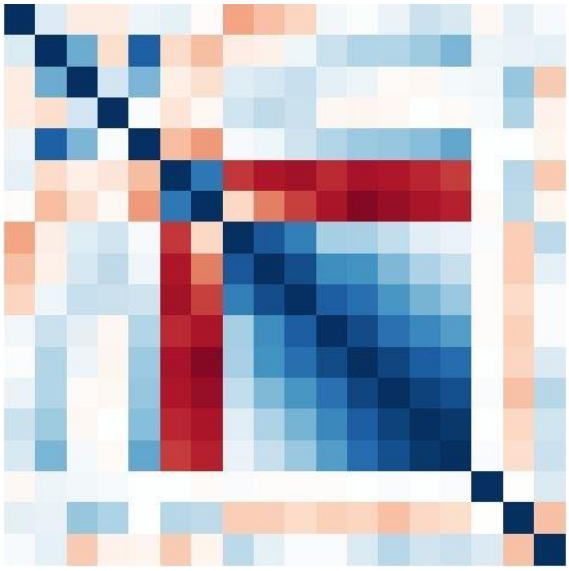
University of Nottingham
UK | CHINA | MALAYSIA

G54DMA: Coursework

SPRING 2019

TAHSINUR RAHMAN KHAN
STUDENT ID# 4335200



**Correlation matrix plot of plant dataset*

Contents

[B L A N K P A G E]

1. Description, Visualization and Pre-Processing

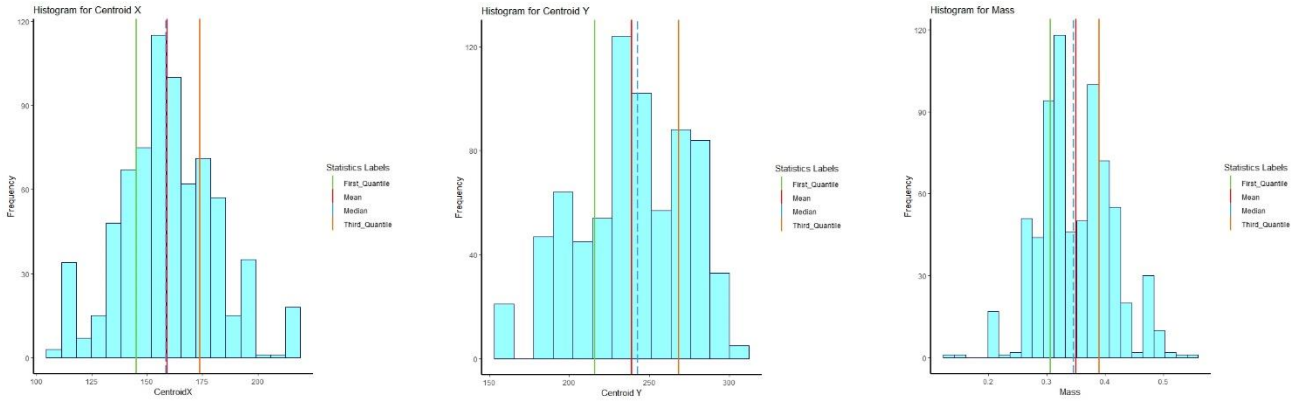
1.A Explore the data

1.A.i Table containing all attributes of the dataset including measures of centrality, dispersion, and how many missing values each attribute has

Attributes	Min.	First_Quart	Median	Mean	Third_Quart	Max.	Miss_Vals
CentroidX	110.3	145.2	158.4	159	173.9	218.2	NA
CentroidY	154.1	216	242.6	239.1	268.3	301.1	NA
Mass	0.1316	0.3065	0.3456	0.3497	0.3892	0.5484	6
Width	138.1	198.4	220.1	218.3	239.5	276.1	6
Depth	328.7	418.2	478.5	527.5	688.2	720.4	4
Orientation0	0.1501	0.2057	0.2492	0.2419	0.2774	0.3177	18
Orientation1	0.1493	0.1778	0.184	0.1836	0.1924	0.217	10
Orientation2	0.1051	0.1138	0.1247	0.1251	0.1338	0.1603	7
Orientation3	0.07216	0.07675	0.08213	0.08504	0.09051	0.12301	7
Orientation4	0.0575	0.06137	0.06546	0.06706	0.07127	0.08884	14
Orientation5	0.04324	0.04718	0.05038	0.05044	0.05186	0.0607	11
Orientation6	0.04375	0.04908	0.05174	0.05207	0.05452	0.06491	4
Orientation7	0.04372	0.05348	0.05505	0.05625	0.0597	0.07354	13
Orientation8	0.04538	0.0574	0.06171	0.06383	0.07005	0.09153	8
Orientation9	0.04941	0.06445	0.06993	0.07433	0.08198	0.1215	4
Leaf.weight	0.01	0.02	0.03	0.0282	0.04	0.05	411
LeafArea	6630	9207	10040	9837	10514	14395	13
Leaf.Hue	49.5	58.9	61.84	61.76	64.77	71.38	11

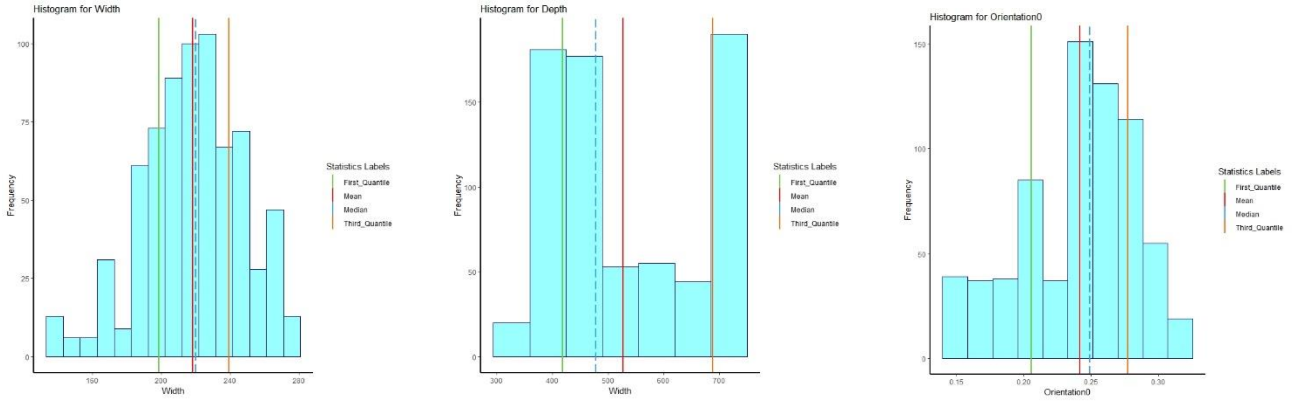
Table 1.1: Measures of centrality, dispersion and missing values for all the attributes in the original plant dataset

1.A.ii Histograms for each attribute



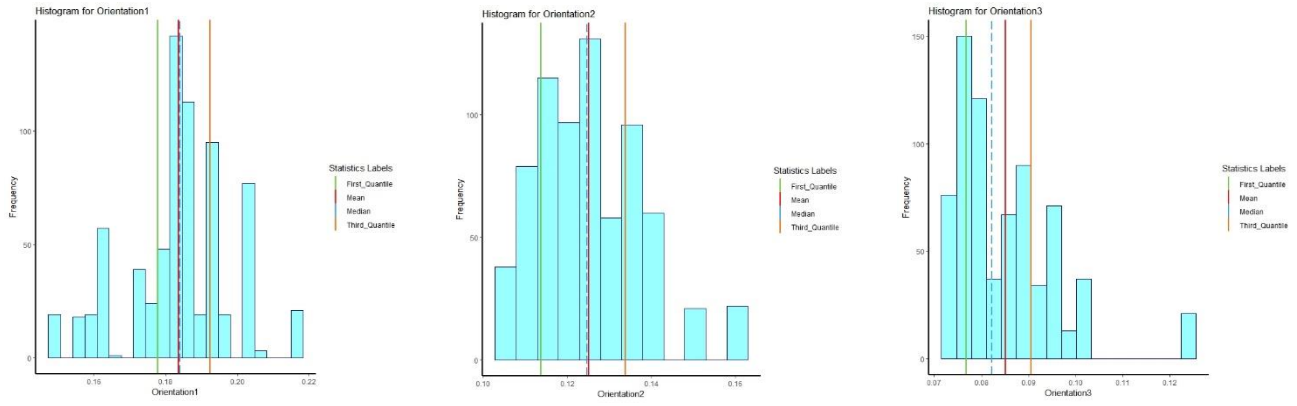
(a) (b) (c)

Fig 1.1: Histogram of Attributes: (a) CentroidX (b) CentroidY (c) Mass



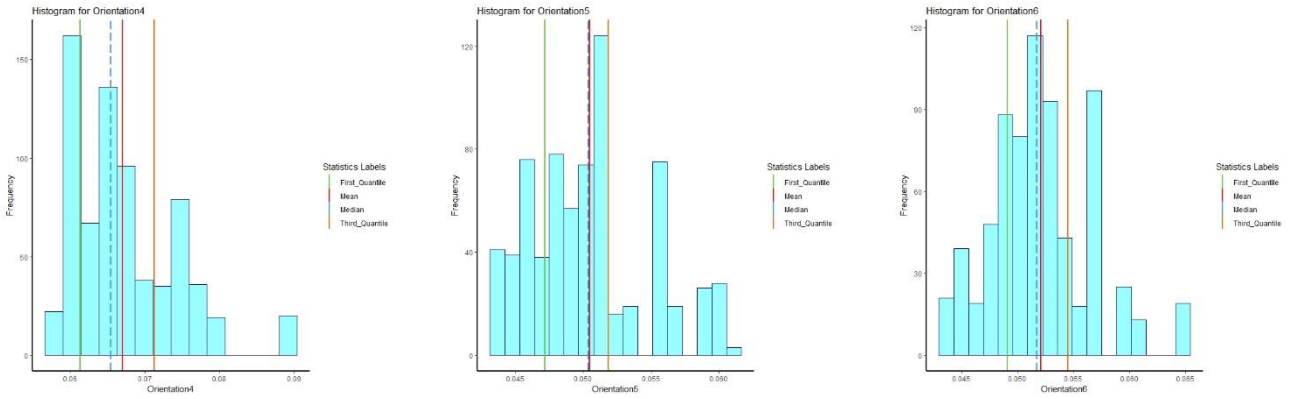
(a) (b) (c)

Figure 1.2: Histogram of Attributes: (a) Width (b) Depth (c) Orientation0



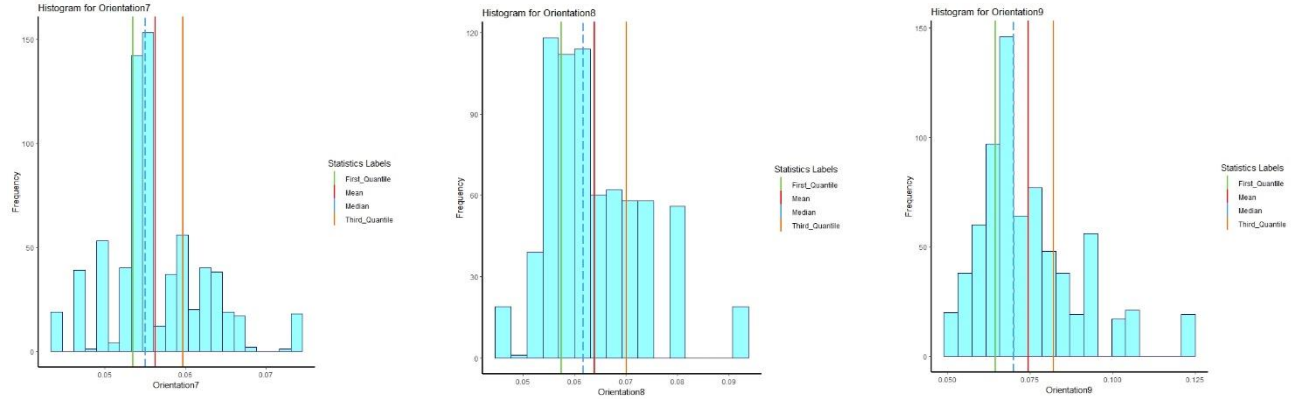
(a) (b) (c)

