INDIAN STATISTICAL INSTITUTE

DELHI CENTER

MSTAT

CEREAL DATA ANALYSIS

Project

Kaulik Poddar - MD2207

Saheli Datta - MD2213

Tiyasa Dutta - MD2225

Professor: Dr. Swagata Nandi

Session: 2022 - 2023

Date of Submission: 09 April 2023

Contents

1 Introduction

Data Analysis is one of the most burning concept around which everyone is focusing upon. So we are also going to be a part of it and for now our focus is on the Cereal Dataset. Now the question comes why this dataset?
Cereals are very important in our life they are a potential source of carbohydrates proteins and many other minerals. In our data we have 43 brands each brand showing different proportions of calories , protein , sodium etc. We will see throughout the report how these variables are going to give a competition among the brand's as well as among the manufacturers.

2 Objectives

As already mentioned the main motive is to analyse the data as well as applying all the theoretical contents on the data which we have learnt.
We know in real life datasets are not always as expected , they may not be normal may not be symmetric.
So our first goal is to make a exploratory analysis over the entire data, checking normality , multivariate normality over the entire data and also group wise and then comment on how significant is the grouping factor.
We are also interested if we consider the grouping are really the three populations

very different if so how much.

We will also do PCA to see whether there is any variable reduction in the data. And last but not the least we will perform factor analysis to see whether our X can be written as a linear combination of factors.

3 Description of the Dataset

The given Dataset contains observations on different type nutrients of cereal meals, produced by three kind of manufacturer. There are total 43 observations on 8 kinds of nutrients, observed on 43 kinds of meals produced by 3 manufacturers.

1. Brand of Cereal: Identifies the brand of cereal meal.

2. Manufacturer: Identifies the manufacturer of the product.

1 - General Mills

2 - Kellogg

3 - Quaker

3. Calories

4. Protein

5. Fat

6. Sodium

7. Fiber

8. Carbohydrates

9. Sugar

10. Potassium

The whole dataset is given in the following table:

Brand of cereal	Manufacturer	Calories	Protein	Fat	Sodium	Fiber	Carbohydrates	Sugar	Potassium
ACCheerios	1	110	2	2	180	1.5	10.5	10	70
Cheerios	1	110	6	2	290	2	17	1	105
CocoaPuffs	1	110	1	1	180	0	12	13	55
CountChocula	1	110	1	1	180	0	12	13	65
GoldenGrahams	1	110	1	1	280	0	15	9	45
HoneyNutCheerios	1	110	3	1	250	1.5	11.5	10	90
Kix	1	110	2	1	260	0	21	3	40
LuckyCharms	1	110	2	1	180	0	12	12	55
MultiGrainCheerios	1	100	2	1	220	2	15	6	90
OatmealRaisinCrisp	1	130	3	2	170	1.5	13.5	10	120
RaisinNutBran	1	100	3	2	140	2.5	10.5	8	140
TotalCornFlakes	1	110	2	1	200	0	21	3	35
TotalRaisinBran	1	140	3	1	190	4	15	14	230
TotalWholeGrain	1	100	3	1	200	3	16	3	110
Trix	1	110	1	1	140	0	13	12	25
Cheaties	1	100	3	1	200	3	17	3	110
WheatiesHoneyGold	1	110	2	1	200	1	16	8	60
AllBran	2	70	4	1	260	9	7	5	320
AppleJacks	2	110	2	0	125	1	11	14	30
CornFlakes	2	100	2	0	290	1	21	2	35
CornPops	2	110	1	0	90	1	13	12	20
CracklinOatBran	2	110	3	3	140	4	10	7	160
Crispix	2	110	2	0	220	1	21	3	30
FrootLoops	2	110	2	1	125	1	11	13	30
FrostedFlakes	2	110	1	0	200	1	14	11	25
FrostedMiniWheats	2	100	3	0	0	3	14	7	100
FruitfulBran	2	120	3	0	240	5	14	12	190
JustRightCrunchyNuggets	2	110	2	1	170	1	17	6	60
MueslixCrispyBlend	2	160	3	2	150	3	17	13	160
NutNHoneyCrunch	2	120	2	1	190	0	15	9	40
NutriGrainAlmondRaisin	2	140	3	2	220	3	21	7	130
NutriGrainWheat	2	90	3	0	170	3	18	2	90
Product19	2	100	3	0	320	1	20	3	45
RaisinBran	2	120	3	1	210	5	14	12	240
RiceKrispies	2	110	2	0	290	0	22	3	35
Smacks	2	110	2	1	70	1	9	15	40
SpecialK	2	110	6	0	230	1	16	3	55
CapNCrunch	3	120	1	2	220	0	12	12	35
HoneyGrahamOhs	3	120	1	2	220	1	12	11	45
Life	3	100	4	2	150	2	12	6	95
PuffedRice	3	50	1	0	0	0	13	0	15
PuffedWheat	3	50	2	0	0	1	10	0	50
QuakerOatmeal	3	100	5	2	0	2.7	1	1	110

4 Methods and Tools Required to Analyse the Dataset

Data Visualisation Tools:

(a) Density Plot:

A density plot is a representation of the distribution of a numeric variable. It uses a kernel density estimate to show the probability density function of the variable. It is a smoothed version of the histogram and is used in the same concept.

```
Package              : Lattice
Function             : densityplot()
```

(b) QQ Plot:

The quantile-quantile (q-q) plot is a graphical technique for determining if two data sets come from populations with a common distribution. A q-q plot is a plot of the quantiles of the first data set against the quantiles of the second data set. Here in the x-axis we plotted Theoretical Normal (0,1) quantiles and in y axis standardised sample quantiles are plotted.

```
Package              : Lattice, LatticeExtra
Function             : qqmath()
```

(c) Correlation Heatmap:

A correlation heatmap is a heatmap that shows a 2D correlation matrix between two discrete dimensions, using colored cells to represent data from usually a monochromatic scale. The values of the first dimension appear as the rows of

the table while of the second dimension as a column. The color of the cell is proportional to the number of measurements that match the dimensional value. This makes correlation heatmaps ideal for data analysis since it makes patterns easily readable and highlights the differences and variation in the same data. A correlation heatmap, like a regular heatmap, is assisted by a colorbar making data easily readable and comprehensible.

`Package` : `ggplot2, reshape2`

`Function` : `ggplot(), melt()`

(d) Gamma Plot:

Consider,

$$d_j{}^2 = (\underset{\sim}{x}_j - \underset{\sim}{\tilde{x}})' S^{-1} (\underset{\sim}{x}_j - \underset{\sim}{\tilde{x}})$$

j= 1, 2, ...,p

Where, S is the sample covariance matrix. $d_j{}^2$ is called the squared generalised distance. The plot of $d_j{}^2$ and χ^2 is called Gamma Plot.

`Package` : `ggplot2, qqplotr`

`Function` : `ggplot(), stat_qq_band()`

(e) biplot:

Biplots are a type of exploratory graph used in statistics, a generalization of the simple two-variable scatterplot. A biplot overlays a score plot with a loading plot. A biplot allows information on both samples and variables of a data matrix to be displayed graphically. Samples are displayed as points while variables are displayed either as vectors, linear axes or nonlinear trajectories. Here, biplots are

displayed for showing the significance of variables in explaining first and second Principal Components only.

```
Package              : base
Function             : biplot()
```

(f) Scree plot:

In multivariate statistics, a scree plot is a line plot of the eigenvalues of factors or principal components in an analysis.[1] The scree plot is used to determine the number of factors to retain in an exploratory factor analysis (FA) or principal components to keep in a principal component analysis (PCA).

```
Package              : ggplot2
Function             : ggplot()
```

Tools used for carrying out several Test Procedures:

(a) UNIVARIATE SHAPIRO-WILK NORMALITY TEST:

The Shapiro-Wilk test is a test for normality. It tests whether a sample x1, x2,..., xk came from a normally distributed population.

The testing hypothesis:

H_0 : The underlying population is normaly distributed

H_1 : The underlying population is not normaly distributed

The test statistic is

$$W = \frac{(\sum_{t=1}^{n} a_i x_{(i)})^2}{\sum_{t=1}^{n} (x_i - \bar{x})^2}$$

where, $x_{(i)}$ is the ith order statistic.

$\bar{x}$ is the sample mean.

