulin Ratio



Sum of Ratio for each Age. Color shows details about Gender. The view is filtered on Age, which keeps 36 of 72 members.

Tableau:

Tableau is a great tool to achieve visualizations for the datasets we have. With too many options in it, it feels like it's slightly complicated in the beginning, but after the introductory videos and reading about how to use several features of the tool it becomes very use to use. The great advantage is most of the desired operations for visualization is allowed. We can load data in all formats and export the output with labels. If we write a program for the same, it would take several hours and the independence to plot any type of charts is much higher with Tableau.

The user Interface is appealing and organized, with few clicks we can switch between charts. It can connect with multiple type of datasets.

Some of the features of Tableau seemed complex to me, I tried to read about using some of them but there was not enough support. May be with more experience using the tool, it might become more convenient. But was fun and easy plotting these charts and it saved a lot of time.

Question 4)

Used Gini and information gain to build the decision trees. Below are the output for 10 different datasets for both Gini and information gain as given by the user. You can run the 4.py file to see the output for all of these datasets. U have to choose the input and based on that the decision tree is built on Gini or Gain. The build tree method in 4.py is called recursively to build the tree. Once I get the best attribute I divide the data into half based on the median of the best attribute. This is done recursively until at a node all the data in that belongs to one single class. This will be treated as leaf node. To get the best attribute I use Gini or information gain value. To test the classifier, the test data will traverse through the decision tree and once a leaf node is reached I classify the value of that class to be the class of the record. I do this for all the records in the test data and with accuracy count for all of these I calculate accuracy for the decision tree.

I have attached another file Output4.docx which has values of accuracy at each fold of the data and the final accuracy for the dataset.

For some of the datasets where the range of the attribute values are very high, it is relatively difficult to train the classifier, hence I got lower accuracy for few datasets. But for datasets where range of the attributes are low, it's easier to build the decision tree and classification also improves resulting in high accuracy.

Gini and Gain does not really have much of a difference when it comes to splitting. They both most of time choose similar best attribute. The way these two are calculated are different though. For some datasets Gain continuously performed slightly better than Gini and it was the other way around for some datasets. I really could not distinguish a pattern where one of them won over the other. There was very small difference in accuracy of the results, so I

References

1) http://blog.bharatbhole.com/creating-boxplots-with-matplotlib/ for boxplots in python using the package matplotlib
2) http://www.onlamp.com/pub/a/python/2006/02/09/ai_decision_trees.html to understand the working of decision tree.