Fig 1.3 : Histogram of Attributes: (a) Orientation1 (b) Orientation2 (c) Orientation3



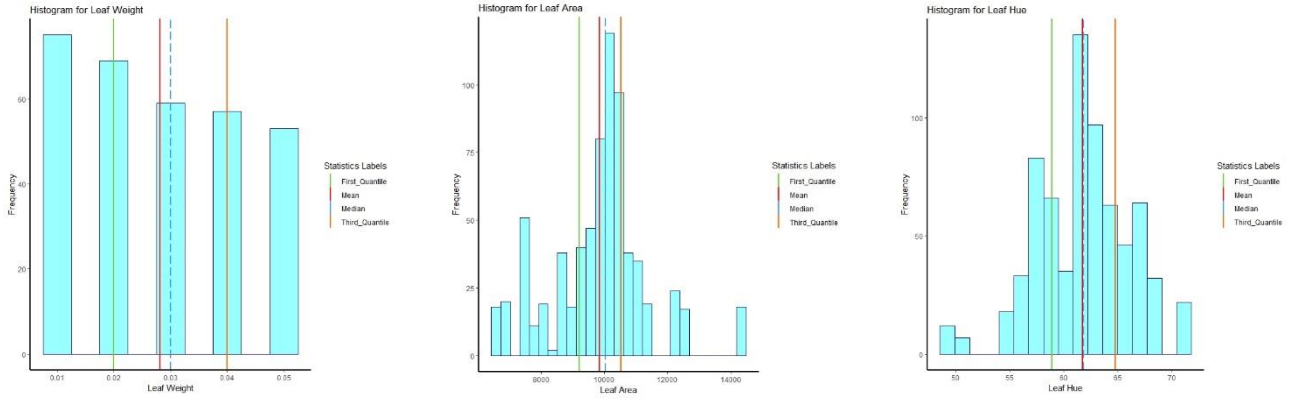
(a) (b) (c)

Figure 1.4: Histogram of Attributes: (a) Orientation4 (b) Orientation5 (c) Orientation6



(a) (b) (c)

Fig 1.5: Histogram of Attributes: (a) Orientation7 (b) Orientation8 (c) Orientation9



(a) 1.6: Histogram of Attributes: (a (b) Veigh (b) Leaf Area (c) Leaf Hue (c)

The histogram for each of each attribute was created using the Freedman-Diaconis rule for optimal bin width. The underlying functionality of the rule minimizes the integrated mean squared error of the histogram model of a true underlying density function [1]. In practice this has proved to be an effective and robust way of automatically calculating bin width for various data distributions [2].

Since, the data distributions of the plant dataset vary greatly for each, hence this model was employed to produce valid binning of different attribute instances.

The model follows the formula:

$$h = 2 * IQR * n^{-\frac{1}{3}}$$

$$Number\ of\ bins = \frac{max. - min.}{h},$$

$Where,$

$h: bin - width$

$IQR: Interquantile\ range$

$n: total\ number\ of\ instances$

$max.: maximum\ value\ of\ an\ attribute\ column$

$min.: minimum\ value\ of\ an\ attribute\ column$

The histogram for the first three attributes – CentroidX, CentroidY and Mass reveal a somewhat normal distributions (fig 1.1: (a), (b), (c)). The close alignment of mean and median lines in the histogram models of these attributes confirm this assumption. Moreover, these attributes contain very few missing attributes and therefore, the entire range of binning spectrum are populated by data points.

The histogram for Width attribute also display a normal distribution (close aligning of mean and median lines), although the entire distribution is a bit skewed to the right. With presence of large interquantile range and maximum peaks on both sides of a low trough of data points, the histogram for Depth do not follow any particular trend and the data are bound within a few discreet bins in the 300 to 700 range. The histogram for Orientation0 is also a bit skewed to the right, with lesser alignment between the mean and median.

Histogram models for Orientation1 to Orientation9 reveal multiple bins being empty in between bin peaks. Presence of missing values in each of those attributes and generally sparse data distribution on the right mean these attributes do not follow any particular distribution pattern and show a tendency of left skewedness.

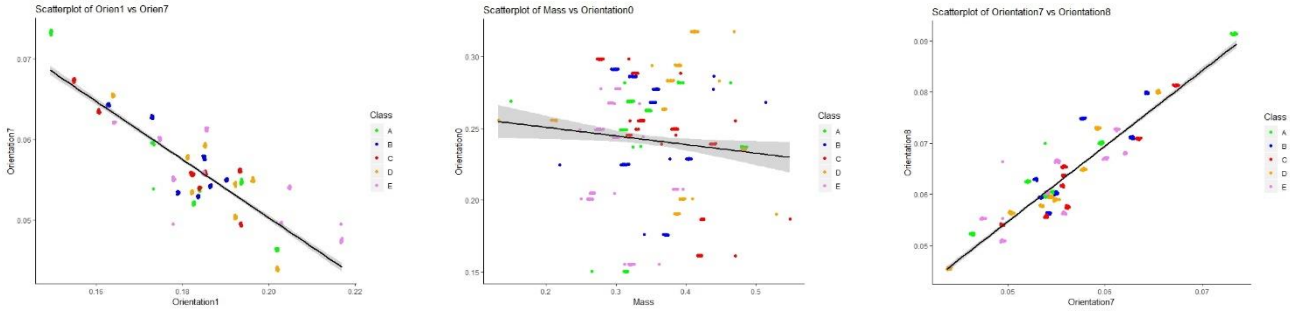
The Leaf Weight attribute contained the most number of missing values, this is also evident from the histogram model itself. Each bin are distinctly sperate from one another, with empty bins in between data peaks. The bin peaks also progressively decreases from left to the right, meaning more instances have lighter leaf weights.

The Leaf Area present a somewhat normal distribution within the interquantile range. However, the distribution loses meaning outside the IQR due to the presence of outlier peaks and missing bins. The histogram model for Leaf Hue distribution reveal a bit right skewedness.

1.B Relationships between Attributes, and between the Class and Attributes

1.B.i Calculating Correlations between attributes:

- Correlation between **Orientation1** and **Orientation7**: -**0.8696721**
- Correlation between **Mass** and **Orientation0**: **-0.08768968**
- Correlation between **Orientation7** and **Orientation8**: **0.9291126**



(a) (b) (c)

Fig 1.7: Scatterplots of variables vs. Class: (a) Orientation7 (b) Orientation8 (c) Orientation9

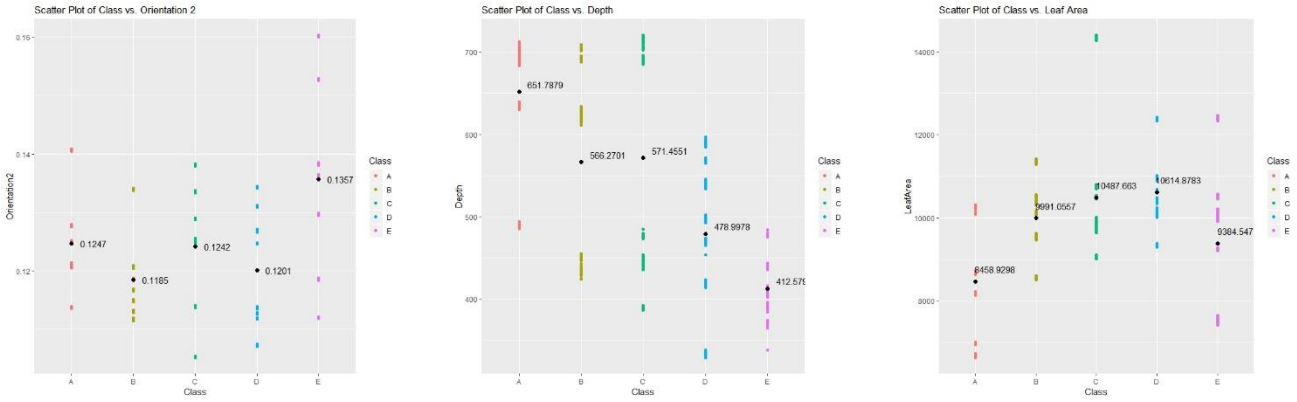
1.B.ii Two most Correlated Variables:

- **Orientation8** and **Orientation9**. The value of their correlation coefficient is **0.960551**. The strong positive correlation between Orientation 8 and 9 indicates that *Orientation* attributes might be closely related group of variables, whose values might also be interdependent in physical space.

1.B.iii Two least Correlated Variables:

- Two least correlated variables: **Leaf Area** and **Depth**. The value of their correlation coefficient is **0.000519**. The weak (positive) correlation between these two variables might indicate that variance in Leaf Area data has no effect to the Depth. In physical space this might translate to the two variables belonging to separate groups that are not interdependent on each other.

1.B.iv Scatterplots between Class Variable and Attributes



(a) (b) (c)

Fig 1.8: Histogram of Attributes: (a) Orientation7 (b) Orientation8 (c) Orientation9

From the scatterplots we can deduce that Orientation2 attribute has relatively high association with Class B and quite sparse relationship with Class E. Depth has quite high association with Class E and less so with other classes. And Leaf Area follows a general pattern of association with all the classes. It can be concluded that higher the association between an attribute and a Class value, the more information will be held by that attribute about that Class value.

1.C General Conclusion

While discussing the importance of attributes, several factors had to be taken into consideration such as the descriptive statistics, visualizations through histogram models, presence of missing values and correlations. Fig 1.9 below will summarize the characteristics of each attribute into a concise format by following conventions discussed below for each of the factors.

Descriptive statistics: a set of statistical values that describe a variable, including the skewness and uniformity of the underlying data distribution and presence of outliers. Values included in the descriptive statistics discussion include the mean, median and IQR (Inter Quantile Range). Data distribution is considered symmetric and uniform if mean $\simeq$ median [3]. Whereas if mean < median, then the distribution is considered to be skewed to the right, and vice versa if the data distribution is skewed left.

Visualizations: The histogram figures produced in Part-1 A would be used for discussing the orientation of the data. Attributes displaying normal distribution will have a bell shaped curve with no missing bins and the mean and median would be located along the highest peak. The skewness of the data distribution and presence of outliers will be determined from the histogram visualizations.

Presence of missing values: Table 1 contains a column of the missing values of each attribute. The greater the number of missing values in an attribute, lesser the information it holds about the dataset.

The number of missing values for each attribute are categorized into three categories: Low (< 10 instances), Medium (≥ 10), and High (> 100)

Correlation: High positive or negative correlation between pairs of attributes will indicate that either of them could be dropped from the dataset while preserving information on the dataset.

The Correlation column contains a list of attributes that has either a high positive or negative correlation with any particular attribute. Any correlation coefficient with a magnitude > 0.7 is considered high.