The coefficients a_i are given by,

$$W = \frac{m^T V^{-}1}{\sqrt{(m^T V^{-}1 V^{-}1 m)}}$$

`Package : stats`

`Function : shapiro.test()`

(b) ROYSTON TEST:

The Royston test is a test for checking whether the variables follow a multivariate normal distribution.

`Package : MVN`

`Function : mvn()`

(c) BOX'S M TEST:

The Box M test is a test for checking whether two populations, that follow multivariate normal distribution, have the same covariance structure.

`Package : biotools`

`Function : boxM()`

Packages Needed to be Installed

`plotly, lattice, MASS, ggplot2, latticeExtra, gridExtra, car tactile, EnvStats, dplyr, reshape, RVAideMemoire, heplots, qqplotr, MVN, biotools, forecast, MVQuickGraphs`

5 Analysis of the Dataset

In this section, we will analyse the whole dataset using suitable measures and graphical representations.

5.1 Exploratory Data Analysis (Not Considering any Grouping Factor)

The dataset considered here can be grouped with respect to the Manufacturer. Clearly, the data is nothing but observations from Manufacturer 1, Manufacturer 2 and Manufacturer 3. So why should we not consider the groups only and check for Multivariate Normality within these three groups?

In this section we are not considering the groups but checking for overall Multivariate Normality because -

1. If the overall Multivariate Normality gets accepted then we are done. Otherwise, from the very beginning we need to test for equality of variances of the three groups and then we need to test for equality of means.

2. These tests may require transformation of variables etc.

3. Also, even if we consider 3 groups, the groups sizes are very small (¡20), testing for multivariate normality for such groups may lead to some false inference about the data.

To avoid these, we will just test for multivariate normality considering the fact that the three groups are homogeneous.

Given below some plots and summary dataset which may bring some insights of the dataset.

Given below the summary of the dataset without considering the grouping factor:

Calories	Protein	Fat	Sodium
Min. : 50.0	Min. :1.000	Min. :0.0000	Min. : 0.0
1st Qu.:100.0	1st Qu.:2.000	1st Qu.:0.0000	1st Qu.:145.0
Median :110.0	Median :2.000	Median :1.0000	Median :190.0
Mean :107.9	Mean :2.465	Mean :0.9767	Mean :180.5
3rd Qu.:110.0	3rd Qu.:3.000	3rd Qu.:1.5000	3rd Qu.:220.0
Max. :160.0	Max. :6.000	Max. :3.0000	Max. :320.0

Fiber	Carbohydrates	Sugar	Potassium
Min. :0.000	Min. : 1.00	Min. : 0.000	Min. : 15.00
1st Qu.:0.500	1st Qu.:12.00	1st Qu.: 3.000	1st Qu.: 37.50
Median :1.000	Median :14.00	Median : 8.000	Median : 60.00
Mean :1.714	Mean :14.26	Mean : 7.605	Mean : 84.42
3rd Qu.:2.850	3rd Qu.:17.00	3rd Qu.:12.000	3rd Qu.:110.00
Max. :9.000	Max. :22.00	Max. :15.000	Max. :320.00

The Boxplot of the variables are given below:

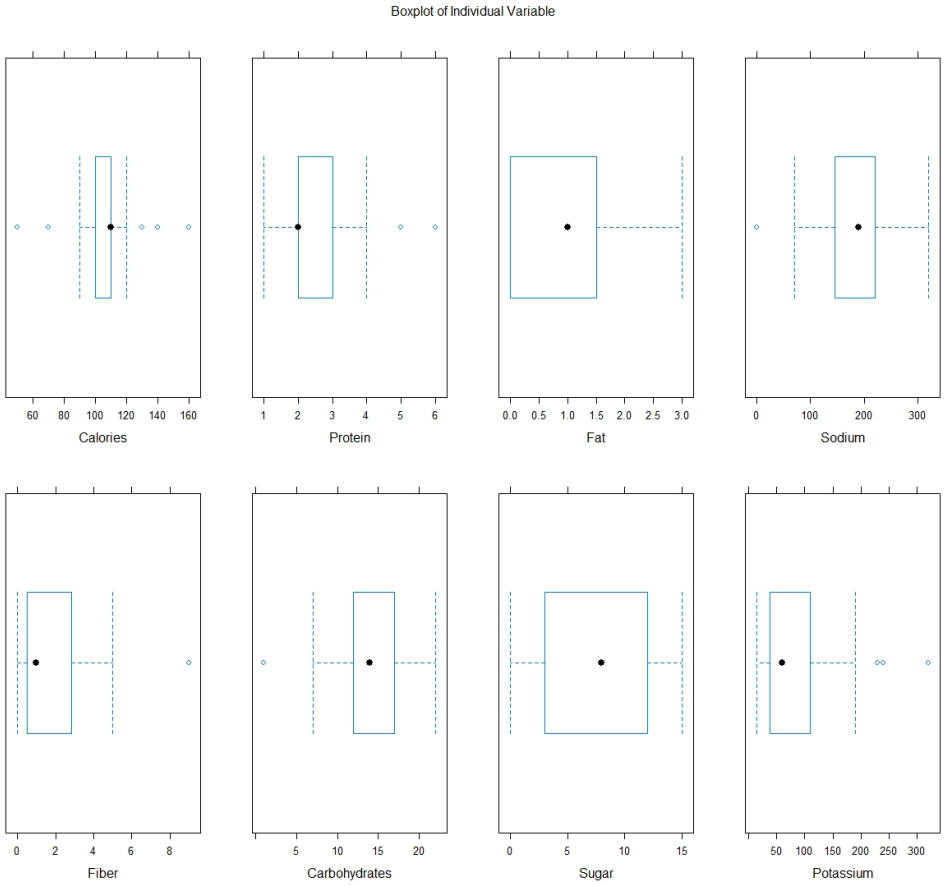

Figure 1: Boxplot of the Variables

The Correlation Heatmap of the variables are given below:

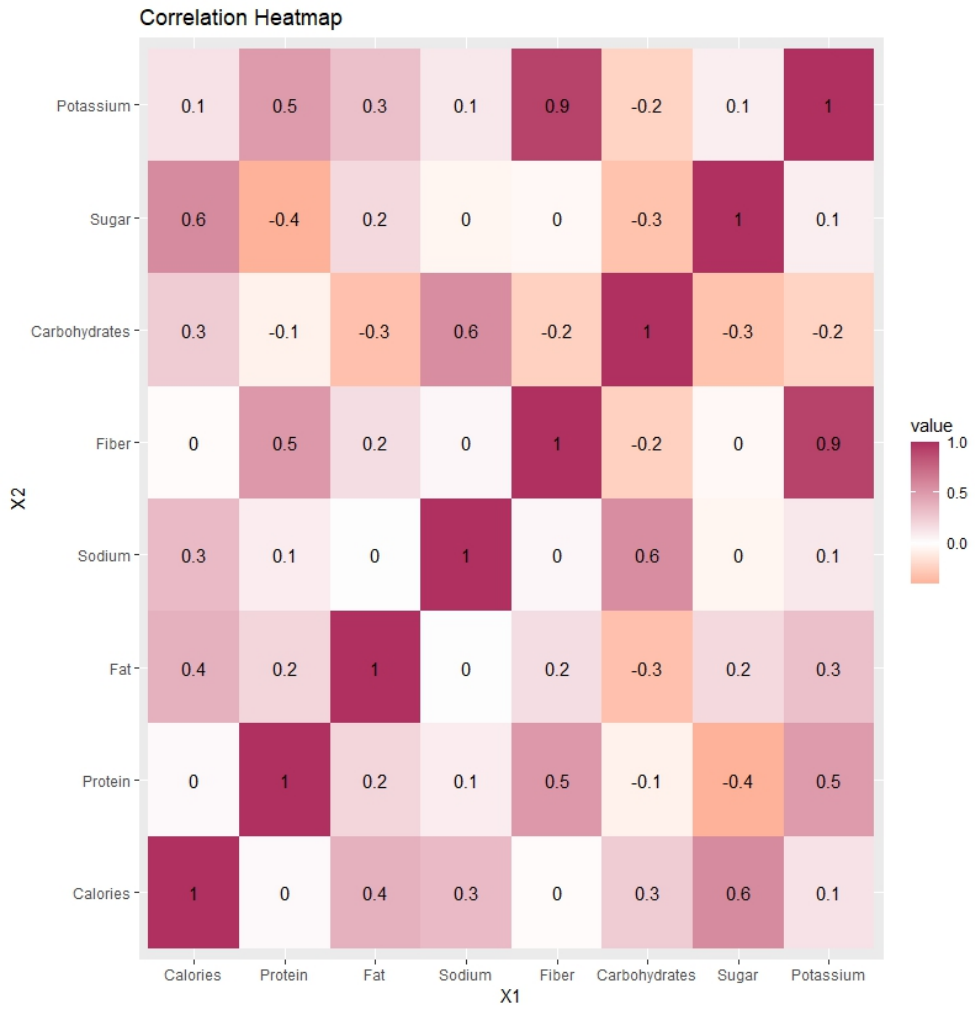

Figure 2: Correlation Heatmap of the Variables

Comment:

1. There is no missing value in the dataset.
2. One thing that is clear from the plots is that there are too many same values in some of the variables which may cause difficulties in testing. So, we may consider adding some jitter to those values.
3. There is one outlier in Sodium and Carbohydrates whereas Potassium has three

outliers. Since, the dataset contains only 43 observations, removing outliers is not a good idea. So, we restrain ourself from removing those outliers.

4. The boxplot shows outliers in the other variables also. But most of the variables has many repeated values, so we do not consider them as outliers.

5. From the boxplot we see that Sugar is symmetrically distributed, Protein, Fiber, Carbohydrates and Potassium have positively skewed distribution and Calories, Fat, Sodium have negatively skewed distribution.

6. Most of the variables are either positively correlated or uncorrelated. Carbohydrate is negatively correlated with most of the variables.

Addition of jitter

We add positive amount of jitter, as negative jitter may make the values negative which is not meaningful in the given context.

The following table shows the variables, how much jitter added to the variables and the reason for such addition of jitter:

Variables	Jitter	Reason
Calories	runif(43,0,7.8)	Many same observation
Protein	runif(43,0,.5)	Many same observation
Fat	runif(43,.1,.5)	Many same observation
Sodium	runif(43,0.001,0.005)	0 observations
Fiber	runif(43,0,2)	Many same observation
Sugar	runif(43,0.001,0.005)	0 observations

5.2 Checking for Multivariate Normality (Not Considering any Grouping Factor)

In this section we will try to check whether the individual variables follow a Multivariate Normal Distribution.

Checking for Univariate Normality

The first step is to check whether the individual variables follow a univariate normal distribution.

Given below the density plots of each variable:

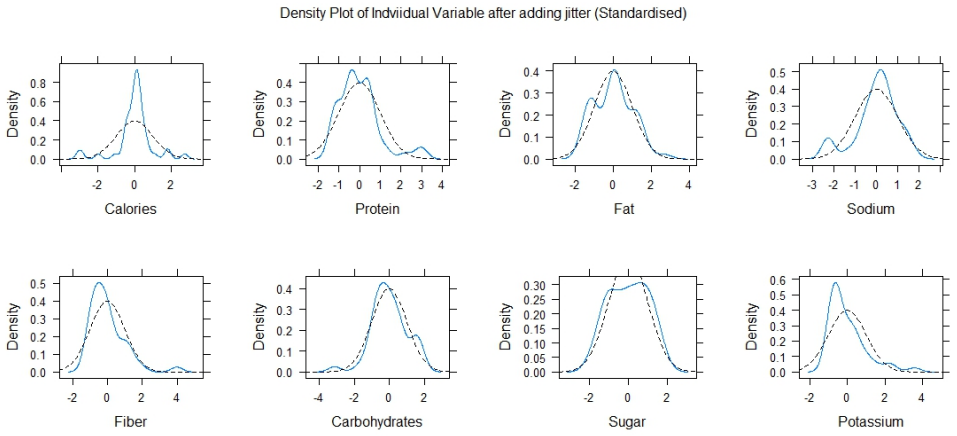

Figure 3: Density plot of the Variables

Given below the QQ plots of each variable:

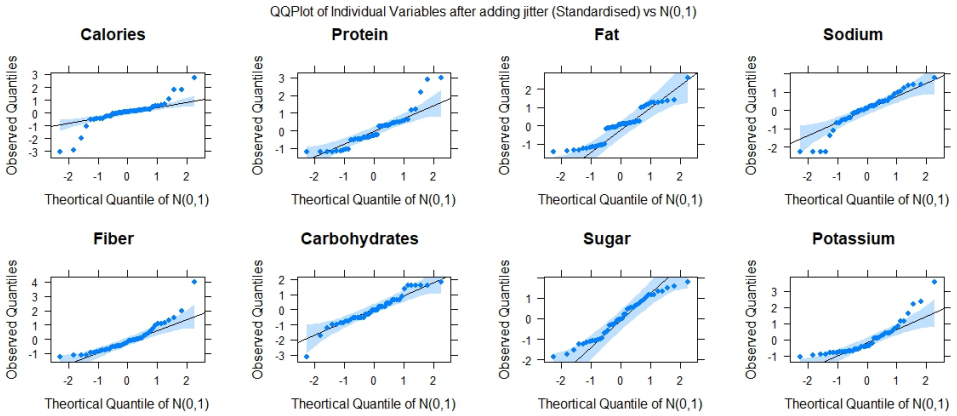

Figure 4: QQ plot of the Variables

Comment:

From Figure 3 and Figure 4 we observe Calories, Sodium, Fiber and Potassium do not seem to follow a univariate normal distribution.

Shapiro Wilk Test for Univariate Normality:

Given below the results of the test:

Variable	p.value	Decision
Calories	3.15E-05	Reject
Protein	0.001392742	Reject
Fat	0.006614561	Reject
Sodium	0.00819374	Reject
Fiber	0.000387199	Reject
Carbohydrates	0.065805603	Accept
Sugar	0.0468579	Accept
Potassium	0.000013	Reject

Note, the level of significance for the test is 0.01.

The variables for which univariate normality is rejected should be transformed so that they follow univariate normal.

Transformation of Variables

Since univariate normality is rejected for some of the variables, we use Univariate Boxcox Transformation to transform the required variables to follow a univariate normal distribution.

The following table shows variables under consideration and the power transformation applied on them:

Variable	Lambda
Calories	1.8412333
Protein	0.1225622
Fat	0.8700501
Sodium	1.6051347
Fiber	0.3700856
Carbohydrates	1.2873041
Sugar	0.8189196
Potassium	-0.1400536

Given below the QQ plot of all the variables after transformation:

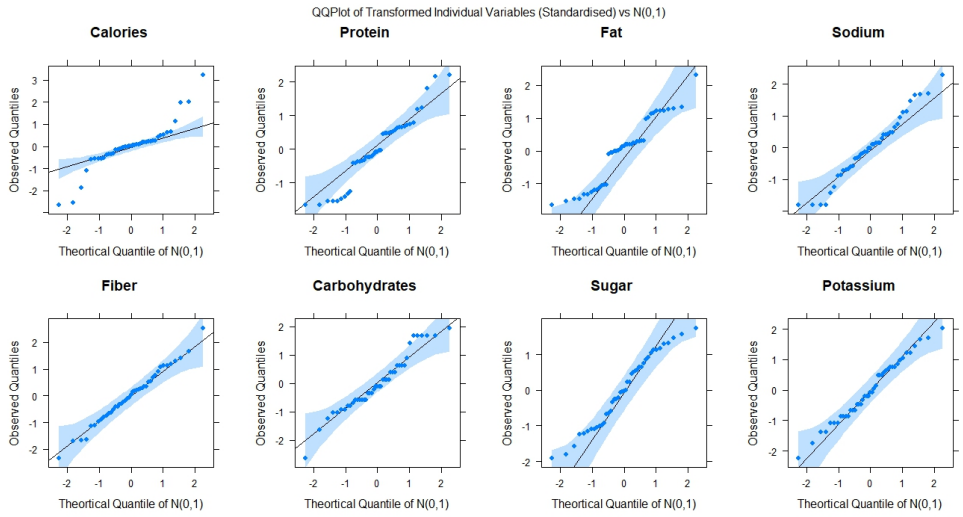

Figure 5: QQ plot of the Transformed Variables

Clearly, the QQ plots of Calories, Fat do not show much improvement after transformation.