Attribute	Descriptive Statistics (Skewness)			Visualizations		Presence of Missing values			Correlation		
	Left	No Skew / Symmetric	Right	Shape	Missing Bins	Low	Medium	High	Positive	Negative	No Significant Correlation
CentroidX		✓		Normal Distribution	No missing bins						✓
CentroidY	-			Normal Distribution	Missing bin on left tail might indicate some outlier				Depth		
Mass		Close to no skew		Normal Distribution	Missing bin on left tail might indicate some outlier						✓
Width	✓			Somewhat normal around the IQR	No missing bins						✓
Depth			✓	-	No missing bins				CentroidY		
Orientation0	✓			-	No missing bins				Orientation1	Orientation2, Orientation3,Orientation4,Orientation5,Orientation6	
Orientation1		Close to no skew		Somewhat normal around the IQR	Missing bins on left and right tails might indicate outliers				Orientation0	Orientation5,Orientation6,Orientation7,Orientation8,Orientation9	
Orientation2			✓	-	Missing bins on right tail might indicate some outliers				Orientation3	Orientation0,	
Orientation3			✓	-	Multiple missing bins on right tail might indicate outliers				Orientation2, Orientation4	Orientation0,	
Orientation4			✓	-	Multiple missing bins on right tail might indicate outliers				Orientation3,Orientation5,Orientation6.	Orientation0	
Orientation5		Close to no skew		-	Multiple missing bins on right tail might indicate outliers				Orientation4,Orientation6,Orientation7	Orientation0,Orientation1	
Orientation6			✓	-	Multiple missing bins on right tail might indicate outliers				Orientation4,Orientation5,Orientation7,Orientation8,Orientation9	Orientation0, Orientation1	
Orientation7			✓	-	Missing bins on left and right tails might indicate outliers				Orientation5,Orientation6,Orientation8,Orientation9	Orientation0, Orientation1	
Orientation8			✓	-	Multiple missing bins on right tail might indicate outliers				Orientation6,Orientation7,Orientation9	Orientation0, Orientation1	
Orientation9			✓	-	Multiple missing bins on right tail might indicate outliers				Orientation6,Orientation7,Orientation8	Orientation1	
Leaf.weight	✓			-	Multiple missing bins and separation of data into discrete bins indicate lots of missing values.						✓
CentroidX			✓	Somewhat normal around the IQR	Multiple missing bins on right and left tails. Might indicate outliers on both sides						✓
CentroidY		Close to no skew		-	Missing bins on both left and right tail might indicate presence of outliers						✓

Fig 1.9: Summarizing all aspects of the attributes in the dataset

1.D Dealing with Missing Values in R

1.D.i Replace missing values with Zero, Mean, Median:

For this part, script was written for replacing missing values in each attribute of the dataset using their mean, median. For brevity, only first 20 instances of two attributes – LeafArea and Leaf.Hue, are considered for illustration purposes.[1]

From table 1.1, it can be seen that LeafArea had a mean value of 10040 and median value of 9837; Leaf.Hue had a mean value of 61.84 and median value of 61.76. It can be seen that the missing values in – instances 13 to 15 of LeafArea and instances 15 to 18 of Leaf.Hue, from original dataset, had been replaced by zero (Table 1.3), mean (Table 1.4) and median (Table 1.5).

Sample_ID	LeafArea	Leaf Hue	Class
1	10257	61.02	A
2	6977	59.61	A
3	10115	64.96	A
4	8683	57.82	A
5	8171	55.64	A
6	6663	65.67	A
7	10277.94	61.10878	A
8	6952.952	59.48862	A
9	10092.69	65.04219	A
10	8704.824	57.82033	A
11	8171.375	55.56148	A
12	6645.046	65.51021	A
13		61.1849	A
14		60.14894	A
15			A
16	8677.927		A
17	8177.976		A
18	6638.353		A
19	10257.11	61.15289	A

Table 1.2: Original dataset containing missing values

Sample_ID	LeafArea	Leaf.Hue	Class
1	10257	61.02	A
2	6977	59.61	A
3	10115	64.96	A
4	8683	57.82	A
5	8171	55.64	A
6	6663	65.67	A
7	10277.94	61.10878	A
8	6952.952	59.48862	A
9	10092.69	65.04219	A
10	8704.824	57.82033	A
11	8171.375	55.56148	A
12	6645.046	65.51021	A
13	0	61.1849	A
14	0	60.14894	A
15	0	0	A
16	8677.927	0	A
17	8177.976	0	A
18	6638.353	0	A
19	10257.11	61.15289	A

Table 1.3: Zero replaced dataset

[1] Code provided in the Appendix

Sample_ID	LeafArea	Leaf.Hue	Class
1	10257	61.02	A
2	6977	59.61	A
3	10115	64.96	A
4	8683	57.82	A
5	8171	55.64	A
6	6663	65.67	A
7	10277.94	61.10878	A
8	6952.952	59.48862	A
9	10092.69	65.04219	A
10	8704.824	57.82033	A
11	8171.375	55.56148	A
12	6645.046	65.51021	A
13	10039.51	61.1849	A
14	10039.51	60.14894	A
15	10039.51	61.83792	A
16	8677.927	61.83792	A
17	8177.976	61.83792	A
18	6638.353	61.83792	A
19	10257.11	61.15289	A

Table 1.4: Mean replaced Dataset

Sample_ID	LeafArea	Leaf.Hue	Class
1	10257	61.02	A
2	6977	59.61	A
3	10115	64.96	A
4	8683	57.82	A
5	8171	55.64	A
6	6663	65.67	A
7	10277.94	61.10878	A
8	6952.952	59.48862	A
9	10092.69	65.04219	A
10	8704.824	57.82033	A
11	8171.375	55.56148	A
12	6645.046	65.51021	A
13	9836.62	61.1849	A
14	9836.62	60.14894	A
15	9836.62	61.76061	A
16	8677.927	61.76061	A
17	8177.976	61.76061	A
18	6638.353	61.76061	A
19	10257.11	61.15289	A

Table 1.5: Median replaced Dataset

1.D.ii Compare Contrast the Approaches

For comparison purposes, it would be interesting to know how the statistical values would change for each of zero, mean and median replaced dataset. Therefore, an attribute had to be chosen that would be affected the most by the modification. Hence Leaf.weight was chosen since it had the most number of missing values.

Dataset	Minimum	First_Quantile	Median	Mean	Third_Quantile	Maximum
Original	0.01	0.02	0.03	0.0282	0.04	0.05
Zero Replaced	0	0	0	0.0122	0.02	0.05
Mean Replaced	0.01	0.02821	0.02821	0.02821	0.02821	0.05
Median Replaced	0.01	0.03	0.03	0.02923	0.03	0.05

Table 1.7: Comparing the effects of zero, mean, median imputation on the dataset

From table 1.7, it could be seen that, all the statistical quantities – mean, median, first quantile, third quantile, changed for Leaf weight attribute when the missing values were replaced.

Illustrations of these changes can be clearly seen in the boxplots below (Fig 1.10 – 1.13). The mean value and median levels are indicated in the plots.

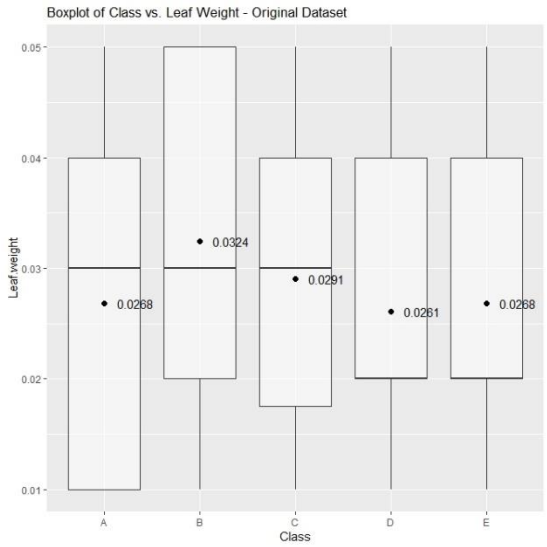


Fig 1.10: Boxplot of Leaf Weight vs. Class from Original Dataset

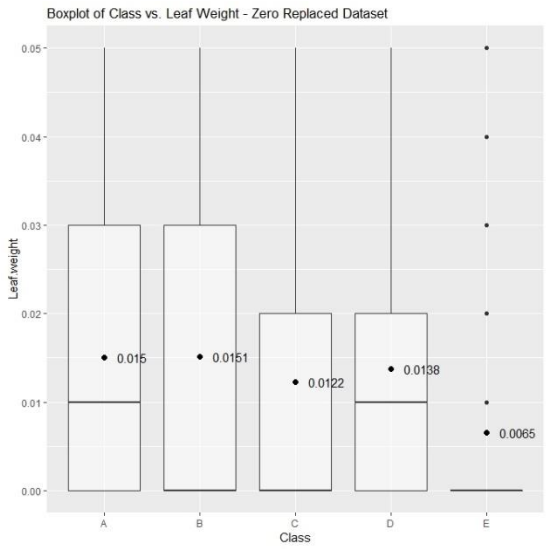


Fig 1.11: Boxplot of Leaf Weight vs. Class from Zero replaced Dataset

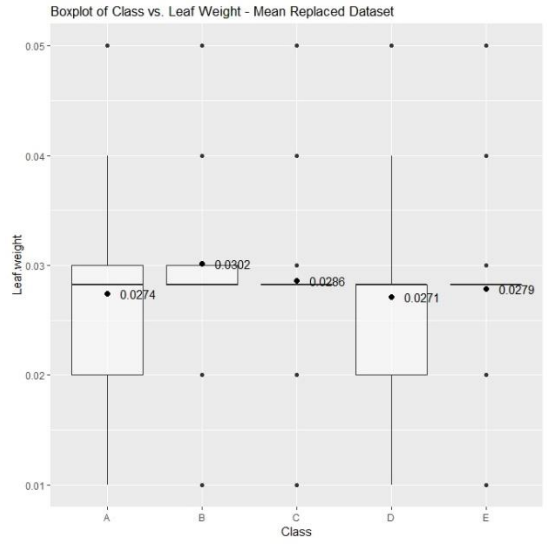


Fig 1.12: Boxplot of Leaf Weight vs. Class from Mean replaced Dataset

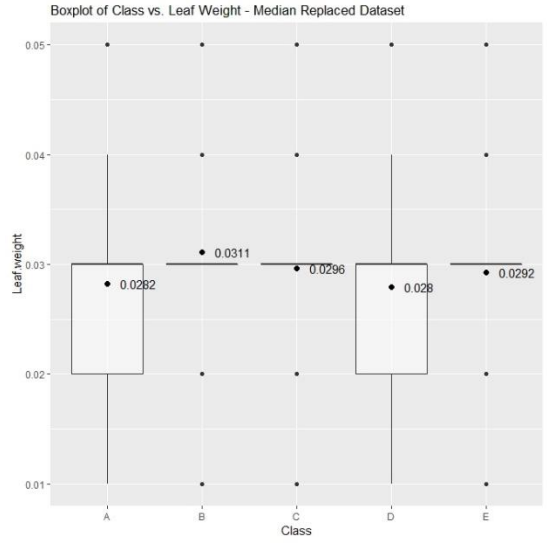


Fig 1.13: Boxplot of Leaf Weight vs. Class from Median replaced Dataset

1.E Attribute Transformation

Three type of transformations – mean centering, normalization and standardization were performed on each of the datasets that were generated in part 1.D. In total nine datasets were generated in this part. [2]

The formulas for each are given below:

Normalization*:*

$$y_i = \frac{x_i - \min(x_1 \dots x_n)}{\max(x_1 \dots x_n) - \min(x_1 \dots x_n)}$$

$Where,$
$y_i: normalized\ value$
$x_i: instances$

Mean Centering:

$$y_i = x_i - \bar{x}$$

$Where,$
$y_i: mean\ centered\ value$
$x_i: instances$
$\bar{x}: mean\ value\ of\ attribute\ column$

Standardization:

$$y_i = \frac{x_i - \bar{x}}{\sigma}$$

$Where,$
$y_i : standarized\ value$
$x_i : instances$
$\bar{x}: mean\ value\ of\ attribute\ column$
$\sigma : standard\ deviation\ of\ attr.col.$

To compare the effects of each transformation technique, Leaf Weight attribute was again considered. The table 1.8 shows how the statistical values of Leaf Weight changed for each transformation.

[2] Code for the transformation functions and datasets are provided in the appendix

Dataset	Minimum	First_Quantile	Median	Mean	Third_Quantile	Maximum
Original	0.01	0.02	0.03	0.0282	0.04	0.05
Mean.Cen Zero Replaced	-0.0122	-0.0122	-0.0122	0	0.007804	0.037804
Mean.Cen Mean Replaced	-0.01821	0	0	0	0	0.02179
Mean.Cen Median Replaced	-0.01923	0.000774	0.000774	0	0.000774	0.020774
NORM Zero Replaced	0	0	0	0.2439	0.4	1
NORM Mean Replaced	0	0.4553	0.4553	0.4553	0.4553	1
NORM Median Replaced	0	0.5	0.5	0.4807	0.5	1
STD Zero Replaced	-0.7257	-0.7257	-0.7257	0	0.4643	2.2494
STD Mean Replaced	-1.954	0	0	0	0	2.338
STD Median Replaced	-2.05371	0.08262	0.08262	0	0.08262	2.21895

Table 1.8: Comparing the effects of mean centering, normalization and standardization

The following effects could be observed by performing attribute transformation on all nine datasets:

- Normalization on the datasets caused the minimum, the statistical values and the maximum to be range bound between zero to one.
- Mean centering caused all the values to decrease from their original value and gravitate more towards zero.
- Performing standardization on all datasets caused the values to scale up by a factor that is roughly equal to the standard deviation of the LeafWeight attribute column in each of the zero, mean and median replaced datasets.

1.F Attribute/Instance selection and PCA

1.F.i Deleting Attribute and Instances

Although there are no hard and fast rules on exact threshold for allowing missing data, it is generally well considered in practice to remove or replace an attribute or an instance that has more than 50% of missing values [4]. Otherwise, missing data can lead to the dataset being *unsuitable for a statistical procedure and the statistical analyses vulnerable to violations of assumptions* [5] For this part, we decided to eliminate all attributes and instances that had more than 50% of missing values. [3]

[3] Codes for: attribute/instance deletion and resulting datasets are provided in the Appendix

After passing the original code through the instance and attribute deletion functions, it could be observed that the Leaf.Weight attribute had been deleted, however none of the attribute were deleted. This result was in line with the previous results, Leaf Weight attribute had 411 missing values, meaning around 57% of its values were missing. The following table shows the resulting dataset after deleting the Leafweight attribute.

Sample_ID	CentroidX	CentroidY	Mass	Width	Depth	Orientation0	Orientation1	Orientation2	Orientation3	Orientation4	Orientation5	Orientation6	Orientation7	Orientation8	Orientation9	LeafArea	Leaf.Hue	Class
1	150.8999	256.3507	0.480334	226	694	0.237142056	0.183958428	0.140673395	0.092410192	0.065385792	0.043597301	0.049197806	0.053838984	0.059633725	0.07416232	10257	61.02	A
2	174.238	244.7633	0.344877	169	491	0.262608173	0.182623489	0.121182557	0.078435691	0.064986253	0.051380247	0.050377344	0.052079114	0.062546759	0.073780372	6977	59.61	A
3	158.415	236.9011	0.389815	264	635	0.281925739	0.20189779	0.127667449	0.079251709	0.062486765	0.044692692	0.04404489	0.046334115	0.052259434	0.059439416	10115	64.96	A
4	132.8219	278.9971	NA	202	707	0.249087598	0.173148721	0.120615839	0.076651113	0.064367446	0.043697661	0.053189039	0.053834261	0.070027226	0.082825179	8683	57.82	A
5	169.5172	247.9947	0.319894	221	699	0.268834017	0.193667428	0.113706349	0.077774388	0.061235928	0.04931211	0.050382427	0.054698585	0.06045293	0.069935837	8171	55.64	A
6	159.2907	271.4112	0.313119	142	689	0.150293737	0.14944887	0.125038532	NA	0.074255738	0.06048503	0.064685861	0.073294347	0.091364102	0.121333736	6663	65.67	A
7	151.1185	256.7095	0.3347	226.7215	695.6081	0.237323351	0.183795106	0.140829645	0.09229251	0.065355732	0.043697661	0.048993423	0.053834261	0.059604401	0.074084165	10277.944	61.10878	A
8	174.4604	244.464	0.3464	171.5732	491.7307	0.262554374	0.1825432	0.121121025	0.078587144	0.064811113	0.051106141	0.050208941	0.052049219	0.062463868	0.0739398	6952.9519	59.48862	A
9	157.992	236.7021	0.4636	266.5965	637.1409	0.281993381	0.202051249	0.127666594	0.079154368	0.062538864	0.044899106	0.044004946	0.04646434	0.052260815	0.059339535	10092.695	65.04219	A
10	132.9763	279.3778	0.3293	203.747	712.3531	0.249168216	0.173198595	0.120672301	0.076664272	0.064270105	0.050549775	0.053021754	0.059579486	0.069884183	0.082730328	8704.8239	57.82033	A

Table 1.9: First 10 instances of the dataset after removing attributes and instances with more than 50% missing values

After deleting the Leafweight attribute, the dataset was analyzed to find out exactly how many instances remained that had at least one missing value. 93 attributes had only one missing value, 17 instances had only two missing values and only three instances had three missing values – the highest number of missing values for any instances in the entire dataset.

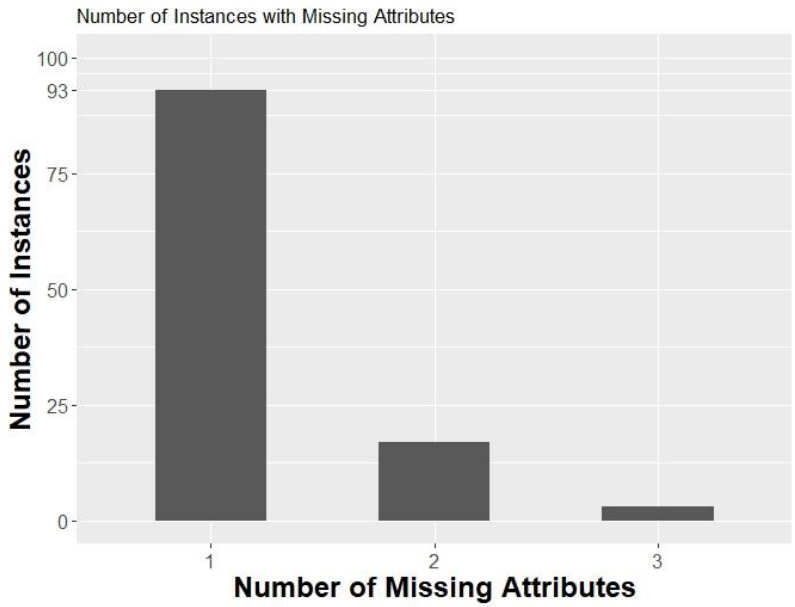


Fig 1.14: Number of instances with missing values

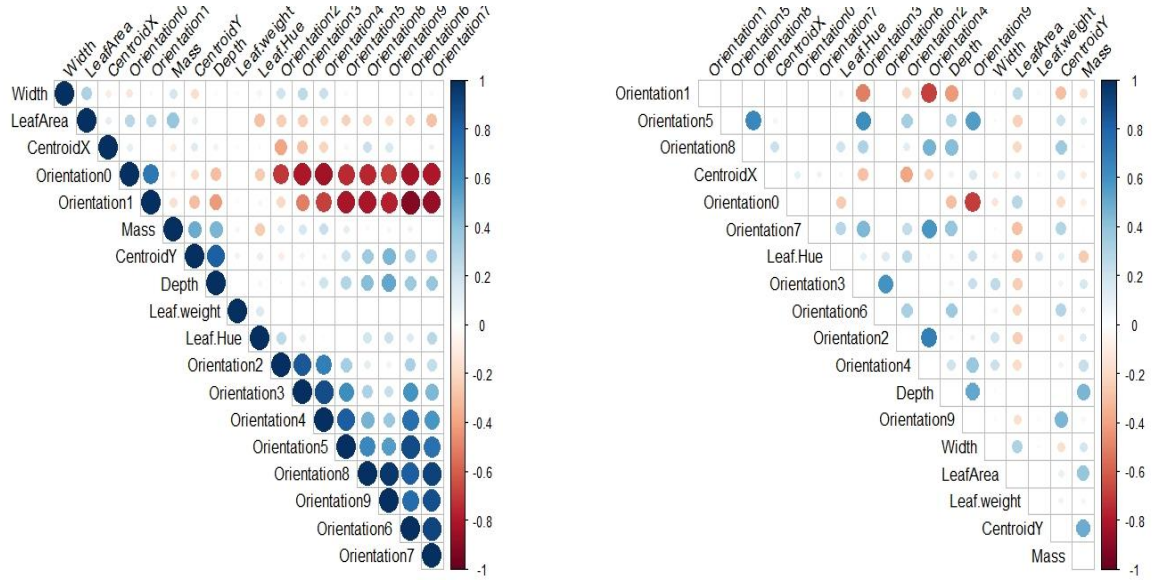
For the missing values that remained in the dataset, **mean values of the corresponding attribute columns were imputed**. This particular strategy was followed because:

- Only a small proportion of missing values remained after deletion of Leadweight attribute [6]
- It was assumed that the values were missing completed at Random (MCAR) [6]
- The missing values had to relation to other values in the dataset. [6]

1.F.ii Reducing dataset to contain only uncorrelated attributes and no missing values

Magnitude of correlation coefficient threshold for selecting correlated attributes was set at 0.7. Although this meant only selecting highly correlated values [7], this value was chosen since the entire dataset was highly correlated and to minimize the loss of information that might result from deletion of lots of instances.

The following figures represent the correlation matrix of the dataset in graphical format.



(a) (b)

Fig 1.15: (a) Correlation matrix of the original dataset. (b) Correlation matrix after deleting highly correlated attributes. Colored dots pertaining to only non-highly correlated is remaining in the matrix. The color scale on the side indicates the relative correlation coefficient values – red dots indicate negative correlation and blue dots indicate positive correlation.

In the table below, attributes (from row and column headers) corresponding to each cell with a dot represent the attribute pairs with correlation coefficient values of > 0.7. [4]

	CentroidX	CentroidY	Mass	Width	Depth	Orientation0	Orientation1	Orientation2	Orientation3	Orientation4	Orientation5	Orientation6	Orientation7	Orientation8	Orientation9	Leaf.weight	LeafArea	Leaf.Hue
CentroidX																		
CentroidY					•													
Mass																		
Width																		
Depth																		
Orientation0							•	•	•	•	•	•	•	•				
Orientation1											•	•	•	•	•			
Orientation2									•									
Orientation3										•								
Orientation4											•	•						
Orientation5												•	•					
Orientation6													•	•	•			
Orientation7														•	•			
Orientation8															•			
Orientation9																		
Leaf.weight																		
LeafArea																		
Leaf.Hue																		

Table 1.10: Attribute pairs with correlation coefficient > 0.7. Attributes corresponding to each cell with a dot are labelled as highly correlated

The following conventions were followed while deleting attributes:

- For a highly correlated pair of attributes, attribute with higher number of missing values was deleted.
- If a pair of attributes had the same number of missing values, then whichever attribute had more number of high correlation pairing, with other attributes, was deleted.

This resulted in a dataset where **Depth**, **Orientation0**, **Orientation1**, **Orientation2**, **Orientation4** , **Orientation6**, **Orientation7** and **Orientation8** were dropped. Missing values that remained in the dataset were imputed by column means.

Sample_ID	CentroidX	CentroidY	Mass	Width	Orientation3	Orientation9	LeafArea	Leaf.Hue	Class
1	150.89985	256.3507449	0.480333761	226	0.092410192	0.07416232	10257	61.02	A
2	174.2380221	244.7632742	0.34487731	169	0.078435691	0.073780372	6977	59.61	A
3	158.4149628	236.9010665	0.389815258	264	0.079251709	0.059439416	10115	64.96	A
4	132.8219106	278.9971395	0.349743618	202	0.076651113	0.082825179	8683	57.82	A
5	169.5171671	247.9946612	0.319893931	221	0.077774388	0.069935837	8171	55.64	A
6	159.2906973	271.4112128	0.313119258	142	0.085036162	0.121333736	6663	65.67	A
7	151.1185	256.7095	0.3347	226.7214678	0.09229251	0.074084165	10277.94402	61.10877879	A
8	174.4604	244.464	0.3464	171.5732327	0.078587144	0.0739398	6952.951853	59.48861913	A
9	157.992	236.7021	0.4636	266.5964501	0.079154368	0.059339535	10092.69482	65.04218511	A
10	132.9763	279.3778	0.3293	203.747013	0.076664272	0.082730328	8704.823872	57.82033084	A

Table 1.11: Dataset after deleting all highly correlated attributes

[4] This table has been processed in excel for representation purposes. The original excel file generated contained the names of the attributes from the column headers instead of the dots. Code attached in Appendix.