The Shapiro Wilk Test for testing of univariate normality gives the following result:

Variable	p.value	Decision
Calories	0.0001697	Reject
Protein	0.1474622	Accept
Fat	0.0070846	Reject
Sodium	0.4565389	Accept
Fiber	0.9046218	Accept
Carbohydrates	0.1071514	Accept
Sugar	0.0591866	Accept
Potassium	0.8077318	Accept

Conclusion:

If we do not consider grouping factor, all the variables do not follow a univariate normal distribution, which implies the variables themselves do not follow a multivariate normal distribution.

5.3 Principal Component Analysis

In this section we will discuss about Principal Component Analysis of the 8 variables.

The rotation matrix for the principal components is given below:

Variables	PC1	PC2	PC3	PC4	PC5	PC6	PC7	PC8
Calories	-0.11429	0.656598	0.135123	-0.10048	0.453703	-0.13578	0.468935	0.288583
Protein	-0.42084	-0.22794	0.253063	-0.37168	0.551363	0.420374	-0.24231	-0.17124
Fat	-0.31569	0.301787	-0.16565	-0.68746	-0.43197	-0.25131	-0.12506	-0.20937
Sodium	-0.02254	0.273076	0.591711	0.075408	-0.50185	0.555796	0.086933	0.028197
Fiber	-0.55812	-0.13467	0.069649	0.376351	-0.0757	-0.19426	0.452959	-0.52463
Carbohydrates	0.238093	0.113458	0.631592	0.058778	0.131506	-0.53484	-0.36979	-0.29661
Sugar	-0.03834	0.565922	-0.36207	0.374915	0.123791	0.25364	-0.43508	-0.37102
Potassium	-0.5831	-0.0001	0.072717	0.296338	-0.11558	-0.21133	-0.40863	0.584717

The following is the biplot of first and second principal component:

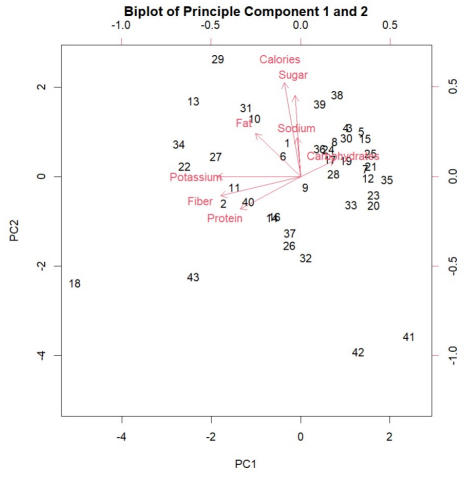

Figure 6: Biplot of First and Second principal components

Comment:

We see that effect of Potassium and Fiber are more in first principal component

than in the second principal component. Again, effect of Calories and Sugar are more in second principal component than in the first principal component. Fat, Protein, Carbohydrates have similar effect on both of the components.

The following table shows the proportion of variation explained by the principal components and the Cumulative sum of proportion of variables:

Principal Component	Proportion of variation explained	Cumulatiove Proportion of Variation
PC 1	0.318446181	31.84462
PC 2	0.231845224	55.02914
PC 3	0.221103745	77.13951
PC 4	0.108535420	87.99306
PC 5	0.062162601	94.20932
PC 6	0.044678557	98.67717
PC 7	0.007562921	99.43346
PC 8	0.005665352	100.00000

Comment:

We can see that 5 principal components are required to explain atleast 90 percent of the variability of the dataset.

Given below the Scree Plot of the 8 Principal components:

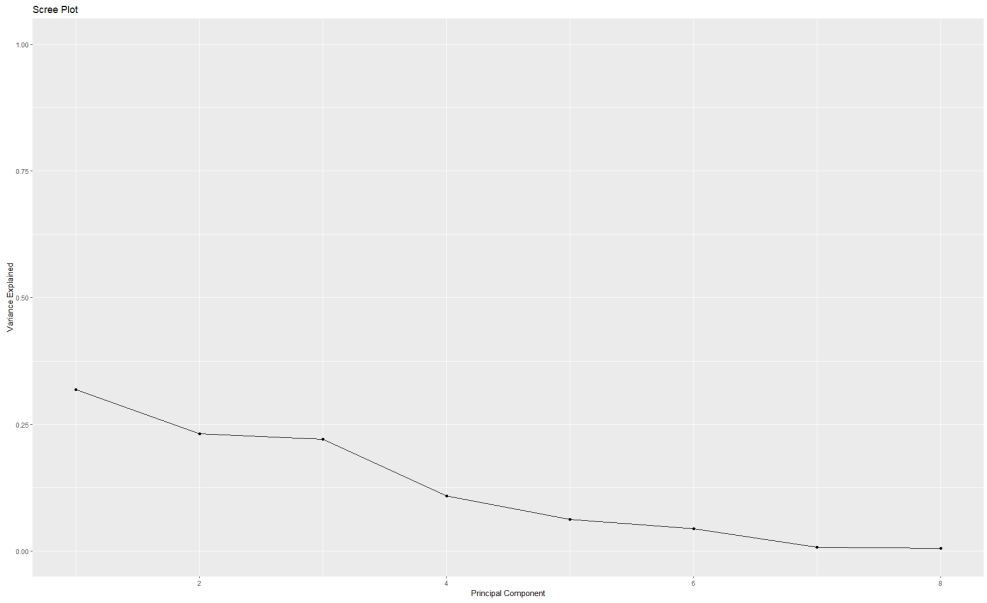

Figure 7: Scree Plot of the principal components

Comment:

Figure 7 shows that 6th, 7th and 8th principal component are unable to explain significant amount of variability in the dataset. So, it is not worthwhile to consider them.

5.4 Clubbing of Two Groups

Previously we have seen that, if we do not consider the grouping factor, the variables do not follow a Multivariate Normal Distribution. So, from this section onwards we will consider the grouping factor and see whether the data on these three groups are form three different population. But group 3 only have 6 observations which may cause trouble while performing Box's M test (as the sample covariance matrix becomes positive semi definite). So, we need to club a group (either manufacturer 1 or manufacturer 2) with Manufacturer 3 to proceed with the analysis and to perform required tests.

We have already obtained Principal Components of the variables. The first Principal Component explains the variability in the dataset the most. So, from the first principal component we extract top 3 variables which have the most absolute weights.
Clearly, Potassium, Protein and Fiber have the maximum absolute weightage on defining the first principal component. Now based on these variable we will combine two groups. Clubbing two groups is done by visualizing which of the two groups, Manufacturer 1 or Manufacturer 2 tend to form a cluster with Manufacturer 3.

Given below a 3D plot which shows the clustering of two groups:

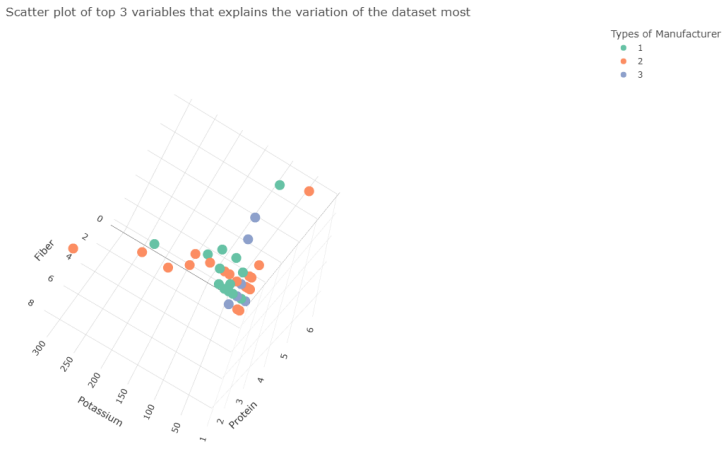

Figure 8: A shot of 3D Scatter Plot of Potassium, Fiber and Protein

As figure 1 shows, we are not being able to identity whether Group 3 forms a cluster with either of the other two variables. So, **without loss of generality, we club Group 2 or Group 3.**

5.5 Exploratory Data Analysis (Groupwise)

Here, we will analyse the data with respect to two groups: Manufacturer 1 and Manufacturer 2.