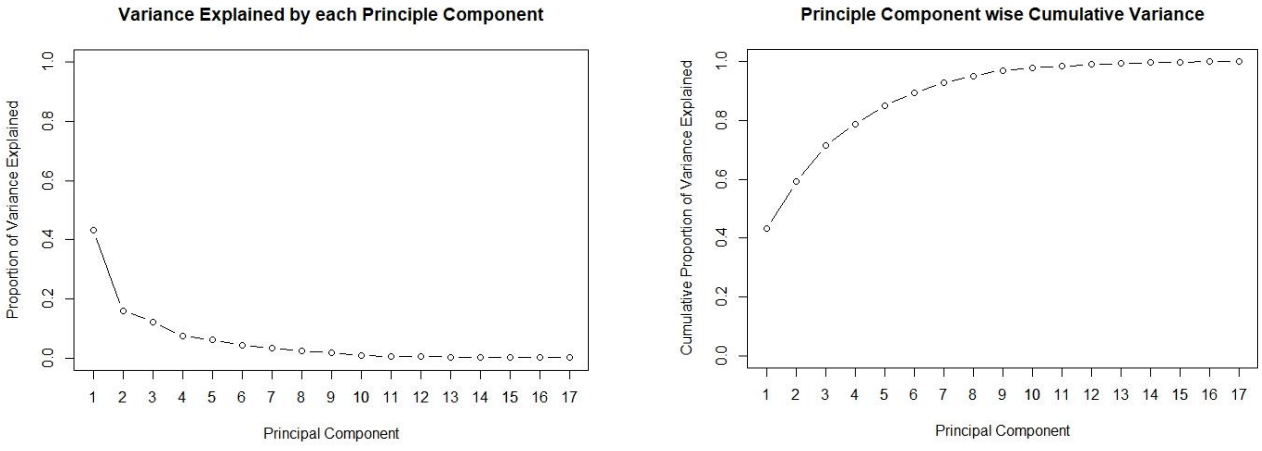
1.F.iii Principal Component Analysis in R with 7 components

The original dataset had to be first cleaned from missing values and standardized before passing it on for principal component Analysis. At first all attributes and instances containing more than 50% of missing values were removed. Any remaining missing value was imputed using column means. The resulting dataset was then standardized since principal component analysis can only be done on standardized values [8]. The *prcomp* function was used for performing PCA in R. Only the first seven principal components were extracted out from the resulting operation, as shown in table 1.12.

	PC1	PC2	PC3	PC4	PC5	PC6	PC7	Class
1	0.262827	0.158186	-2.46979	-1.53576	-0.57978	-0.10201	-0.50046	A
2	0.720632	-0.6746	1.341224	-0.8442	1.807203	0.571641	-0.92949	A
3	3.11601	0.385173	-1.49079	-1.03073	-1.69253	-0.08847	-1.1656	A
4	-0.0518	-1.56512	-0.43554	-1.29956	0.795841	-2.00985	-0.4845	A
5	1.309828	-1.32546	0.017954	-0.09628	0.932458	-0.63107	-1.91935	A
6	-6.97978	-2.32586	2.944116	-1.01966	1.489087	-0.51823	0.644704	A
7	0.402632	0.38055	-1.10794	-1.24122	-0.76512	-0.44023	-0.77642	A
8	0.75526	-0.66686	1.299138	-0.81882	1.752728	0.526824	-0.99606	A
9	3.028337	0.291641	-2.20754	-1.16455	-1.66052	0.054879	-1.06549	A
10	-0.84269	-1.60268	-0.11958	-0.92878	1.096133	-1.84083	-0.39183	A

Table 1.12: First 10 instances of the resulting dataset obtained after performing PCA in R. Only the first seven principal components were extracted out.

The percentage of variance explained by the first seven principal component was analyzed and it was found to be 92.7%. Figure 1.16 (a) shows the percentage of variance explained by each principal component and figure 1.16 (b) shows the trend in cumulative variance explained by increasing the number of principal components from 1 to 17.



(a) (b)

Fig 1.16: (a) Percentage of variance explained by each principal component. (b) Trend in cumulative variance explained by all 17 principal components

Figure 1.17 shows the data distribution according to the first two principal components. The original attributes are overlaid, as arrows, on the figure to show their relative contribution to PC1 and PC2. Bigger the magnitude of an arrow along an axis, higher will be the contribution of the corresponding original attribute to that axis [8].

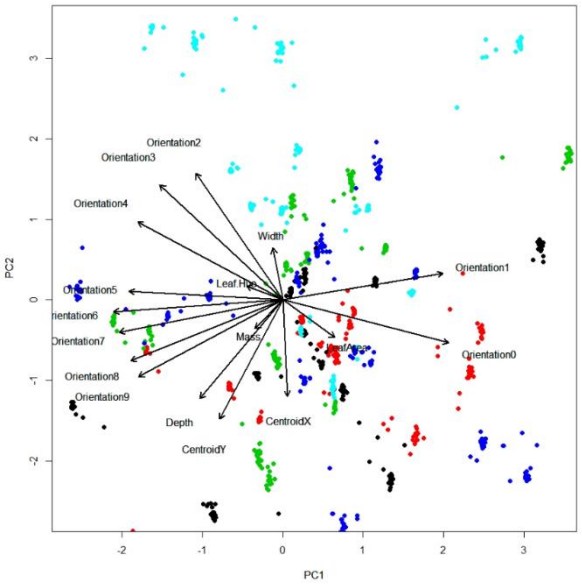


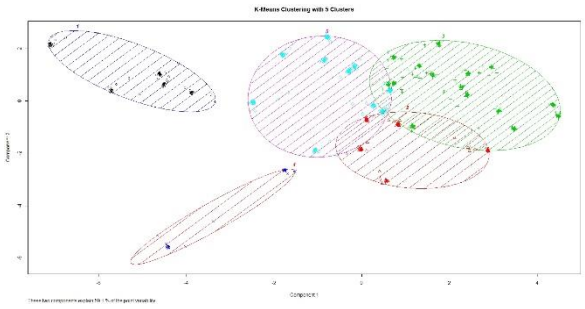
Fig 1.17: Graph of first two principal components. Colored dots represent the instances according to the five classes. The arrows represent the original attributes. The relative sizes of the arrows represent the amount of contribution of each of the attributes to PC1 and PC2.

2. Clustering

2.A Applying hierarchical, k-means and PAM clustering on Preprocessed Dataset

External metrics, such as the confusion matrix, true positive rate (TPR) and purity rates (PR) were used to describe the performance of the three clustering algorithms. Although clustering is an unsupervised method, and external metrics are generally used for supervised learning, use of external metrics could be done in this case since the dataset contained the actual classes.

The clustering output from each algorithm are recorded in a confusion matrix that has been programmed to maximize the diagonal. Maximizing the diagonal aligns the predicted classes with the actual classes as precisely as possible, under the parametric restrictions of a clustering algorithm. Hence, further analysis of confusion matrix to reveal the true positive and purity rates, which are indicators of the performance of the algorithm. The following figures contain the clustering output and accuracy of each of the algorithms.

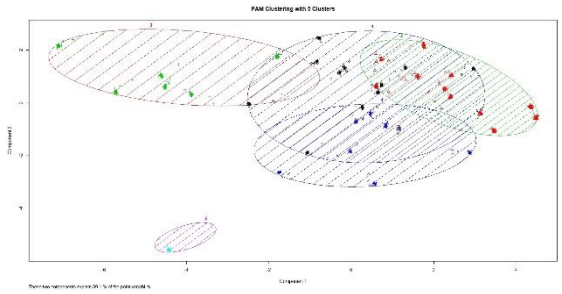


(a)

	1	4	5	3	2
A	0	39	19	40	20
B	0	57	19	57	0
C	0	57	38	19	38
D	0	39	18	76	19
E	64	0	0	21	84

(b)

Fig 2.1: (a) Clustering output of K-means with 5 clusters. (b) Confusion Matrix obtained from K-means clustering

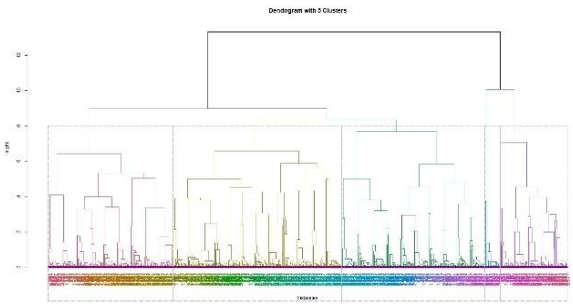


(a)

	1	2	3	5	4
A	81	18	19	0	0
B	38	57	38	0	0
C	39	38	38	0	37
D	51	57	19	0	25
E	22	42	0	21	84

(b)

Fig 2.2: (a) Clustering output of PAM with 5 clusters. (b) Confusion Matrix obtained from Pam



(a)

	1	5	4	3	2
A	39	0	19	20	40
B	38	0	19	38	38
C	57	0	38	38	19
D	58	0	18	57	19
E	43	21	0	21	84

(b)

Fig 2.3: (a) Dendogram of Hierarchical clustering with 5 clusters. (b) Confusion Matrix obtained from Hierarchical clustering

Table 2.1 shows the performance of the three algorithms in terms of TPR and PR. Although the true positive rates for K-means and PAM are similar, the purity rate for K-means are way higher. Moreover by comparing the clustering output of K-means and PAM it can be seen that PAM clusters tend to overlap each other much more, whereas K-means cluster maintained much more separation. Hence it can be said that K-means produces the best result when compared to the class attribute.

Algorithms	TP.Rate	Purity.Rate
K-means	0.35221	0.440608
PAM	0.359116	0.388122
Hierarchical	0.301105	0.356354

Table 2.1: Performance of three clustering algorithms when compared to class attribute

2.B Exploring Tuning of Parameters for K-means, PAM and Hclust

The parameters for each algorithm were chosen according to the following guideline:

- **K-means:** The number of clusters were kept constant at 5, the *nstart* parameter was set at 50 and maximum number of iterations were set at 100. Since the performance of K-means depend largely on initialization points, a large value for *nstart* ensures that the algorithms will search over large number of initialization points and only return the best results. The number of iteration was set at a large value so that the algorithm can iterate over a large number of possible orientations and output a fairly satisfactory value. Different variants of K-means algorithm were tested such as Hartigan-Wong, Lloyd/Forgy and Macqueen
- **PAM:** For PAM, the number of clusters were again set at 5, however the distance metric parameter was varied between *Euclidean* and *manhanttan.*
- **Hierarchical:** For hierarchical clustering, the linkage method was varied between Complete, Single and Average since these are the most functional, while keeping the number of clusters constant at 5. The other linkage type such as the centroid was not tested since it is mainly used in genomics and *suffers from inversion* (ISLR, pg 395).

The following tables represent the performance of different variants of these three algorithms.