GROUP 1:

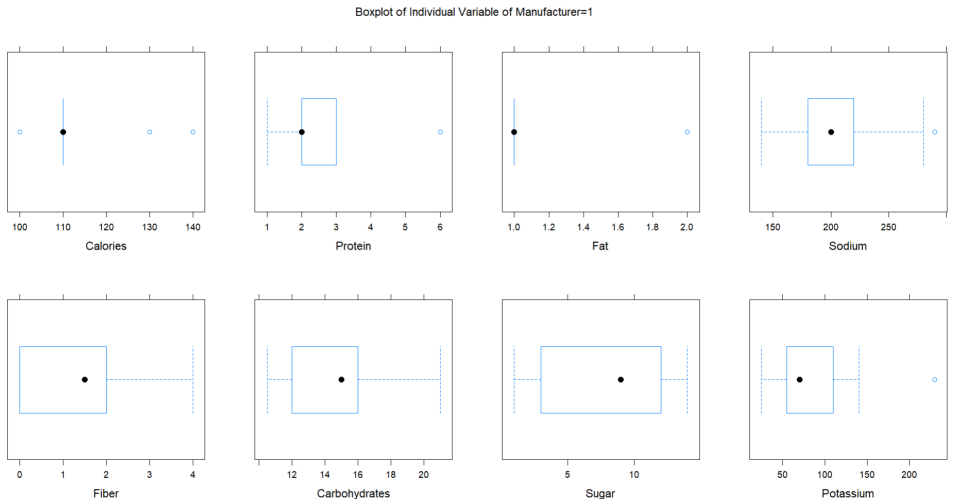

Figure 9: Boxplot of the variables

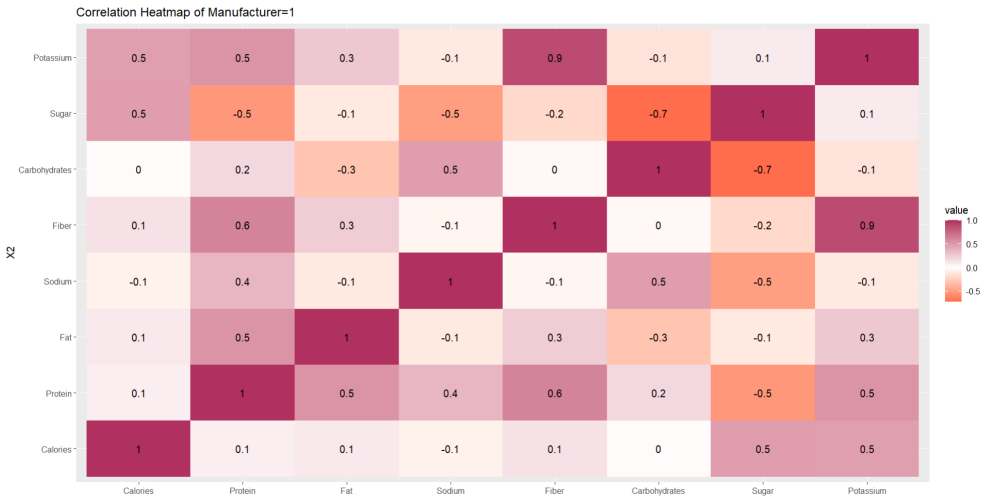

Figure 10: Correlation Heatmap of the variables

Comment:

1. There is one outlier in Sodium and Potassium.

2. The boxplot shows outliers in the other variables also. But most of the variables has many repeated values, so we do not consider them as outliers.

3. From the boxplot we see that Sodium is symmetrically distributed, Protein and Potassium have positively skewed distribution and Fiber, Carbohydrate and Sugar have negatively skewed distribution.

GROUP 2:

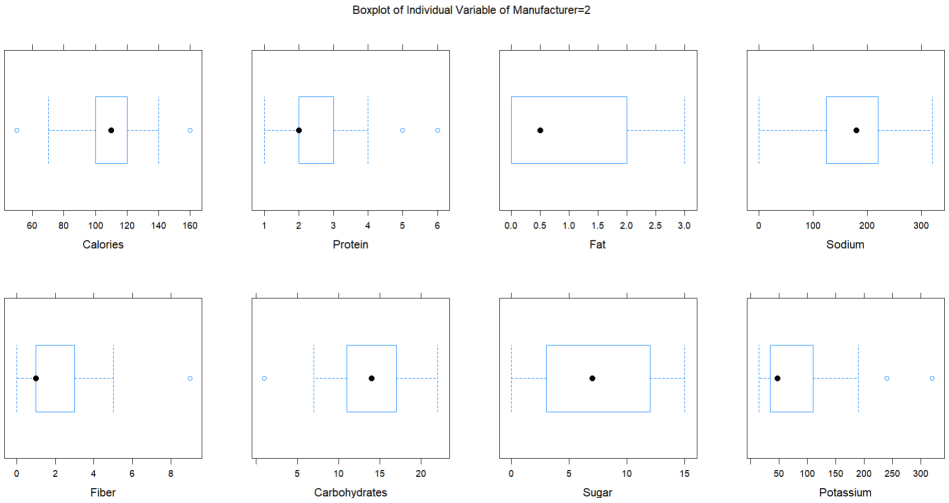

Figure 11: Boxplot of the variables

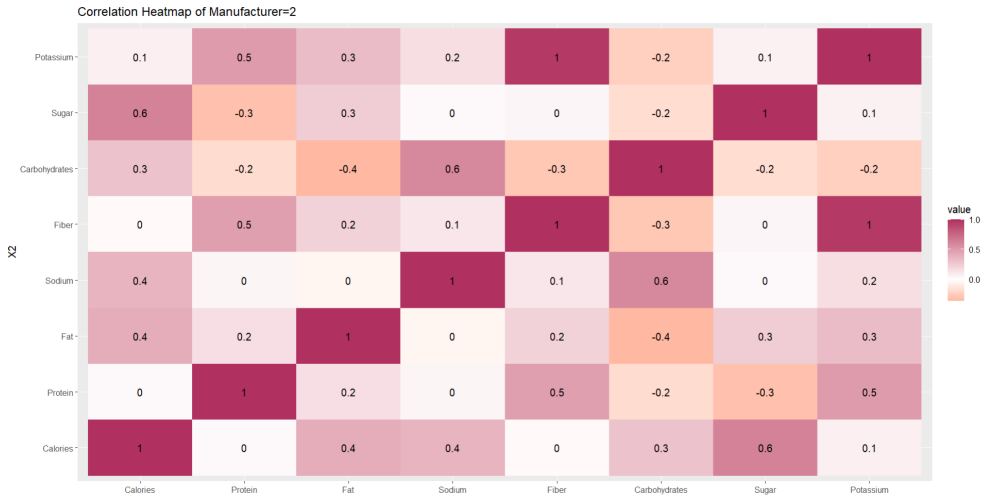

Figure 12: Correlation Heatmap of the variables

Comment:

1. Only Fat, Sodium and Sugar do not have any outliers.

2. From the boxplot we see that Calories, Sodium and Carbohydrates are symmetrically distributed, Protein, Fat, Fiber, Sugar and Potassium are positively skewed.

Observations:

1. In Group 2, almost all of the variables have more variation than that of the variables of group 1. 2. In Group 2, there is no negatively skewed variable. 3. We can see that number of uncorrelated variables is more in Group 2.

5.6 Checking for Multivariate Normality

In this section our focus is to check whether the 8 variables, within the two groups follow a Multivariate Normal Distribution. Note that,here, the data under consideration is the one which has jitter added to it.-

Checking for Univariate Normality

Our first step is check whether the variables themselves follow a univariate normal distribution.

Given below the group wise QQ plots of each of the variable:

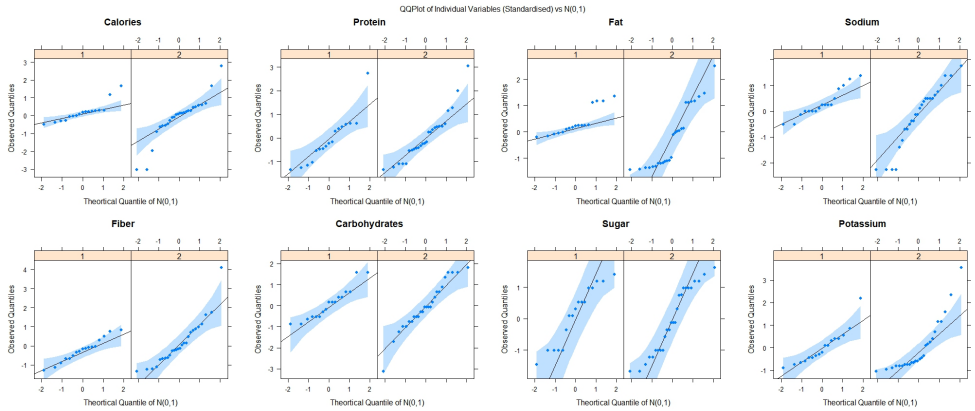

Figure 13: QQ Plot of the variables

Comment:

Except Carbohydrate and Sugar we cannot visualise normality for the rest of the variables for both of the groups.

Now, we consider Shapiro Wilk Test to see whether within a group, the variables follow a univariate normal distribution.

Note that, here the level of significance $\alpha = 0.01$

GROUP 1:

Test	Variable	Statistic	p.value	Normality
Shapiro-Wilk	Calories	0.7818	0.0012	NO
Shapiro-Wilk	Protein	0.8256	0.0047	NO
Shapiro-Wilk	Fat	0.7538	0.0005	NO
Shapiro-Wilk	Sodium	0.9144	0.1185	YES
Shapiro-Wilk	Fiber	0.9686	0.793	YES
Shapiro-Wilk	Carbohydrates	0.9159	0.1259	YES
Shapiro-Wilk	Sugar	0.9125	0.1101	YES
Shapiro-Wilk	Potassium	0.8746	0.026	YES

Note that Calories, Protein and Fat rejects the univariate shapiro wilk test for normality.

GROUP 2:

Test	Variable	Statistic	p.value	Normality
Shapiro-Wilk	Calories	0.8728	0.0041	NO
Shapiro-Wilk	Protein	0.9142	0.033	YES
Shapiro-Wilk	Fat	0.8704	0.0036	NO
Shapiro-Wilk	Sodium	0.9318	0.0855	YES
Shapiro-Wilk	Fiber	0.863	0.0026	NO
Shapiro-Wilk	Carbohydrates	0.9562	0.3231	YES
Shapiro-Wilk	Sugar	0.9224	0.0511	YES
Shapiro-Wilk	Potassium	0.7921	0.0001	NO

Note that Calories,Fat, Fiber and Sugar reject the univariate shapiro wilk test for normality.

Transformation of Variables

Since univariate normality is rejected for some of the variables, we use Boxcox Transformation to transform the required variables to follow a univariate normal

distribution.

The following table shows variables under consideration and the power transformation applied on them:

Variable	Power
Calories	1.68149156
Protein	0.32994981
Fat	0.37779543
Fiber	-0.26965518
Potassium	0.01459788

Given below the QQ plot of the transformed variables (Groupwise):

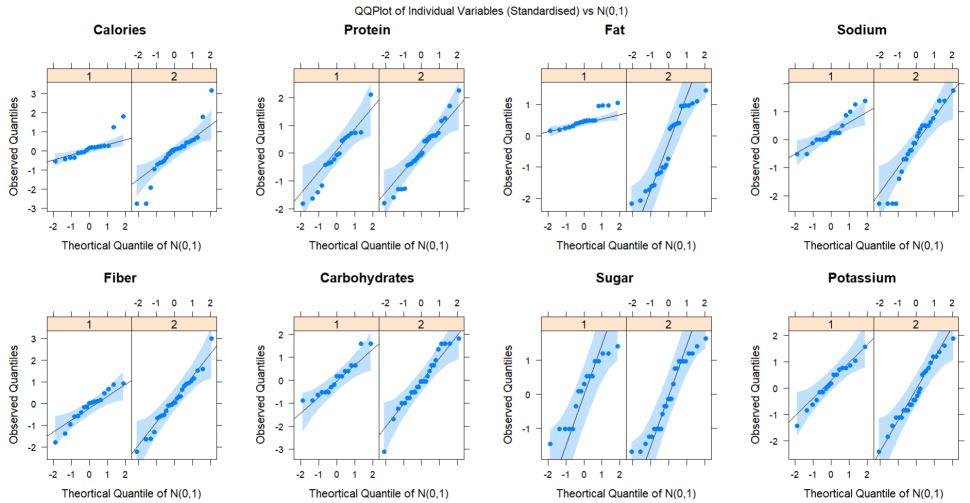

Figure 14: QQ Plot of the variables after appropriate transformation

Comment:

Except Calories and Fat the rest of the variables seems to follow univariate normal distribution.

Royston Test for Multivariate Normality

After the transformation of variables we now apply the Roystone Test for Multivariate Normality.

The results of the test is given below:

GROUP 1:

Test	H	Pvalue	MVN
Royston	26.1739	.00031	NO

Hence, we conclude that for Group 1 the variables do not follow a multivariate normal distribution.

Gamma Plot:

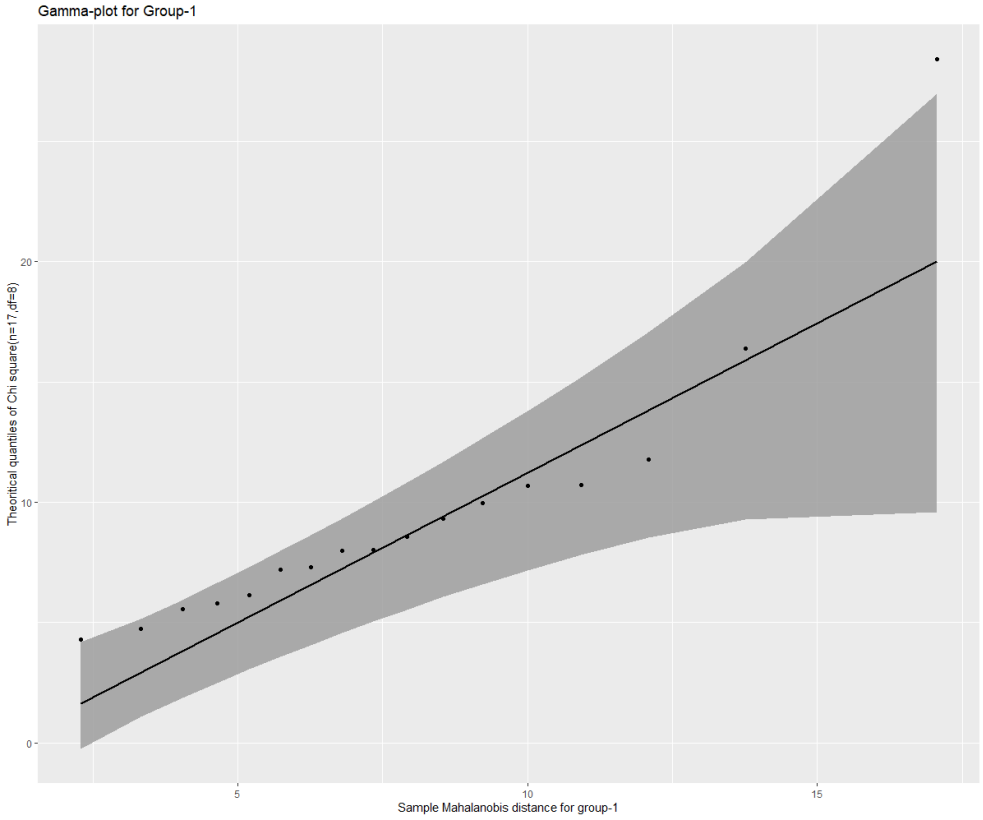

Figure 15: Gamma Plot

From the Gamma Plot, we may consider that the variables follow a Multivariate Normal Distribution.

GROUP 2:

Test	H	Pvalue	MVN
Royston	19.49517	.01044421	YES

Hence, we conclude that for Group 2 the variables follow a multivariate normal distribtuion.

5.7 Test for Equality of Covariance Matrices

Since we have sample sample size in each group, we perform Box's M test for homogeneity of covariance matrices.

Results:

Observed value of Test Statistic = 64.851

Degrees of Freedom = 36

Level of Significance = 0.01

P-Value = .002238

Hence, we conclude that the covariance matrices are not same.

5.8 Quadratic Discriminant Analysis

Now we are interested in separation and allocation problem.