Algorithms	Types	Mean. Diameter	Mean.of.Avg. Distances
K-means	Hartigan-Wong	6.862731	3.526381
-	Lloyd/Forgy	6.340349	3.064656
-	MacQueen	6.862731	3.526381
PAM	Euclidean	6.786556	3.207494
-	Manhattan	7.336368	3.631758
Hierarchical	Complete	6.292731	3.186001
-	Average	5.42512	2.441626
-	Single	4.313027	1.446351

(a)

Algorithms	Types	TP.Rate	Purity.Rate
K-means	Hartigan-Wong	0.372928	0.437845
-	Lloyd/Forgy	0.378453	0.407459
-	MacQueen	0.372928	0.437845
PAM	Euclidean	0.359116	0.388122
-	Manhattan	0.353591	0.414365
Hierarchical	Complete	0.301105	0.356354
-	Average	0.324586	0.354972
-	Single	0.266575	0.324586

(b)

Table 2.2: (a) Performance of the Algorithms based on internal metrics. (b)External metrics

According to the internal metrics, Hierarchical clustering with single linkage produced the best result – since both its mean diameter of clusters and average distances of each point in a cluster to its centroid is the lowest. However, based on external metrics, K-means - MacQueen produces the best results since both its TPR and PR are the highest.

Therefore, further inspection was carried out on the internal metrics between Hierarchical – Single and K-means Macqueen. Table 2.3 summarizes the findings.

It can be seen that cluster diameters and average distances of points in the cluster to centroid vary greatly for hierarchical single meaning there is higher probability of missclassifcation, whereas both the values are quite consistent for K-means Macqueen. Therefore, it can be concluded that clusters in K-means Macqueen are much more well distributed and presents better classification of instances.

Algorithms	Cluster	Diameter	Avg.Distance
Hclust Single	1	10.83145	4.961360236
-	2	3.321484	0.741001505
-	3	2.7744	0.516466619
-	4	0.90451	0.281519912
-	5	3.733293	0.731409082
K-means MacQueen	1	6.399071	3.613853509
-	2	6.364396	3.67279622
-	3	5.513403	3.054161216
-	4	8.978513	4.003800984
-	5	7.058275	3.287293443

Table 2.3: Comparing the internal metrics of Hierarchical Single and K-means Macqueen

2.C Performing Clustering on Alternative Datasets

The K-means Macqueen Algorithm was chosen since its performance was the best based on both internal and external metrics. Performance for alternative datasets were measured by keeping *Class* attribute under consideration. Table 2.4 summarizes the result of applying K-means on the alternative datasets.

Based on the results obtained, it can be said that the three datasets – dataset with no missing values from part – 1(f)(i), and datasets with missing values replaced by mean and median respectively, from part – 1(d)(i), had the most positive impact on the quality of the clustering.

Dataset	TP.Rate	Purity.Rate
10 Prin Comps	0.352209945	0.440607735
With no NA	0.371546961	0.377071823
NA replaced with zero	0.349447514	0.354972376
NA replaced with mean	0.371546961	0.377071823
NA replaced with median	0.371546961	0.377071823

Table 2.4: Performance of K-means Macqueen on five different datasets

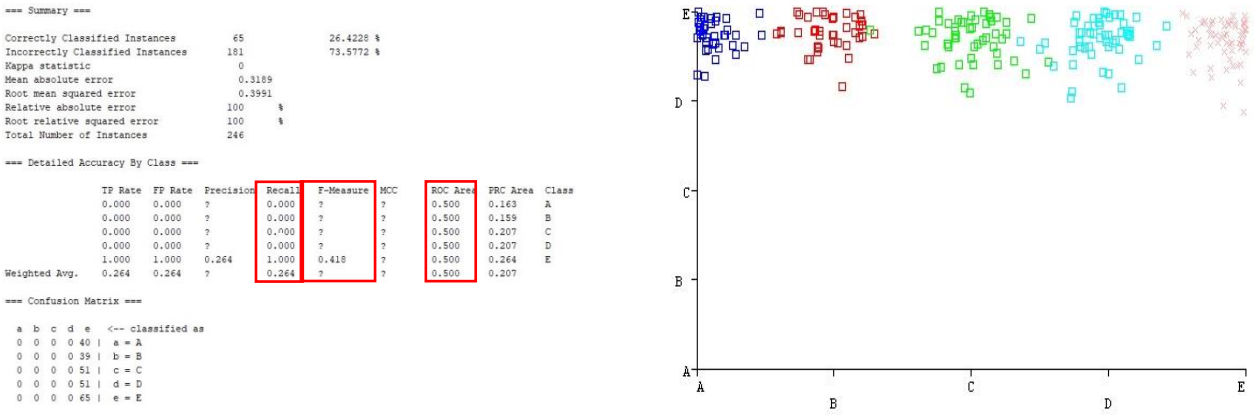
3. Classification using WEKA

3.A Using classification Algorithms

The dataset used for part 3 had been pre-processed to remove all missing attributes and instances. Mean imputation was performed on instances with missing values. Finally the dataset was standardized.

The dataset was then split into training set (66%) and test set (33%) and five different classification algorithms were applied on it: ZeroR, OneR, NaiveBayes, iBk (k-NN) and J48(C4.5).

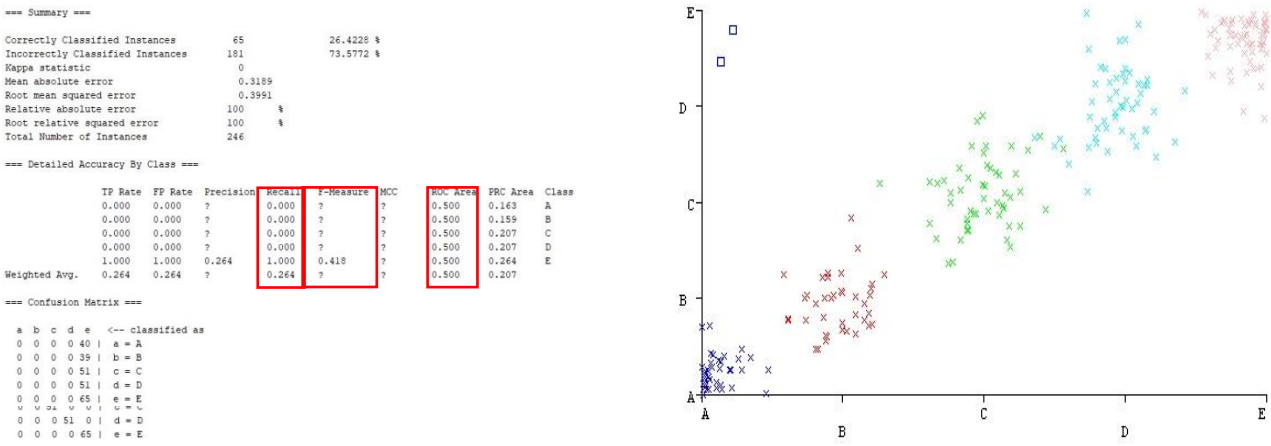
The following figures illustrate the performance of these algorithms on the dataset.



```
=== Summary ===

Correctly Classified Instances          65               26.4228 %
Incorrectly Classified Instances       181               73.5772 %
Kappa statistic                          0
Mean absolute error                      0.3189
Root mean squared error                  0.3991
Relative absolute error                100      %
Root relative squared error            100      %
Total Number of Instances              246

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure  MCC      ROC Area  PRC Area  Class
                 0.000    0.000    ?          0.000    ?          ?        0.500     0.163     A
                 0.000    0.000    ?          0.000    ?          ?        0.500     0.159     B
                 0.000    0.000    ?          0.000    ?          ?        0.500     0.207     C
                 0.000    0.000    ?          0.000    ?          ?        0.500     0.207     D
                 1.000    1.000    0.264      1.000    0.418      ?        0.500     0.264     E
Weighted Avg.    0.264    0.264    ?          0.264    ?          ?        0.500     0.207

=== Confusion Matrix ===

  a  b  c  d  e   <-- classified as
  0  0  0  0 40 |  a = A
  0  0  0  0 39 |  b = B
  0  0  0  0 51 |  c = C
  0  0  0  0 51 |  d = D
  0  0  0  0 65 |  e = E
```

(a) (b)

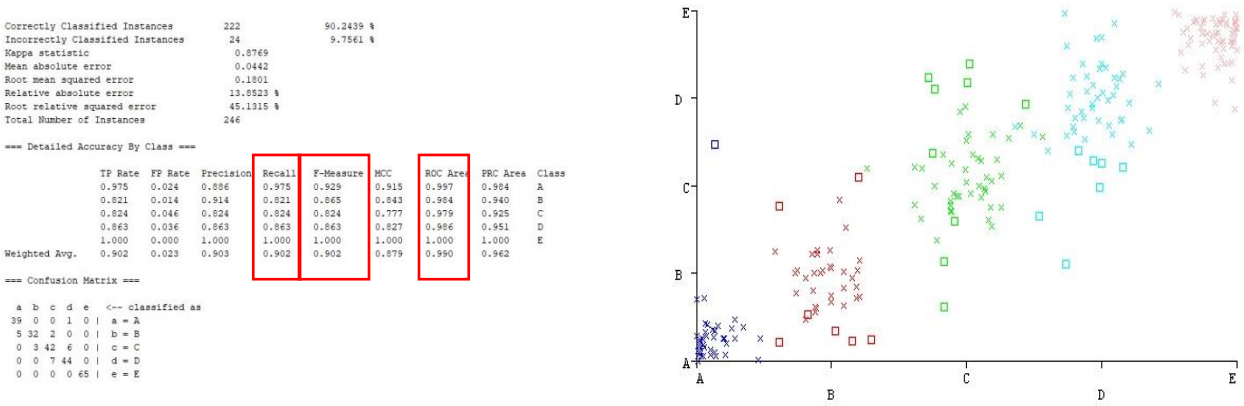
Fig 3.1: (a) Performance of ZeroR (b) Classifier errors of ZeroR



```
=== Summary ===

Correctly Classified Instances          65               26.4228 %
Incorrectly Classified Instances       181               73.5772 %
Kappa statistic                          0
Mean absolute error                      0.3189
Root mean squared error                  0.3991
Relative absolute error                100      %
Root relative squared error            100      %
Total Number of Instances              246

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure  MCC      ROC Area  PRC Area  Class
                 0.000    0.000    ?          0.000    ?          ?        0.500     0.163     A
                 0.000    0.000    ?          0.000    ?          ?        0.500     0.159     B
                 0.000    0.000    ?          0.000    ?          ?        0.500     0.207     C
                 0.000    0.000    ?          0.000    ?          ?        0.500     0.207     D
                 1.000    1.000    0.264      1.000    0.418      ?        0.500     0.264     E
Weighted Avg.    0.264    0.264    ?          0.264    ?          ?        0.500     0.207

=== Confusion Matrix ===

  a  b  c  d  e   <-- classified as
  0  0  0  0 40 |  a = A
  0  0  0  0 39 |  b = B
  0  0  0  0 51 |  c = C
  0  0  0  0 51 |  d = D
  0  0  0  0 65 |  e = E
  0  0  0 51  0 |  d = D
  0  0  0  0 65 |  e = E
```

(a) (b)