Since, the variance covariance matrix of two groups (Manufacturer 1 and 2) are not equal, here we will do Quadratic Discriminant Analysis (QDA).

Here we will find the regions by minimizing Expected Cost of Misclassification (ECM).

We assume the cost of misclassifications are equal.

Set up

Here we consider two normal distributions π_1 and π_2.

$$\pi_1 : N(\underset{\sim}{\mu_1}, \sum_1)$$
$$\pi_2 : N(\underset{\sim}{\mu_2}, \sum_2)$$

Let, R_1 and R_2 be two separation regions. Then R_1 and R_2 is given by,

$$R_1 : \{\underset{\sim}{x} : -\frac{1}{2}\underset{\sim}{x}'(\sum_1^{-1} - \sum_2^{-1})\underset{\sim}{x}' + (\underset{\sim}{\mu_1}'\sum_1^{-1} - \underset{\sim}{\mu_2}'\sum_2^{-1}) - K \geq \log(\frac{p_2}{p_1})\}$$
$$R_2 : \{\underset{\sim}{x} : -\frac{1}{2}\underset{\sim}{x}'(\sum_1^{-1} - \sum_2^{-1})\underset{\sim}{x}' + (\underset{\sim}{\mu_1}'\sum_1^{-1} - \underset{\sim}{\mu_2}'\sum_2^{-1}) - K < \log(\frac{p_2}{p_1})\}$$

Where, $K = \frac{1}{2}\log(\frac{|\sum_1|}{|\sum_2|}) + \frac{1}{2}(\underset{\sim}{\mu_1}'\sum_1^{-1}\underset{\sim}{\mu_1} - \underset{\sim}{\mu_2}'\sum_2^{-1}\underset{\sim}{\mu_2})$
and p_1, p_2 are the prior probabilities of Population π_1 and π_2 respectively.

Now we will check how good the classification formula is. So we calculate Apparent Error Rate (APER). The expression of APER is :

$$APER = \frac{n_{1M} + n_{2M}}{n_1 + n_2}$$

Where,

n_{iM} : Number of misclassified observations in π_i.

n_i : Number of observations in π_i, i = 1,2.

But there is a problem that it tends to underestimate the Actual Error Rate (AER).

We can solve this by two methods :

1. Spliting the dataset into training and validation sample.

But the number of observations is 43, which is quite small. Hence we do not use this method.

2. A reasonable approach that works for most classification rule is the leave one out cross validation.

Using this method , we get APER as 30.23 percent which is moderately good.

Now we will draw the separation region based on R_1 and R_2.

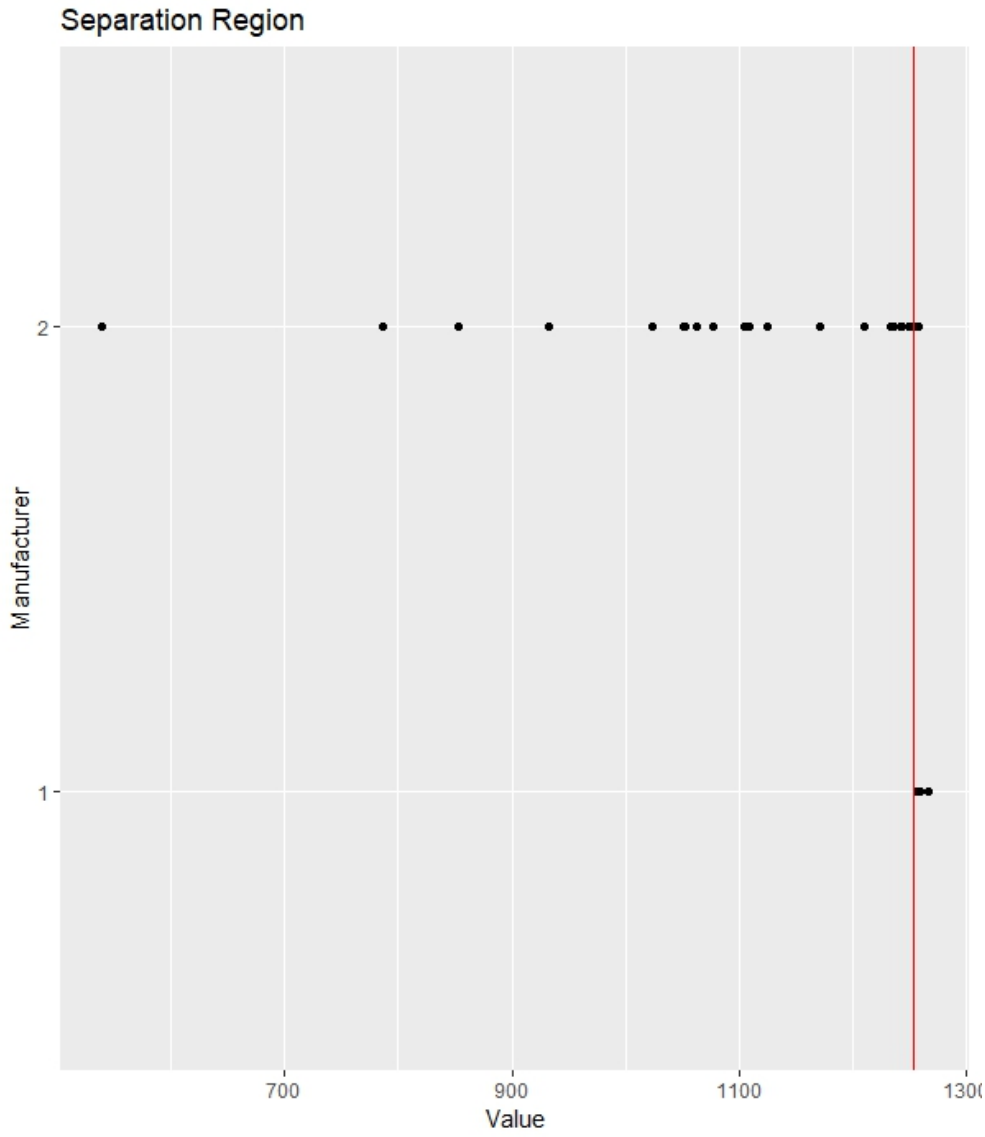

Figure 16: Separation Region obtained by QDA

Here, the above figure shows the 2 separation region. R_1 is the region left of the red line and R_2 is the region right of the red line.

Comment: From figure 1, we notice that the data is quite well separated.

The Confusion Matrix is given below:

	Population 1(Predicted)	Population 2(Predicted)
Population 1(Actual)	5	12
Population 2(Actual)	5	21

5.9 Factor Analysis

A brief theoretical glimpse which is implemented through our data analysis:

We have a dataset which contains 8 variables $\underset{\sim}{X}=(X_1, X_2, ..., X_8)$ (continuous), so the purpose of doing factor analysis is to write $\underset{\sim}{X}$ as a linear combination of m common factors $F_1, ..., F_m$ and some specific factors .

Suppose $\underset{\sim}{X}$ has finite mean $\underset{\sim}{\mu}$ and dispersion matrix $\sum$ and the model is given by:

$$\underset{\sim}{X} - \underset{\sim}{\mu} = LF + \underset{\sim}{\varepsilon}$$

Assumptions :

• E[$\underset{\sim}{F}$]=0

• Cov($\underset{\sim}{F}$)=I_{mxm}

• E[$\underset{\sim}{\varepsilon}$]=0

• cov[$\underset{\sim}{\varepsilon}$]= ψ where ψ is a diagonal matrix whose diagonal entries are the specific variances also termed as the uniqueness.

• Cov($\underset{\sim}{F}$, $\underset{\sim}{\varepsilon}$)= $\underset{\sim}{0}_{mxp}$

This factor model is known as Orthogonal Factor Model(OFM) .

The difference between factor model and the linear model is that in the equation above the right side is entirely unobservable , i.e the loadings are unknown we need to estimate them. So for estimating them we have three methods:

• PC METHOD

• ITERATIVE PC METHOD

• MLE

Further we perform **Princpal Factor Analysis (PFA)** when the value of s >0 , where s= difference in the number of parameters when $\sum$ is unconstrained and $\sum$ in OFM.

Sometimes data is summarised by the correlation matrix R instead of the dispersion matrix $\sum$. In this case the communalities and uniqueness are given by the formula :

• Communality $=\sum_{j=1}^{m} l_{ij}{}^2$ where l_{ij} is the loading of the ith variable on the jth common factor.