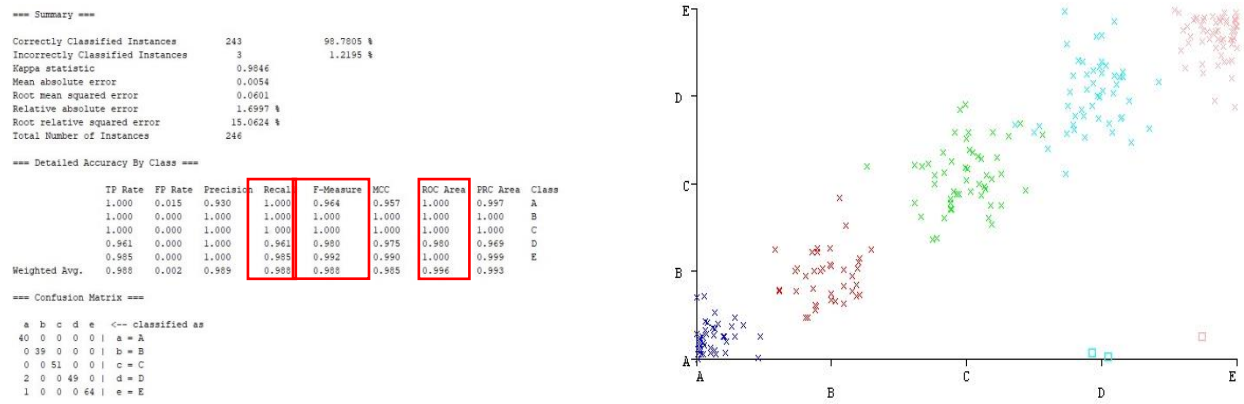
Fig 3.2: (a) Performance of OneR. (b) Classifier errors OneR



```
Correctly Classified Instances         222               90.2439 %
Incorrectly Classified Instances        24                9.7561 %
Kappa statistic                          0.8769
Mean absolute error                      0.0442
Root mean squared error                  0.1801
Relative absolute error                 13.8523 %
Root relative squared error             45.1315 %
Total Number of Instances              246

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   MCC      ROC Area  PRC Area  Class
                 0.975    0.024      0.886     0.975      0.929     0.915     0.997     0.984     A
                 0.821    0.014      0.914     0.821      0.865     0.843     0.984     0.940     B
                 0.824    0.046      0.824     0.824      0.824     0.777     0.979     0.925     C
                 0.863    0.036      0.863     0.863      0.863     0.827     0.986     0.951     D
                 1.000    0.000      1.000     1.000      1.000     1.000     1.000     1.000     E
Weighted Avg.    0.902    0.023      0.903     0.902      0.902     0.879     0.990     0.962

=== Confusion Matrix ===

  a  b  c  d  e   <-- classified as
 39  0  0  1  0 |  a = A
  5 32  2  0  0 |  b = B
  0  3 42  6  0 |  c = C
  0  0  7 44  0 |  d = D
  0  0  0  0 65 |  e = E
```


(a) (b)

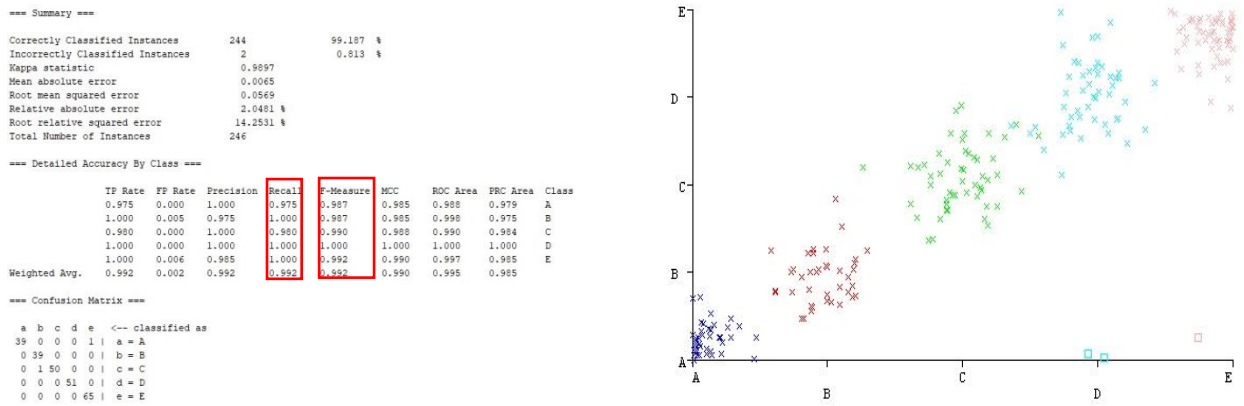
Fig 3.3: (a) Performance of NaiveBayes. (b) Classifier errors of Naïve Bayes



```
=== Summary ===

Correctly Classified Instances         243               98.7805 %
Incorrectly Classified Instances         3                1.2195 %
Kappa statistic                          0.9846
Mean absolute error                      0.0054
Root mean squared error                  0.0601
Relative absolute error                  1.6997 %
Root relative squared error             15.0624 %
Total Number of Instances              246

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   MCC      ROC Area  PRC Area  Class
                 1.000    0.015      0.930     1.000      0.964     0.957     1.000     0.997     A
                 1.000    0.000      1.000     1.000      1.000     1.000     1.000     1.000     B
                 1.000    0.000      1.000     1 000      1.000     1.000     1.000     1.000     C
                 0.961    0.000      1.000     0.961      0.980     0.975     0.980     0.969     D
                 0.985    0.000      1.000     0.985      0.992     0.990     1.000     0.999     E
Weighted Avg.    0.988    0.002      0.989     0.988      0.988     0.985     0.996     0.993

=== Confusion Matrix ===

  a  b  c  d  e   <-- classified as
 40  0  0  0  0 |  a = A
  0 39  0  0  0 |  b = B
  0  0 51  0  0 |  c = C
  2  0  0 49  0 |  d = D
  1  0  0  0 64 |  e = E
```


(a) (b)

Fig 3.4: (a) Performance of iBk (k-NN) (b) Classifier errors iBk (k-NN)



```
=== Summary ===

Correctly Classified Instances         244               99.187  %
Incorrectly Classified Instances         2                0.813  %
Kappa statistic                          0.9897
Mean absolute error                      0.0065
Root mean squared error                  0.0569
Relative absolute error                  2.0481 %
Root relative squared error             14.2531 %
Total Number of Instances              246

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   MCC      ROC Area  PRC Area  Class
                 0.975    0.000      1.000     0.975      0.987     0.985     0.988     0.979     A
                 1.000    0.005      0.975     1.000      0.987     0.985     0.998     0.975     B
                 0.980    0.000      1.000     0.980      0.990     0.988     0.990     0.984     C
                 1.000    0.000      1.000     1.000      1.000     1.000     1.000     1.000     D
                 1.000    0.006      0.985     1.000      0.992     0.990     0.997     0.985     E
Weighted Avg.    0.992    0.002      0.992     0.992      0.992     0.990     0.995     0.985

=== Confusion Matrix ===

  a  b  c  d  e   <-- classified as
 39  0  0  0  1 |  a = A
  0 39  0  0  0 |  b = B
  0  1 50  0  0 |  c = C
  0  0  0 51  0 |  d = D
  0  0  0  0 65 |  e = E
```


(a) (b)

Fig 3.5: (a) Performance of J48 (C4.5) (b) Classifier errors J48 (C4.5)

OneR and iBk has the highest rate of correctly classified instances with 99.187% for each. However, for comparison purposes we will consider three different metrics – ROC Area, F-Measure and Recall.

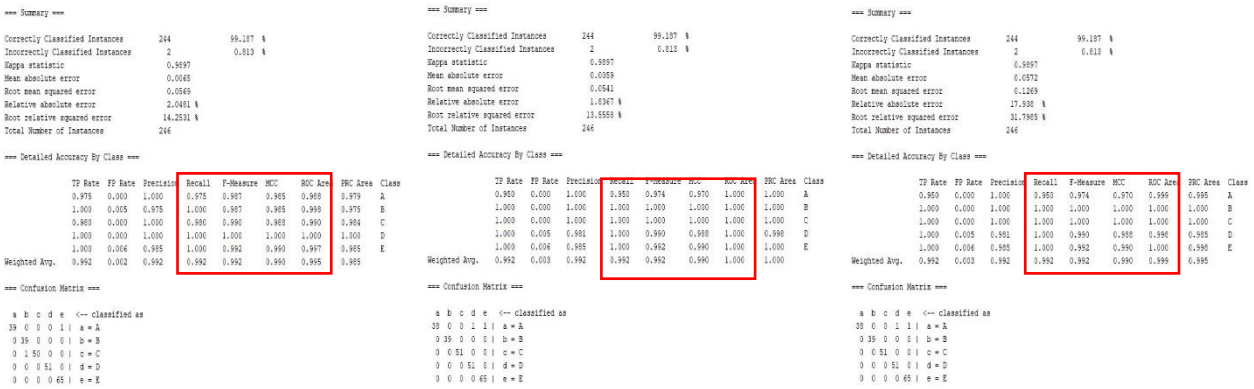
- **ROC:** The area under ROC curves indicates how much a model can distinguish between classes, with higher values being better [9]. The area under ROC varies between 0 to 1, with values closer to 1 being considered good. ROCs are generally considered a good metric for comparing various classifiers [10].
- **F-measure:** considers both precision and recall, two values that are widely used in distinguishing between true positives, false positives and false negatives, in its calculation. The formulation of F-measure is such that it is a harmonic mean of precision and recall. Combination of such important quantities make F measure an important metric to consider while evaluating classifiers.
- **Recall:** is somewhat a much simpler metric than both ROC and F-measure, however it is a powerful tool to quickly check the performance of classifiers.

Comparing the three quantities described above for both OneR and iBk, it can be seen that OneR slightly edges out iBk by having 100% accuracy rate in three of the five classes. **Hence OneR produces the best results.**

3.B Finding the Parameters that Improve the Predictive Ability of the Algorithm:

For this part parameters of iBk (k-NN) were varied to find the optimal settings. The formulation of the k-NN algorithm is such that – increasing the value of k would increase the predictive ability. For very low values of K, the algorithm can overfit the data and display low bias but very high variance. Whereas, very high values of K would cause the algorithm to behave like a linear classifier, i.e. low variance, but high bias characteristics [11]. Therefore an optimal value should be found that negates the inefficiencies displayed by extreme values of K.

The following figures show the results of using three different values of K on the preprocessed dataset.



(a) (b) (c)

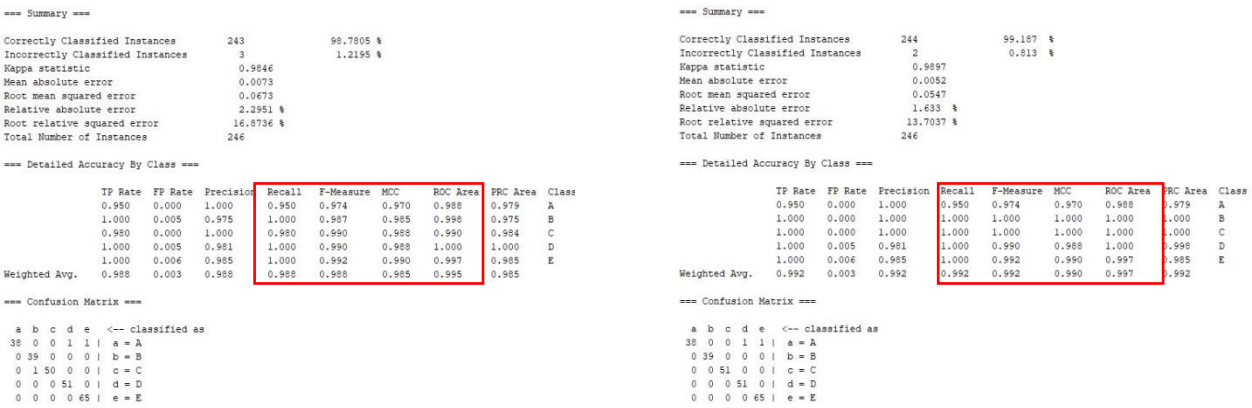
Fig 3.6: Performance of iBk-kNN for (a) K = 1 (b) K = 10 (c) K = 15

From the performance metrics it can be concluded that as value of K was increased from 1 to 10, the performance metrics such as ROC Area, F measures and Recall all improved slightly. However, further increasing the value of K to 15, actually caused the performance to dip. This might have been due to the fact that at higher values of K, the k-NN algorithm starts to display tendencies similar to linear classifiers. **Hence setting K = 10 would improve the predictive ability of the algorithm.**

3.C Applying Classification Algorithm on Alternative Datasets.

While determining which dataset had the best impact on IBk (k-NN), the same performance metrics as before – ROC Area, F-measure and Recall rates were taken into consideration.

The following figures reflect these metrics for the five datasets under inspection.



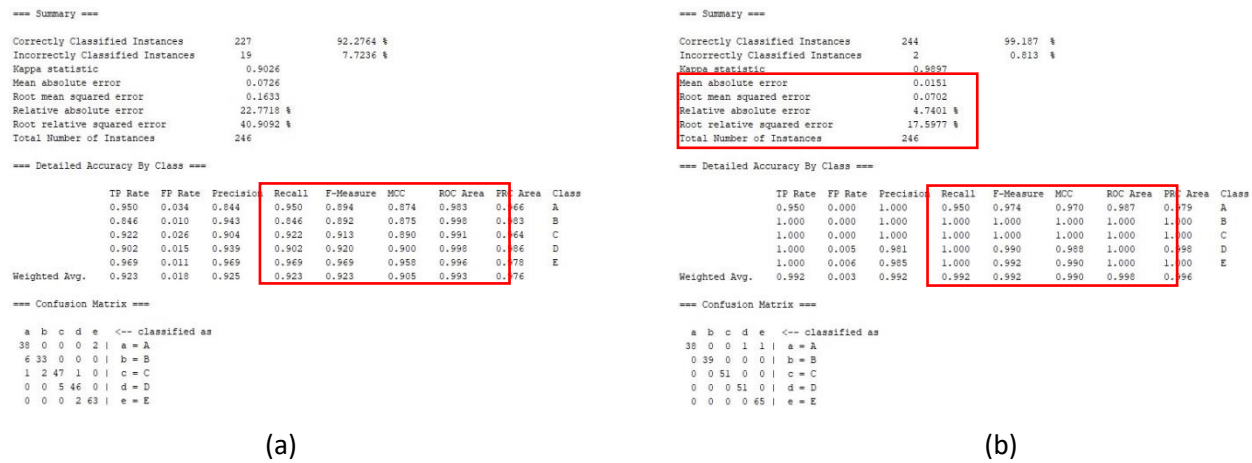
```
=== Summary ===

Correctly Classified Instances         243               98.7805 %
Incorrectly Classified Instances         3                1.2195 %
Kappa statistic                          0.9846
Mean absolute error                      0.0073
Root mean squared error                  0.0673
Relative absolute error                  2.2951 %
Root relative squared error             16.8736 %
Total Number of Instances              246

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   MCC      ROC Area  PRC Area  Class
                 0.950    0.000     1.000      0.950     0.974     0.970     0.988     0.979     A
                 1.000    0.005     0.975      1.000     0.987     0.985     0.998     0.975     B
                 0.980    0.000     1.000      0.980     0.990     0.988     0.990     0.984     C
                 1.000    0.005     0.981      1.000     0.990     0.988     1.000     1.000     D
                 1.000    0.006     0.985      1.000     0.992     0.990     0.997     0.985     E
Weighted Avg.    0.988    0.003     0.988      0.988     0.988     0.985     0.995     0.985

=== Confusion Matrix ===

  a  b  c  d  e   <-- classified as
 38  0  0  1  1 |  a = A
  0 39  0  0  0 |  b = B
  0  1 50  0  0 |  c = C
  0  0  0 51  0 |  d = D
  0  0  0  0 65 |  e = E
```

(a)

```
=== Summary ===

Correctly Classified Instances         244               99.187  %
Incorrectly Classified Instances         2                0.813  %
Kappa statistic                          0.9897
Mean absolute error                      0.0052
Root mean squared error                  0.0547
Relative absolute error                  1.633  %
Root relative squared error             13.7037 %
Total Number of Instances              246

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   MCC      ROC Area  PRC Area  Class
                 0.950    0.000     1.000      0.950     0.974     0.970     0.988     0.979     A
                 1.000    0.000     1.000      1.000     1.000     1.000     1.000     1.000     B
                 1.000    0.000     1.000      1.000     1.000     1.000     1.000     1.000     C
                 1.000    0.005     0.981      1.000     0.990     0.988     1.000     0.998     D
                 1.000    0.006     0.985      1.000     0.992     0.990     0.997     0.985     E
Weighted Avg.    0.992    0.003     0.992      0.992     0.992     0.990     0.997     0.992

=== Confusion Matrix ===

  a  b  c  d  e   <-- classified as
 38  0  0  1  1 |  a = A
  0 39  0  0  0 |  b = B
  0  0 51  0  0 |  c = C
  0  0  0 51  0 |  d = D
  0  0  0  0 65 |  e = E
```

(b)

Fig 3.7: Performance of IBk (k-NN) on: (a) Dataset with 10 Prin. Components (b) Dataset with no missing values



```
=== Summary ===

Correctly Classified Instances         227               92.2764 %
Incorrectly Classified Instances        19                7.7236 %
Kappa statistic                          0.9026
Mean absolute error                      0.0726
Root mean squared error                  0.1633
Relative absolute error                 22.7718 %
Root relative squared error             40.9092 %
Total Number of Instances              246

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   MCC      ROC Area  PRC Area  Class
                 0.950    0.034     0.844      0.950     0.894     0.874     0.983     0.966     A
                 0.846    0.010     0.943      0.846     0.892     0.875     0.998     0.983     B
                 0.922    0.026     0.904      0.922     0.913     0.890     0.991     0.964     C
                 0.902    0.015     0.939      0.902     0.920     0.900     0.998     0.986     D
                 0.969    0.011     0.969      0.969     0.969     0.958     0.996     0.978     E
Weighted Avg.    0.923    0.018     0.925      0.923     0.923     0.905     0.993     0.976

=== Confusion Matrix ===

  a  b  c  d  e   <-- classified as
 38  0  0  0  2 |  a = A
  6 33  0  0  0 |  b = B
  1  2 47  1  0 |  c = C
  0  0  5 46  0 |  d = D
  0  0  0  2 63 |  e = E
```

(a)

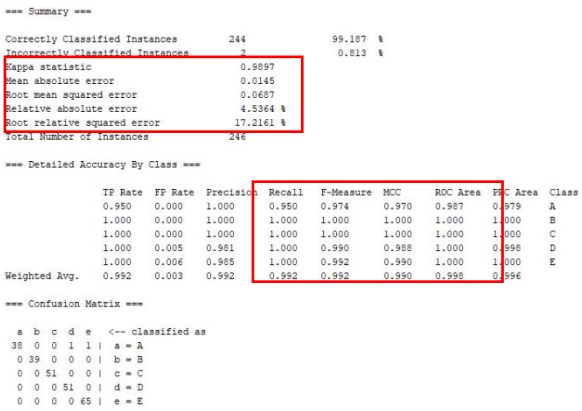
```
=== Summary ===

Correctly Classified Instances         244               99.187  %
Incorrectly Classified Instances         2                0.813  %
Kappa statistic                          0.9897
Mean absolute error                      0.0151
Root mean squared error                  0.0702
Relative absolute error                  4.7401 %
Root relative squared error             17.5977 %
Total Number of Instances              246

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   MCC      ROC Area  PRC Area  Class
                 0.950    0.000     1.000      0.950     0.974     0.970     0.987     0.979     A
                 1.000    0.000     1.000      1.000     1.000     1.000     1.000     1.000     B
                 1.000    0.000     1.000      1.000     1.000     1.000     1.000     1.000     C
                 1.000    0.005     0.981      1.000     0.990     0.988     1.000     0.998     D
                 1.000    0.006     0.985      1.000     0.992     0.990     1.000     1.000     E
Weighted Avg.    0.992    0.003     0.992      0.992     0.992     0.990     0.998     0.996

=== Confusion Matrix ===

  a  b  c  d  e   <-- classified as
 38  0  0  1  1 |  a = A
  0 39  0  0  0 |  b = B
  0  0 51  0  0 |  c = C
  0  0  0 51  0 |  d = D
  0  0  0  0 65 |  e = E
```

(b)

Fig 3.8: Performance of IBk (k-NN) on: (a) Dataset with Zero Imputation (b) Dataset with Median Imputation

From the performance metrics it can be said concluded that IBk (k-NN) performs the best on the datasets that had been imputed by mean and median column values. However, the mean imputed dataset slightly edges out the median imputed dataset by having better Root mean squared values, mean absolute error and relative absolute error.

This might have been due to the fact that k-NN algorithm functions by selecting an observed point and choosing *K* points around it to make a class label prediction on that observed data point. Therefore, if missing values in a dataset are replaced by the mean values of the attribute column, then there would be higher possibility that any arbitrarily chosen point would be closer to its corresponding point(s); and the average distance between any two arbitrarily chosen points would be less. This would result in lower absolute errors in Euclidean space.



```
=== Summary ===

Correctly Classified Instances         244               99.187  %
Incorrectly Classified Instances         2                0.813  %
Kappa statistic                          0.9897
Mean absolute error                      0.0145
Root mean squared error                  0.0687
Relative absolute error                  4.5364 %
Root relative squared error             17.2161 %
Total Number of Instances              246

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   MCC      ROC Area  PRC Area  Class
                 0.950    0.000    1.000      0.950    0.974      0.970    0.987     0.979     A
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     B
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     C
                 1.000    0.005    0.981      1.000    0.990      0.988    1.000     0.998     D
                 1.000    0.006    0.985      1.000    0.992      0.990    1.000     1.000     E
Weighted Avg.    0.992    0.003    0.992      0.992    0.992      0.990    0.998     0.996

=== Confusion Matrix ===

  a  b  c  d  e   <-- classified as
 38  0  0  1  1 |  a = A
  0 39  0  0  0 |  b = B
  0  0 51  0  0 |  c = C
  0  0  0 51  0 |  d = D
  0  0  0  0 65 |  e = E
```

Fig 3.9: IBk (k-NN)on Dataset with Mean Imputation.

4. References

[1] Knuth, Kevin. (2006). Optimal Data-Based Binning for Histograms.

[2] histogram, C., Stark, T., Hyndman, R., Motulsky, H., Turner, I. and Bannier, B. (2019). *Calculating optimal number of bins in a histogram*. [online] Cross Validated. Available at: https://stats.stackexchange.com/questions/798/calculating-optimal-number-of-bins-in-a-histogram?fbclid=IwAR2Ew2E_8QmfKkpKJ2i49L4N9ULSxAFZICoa8JHFP8ge8AB5L7NJlG2t6e4 [Accessed 13 May 2019].

[3] Support.minitab.com. (2019). *Interpret all statistics and graphs for Descriptive Statistics - Minitab Express*. [online] Available at: https://support.minitab.com/en-us/minitab-express/1/help-and-how-to/basic-statistics/summary-statistics/descriptive-statistics/interpret-the-results/all-statistics-and-graphs/ [Accessed 13 May 2019].

[4] Towards Data Science. (2019). *How to Handle Missing Data*. [online] Available at: https://towardsdatascience.com/how-to-handle-missing-data-8646b18db0d4 [Accessed 13 May 2019].

[5] Dong, Y. and Peng, C. (2013). Principled missing data methods for researchers. *SpringerPlus*, 2(1).

[6] Dray, S. and Josse, J. (2014). Principal component analysis with missing values: a comparative survey of methods. *Plant Ecology*, 216(5), pp.657-667.

[7] Ratner, B. (2009). The correlation coefficient: Its values range between +1/−1, or do they?. *Journal of Targeting, Measurement and Analysis for Marketing*, 17(2), pp.139-142.

[8] James, G., Witten, D., Hastie, T. and Tibshirani, R. (2013). *An Introduction to Statistical Learning: with Applications in R*. 1st ed. New York: Springer, pp.374 -384.

[9] Towards Data Science. (2019). *Understanding AUC - ROC Curve*. [online] Available at: https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5 [Accessed 13 May 2019]

[10] James, G., Witten, D., Hastie, T. and Tibshirani, R. (2013). *An Introduction to Statistical Learning: with Applications in R*. 1st ed. New York: Springer, pp.147.

[11] James, G., Witten, D., Hastie, T. and Tibshirani, R. (2013). *An Introduction to Statistical Learning: with Applications in R*. 1st ed. New York: Springer, pp.40.

5. Appendix

Github Link for all R code: https://github.com/mtrkhan854/G54DMA-Coursework

(Please let me know if there are any broken links)