• Uniqueness (Specific Variances) = 1- Communality

R- $\widetilde{\psi}$ is known as the **Reduced Correlation Matrix.**

We can perform Factor Rotation , if we perform Varimax rotation it finds the orthogonal transformation of the loading matrix but the specific variances and so the communalities also remains the same, thus rotation leads to change the factor loadings only. Apart from this we also have Quartimax rotation , Promax rotation. Manytimes we need the estimated factors known as the factor scores which are often used for diagonostic purposes as well as inputs of other studies.

There are several methods to calculate the scores :

• Bartletts's Weighted Least Square method

• Regression method

Implementing the theory in the data set:

We are carrying out our data analysis through the R software and the function which helps to do Factor Analysis in R is

`factanal ()`

The function is available in the build -in

`stats`

package.

Description of the function factanal() :-

• The function performs maximum likelihood factor analysis on a covariance matrix or data matrix.

• The number of factors to be fitted in specified by the argument factors .

• The factor scores may be calculated either by using **Thompson's estimator or Bartlett's weighted least-squares scores method**. The particular method is specified by an additional argument **scores="regression" or scores="Bartlett".**

• There is an additional argument **"rotation"** , the transformation of the factors may be specified by either **rotation="varimax" for orthogonal rotation, rotation="Bartlett" for oblique rotation or rotation="none"** .

Now diving into our dataset we have p=8 many variables and n=43 observations. The first question comes into our mind that how many factors are required to explain the majority of variability in the model?

Answer of this can be the PCA approach , from the section where we have discussed PCA we note that 4 principal components were needed to explain 87 percent of the variability which is quite high so we can take the number of factors to be 4.

```
> f1=factanal(data4[,-1],factors=4)
> f1
```

```
Call:
factanal(x = data4[, -1], factors = 4)


Uniquenesses:
     Calories       Protein           Fat
        0.273         0.322         0.227
       Sodium         Fiber
        0.582         0.165
Carbohydrates         Sugar     Potassium
        0.005         0.005         0.104


Loadings:
              Factor1 Factor2 Factor3 Factor4
Calories                0.454   0.644   0.318
Protein         0.725          -0.329   0.201
Fat             0.174           0.194   0.838
Sodium                  0.635
Fiber           0.906
Carbohydrates -0.110   0.949  -0.144  -0.249
Sugar         -0.154  -0.172   0.963   0.120
Potassium       0.842           0.119   0.413

                Factor1 Factor2 Factor3 Factor4
SS loadings       2.131   1.554   1.526   1.107
```

```
Proportion Var   0.266   0.194   0.191   0.138
Cumulative Var   0.266   0.461   0.651   0.790

Test of the hypothesis that 4 factors are sufficient.
The chi square statistic is 6.02 on 2 degrees of freedom.
The p-value is 0.0492
```

Observations :

1. From the previous output we can see the values of uniqueness it corresponds to the proportion of variability which cannot be explained by a linear combination of the factors and here this proportion is quite low for each of the variable so the factors are performing well in explaining the variability.
2. The next section is the loadings, which range from -1 to 1. The loadings are the contribution of each original variable to the factor. Variables with a high loading are well explained by the factor.

Next we are interested to see the communalities .

```
     Calories       Protein           Fat        Sodium         Fiber
    0.7270756     0.6782354     0.7734081     0.4182931     0.8353868
Carbohydrates         Sugar     Potassium
    0.9950079     0.9950153     0.8959673
```

3. By squaring the loading we compute the fraction of the variable's total variance explained by the factor. This proportion of the variability is denoted as communality. Another way to calculate the communality is to subtract the uniquenesses

from 1. An appropriate factor model results in low values for uniqueness and high values for communality.

4. And in our model we see that the values of the communalities are really high and that of uniqueness are low so our factor model fits well.

5. The table beneath the loadings shows the proportion of variance explained by each factor. The row Cumulative Var gives the cumulative proportion of variance explained. These numbers range from 0 to 1. The row Proportion Var gives the proportion of variance explained by each factor, and the row SS loadings gives the sum of squared loadings. This is sometimes used to determine the value of a particular factor. A factor is worth keeping if the SS loading is greater than 1 (Kaiser's rule). And here all the SS are >1 so all the 4 factors are important.

6. The last section of the function output shows the results of a hypothesis test. The null hypothesis, H_0, is that the number of factors in the model is sufficient, in our analysis 4 factors, is sufficient to capture the full dimensionality of the data set. Conventionally, we reject H_0 if the p-value is less than 0.01. Such a result indicates that the number of factors is too small. In contrast, we do not reject H_0 if the p-value exceeds 0.01. Such a result indicates that there are likely enough (or more than enough) factors capture the full dimensionality of the data set. The high p-value in our example above leads us to not reject the H0, and indicates that we fitted an appropriate model.

Next we compute the Residual matrix : (TABLE 1)

	Calories	Protein	Fat	Sodium	Fiber	Carbohydrates	Sugar	Potassium
Calories	2.97E-09	0.053298	0.002746	-0.05796	0.005837	0.000359779	0.000426	-0.019128273
Protein	0.053298	-1.00E-06	-0.11393	-0.01858	-0.00573	-0.001052599	-0.00096	-0.002132675
Fat	0.002746	-0.11393	-6.50E-07	0.077001	-0.00832	-0.000537775	-0.00092	0.040296418
Sodium	-0.05796	-0.01858	0.077001	4.41E-08	-0.02299	0.00109556	0.001021	0.029352022
Fiber	0.005837	-0.00573	-0.00832	-0.02299	-2.30E-07	7.29E-06	-0.00012	0.002459063
Carbohydrates	0.00036	-0.00105	-0.00054	0.001096	7.29E-06	-1.37E-05	-1.50E-05	0.00025147
Sugar	0.000426	-0.00096	-0.00092	0.001021	-0.00012	-1.54E-05	-1.60E-05	0.000302333
Potassium	-0.01913	-0.00213	0.040296	0.029352	0.002459	0.00025147	0.000302	-1.82E-07

Observation:

Numbers close to 0 indicate that our factor model is a good representation of the underlying concept. Here maximum of them are close to 0 rather in $\exp -4$ which is very close to 0 so our factor model is good.

Now we are interested in factor rotation , let us fit three factor models, one with no rotation, one with varimax rotation, and one with promax rotation. We see that the residual matrix, communalities as a result the uniqueness do not change when there is no rotation and when there is a varimax rotation which matches the theory too.

The Residual Matrix for varimax rotation: (TABLE 2)

	Calories	Protein	Fat	Sodium	Fiber	Carbohydrates	Sugar	Potassium
Calories	2.97E-09	0.053298	0.002746	-0.05796	0.005837	0.000359779	0.000426	-0.019128273
Protein	0.053298	-1.00E-06	-0.11393	-0.01858	-0.00573	-0.001052599	-0.00096	-0.002132675
Fat	0.002746	-0.11393	-6.50E-07	0.077001	-0.00832	-0.000537775	-0.00092	0.040296418
Sodium	-0.05796	-0.01858	0.077001	4.41E-08	-0.02299	0.00109556	0.001021	0.029352022
Fiber	0.005837	-0.00573	-0.00832	-0.02299	-2.30E-07	7.29E-06	-0.00012	0.002459063
Carbohydrates	0.00036	-0.00105	-0.00054	0.001096	7.29E-06	-1.37E-05	-1.50E-05	0.00025147
Sugar	0.000426	-0.00096	-0.00092	0.001021	-0.00012	-1.54E-05	-1.60E-05	0.000302333
Potassium	-0.01913	-0.00213	0.040296	0.029352	0.002459	0.00025147	0.000302	-1.82E-07

Cleary, TABLE 2 is nothing but TABLE 1.

Communality when no rotation is done:

```
     Calories       Protein           Fat        Sodium         Fiber
    0.7270756     0.6782354     0.7734081     0.4182931     0.8353868
Carbohydrates         Sugar     Potassium
    0.9950079     0.9950153     0.8959673
```

Communality when orthogonal rotation is done:

```
     Calories       Protein           Fat        Sodium         Fiber
    0.7270756     0.6782354     0.7734081     0.4182931     0.8353868
Carbohydrates         Sugar     Potassium
    0.9950079     0.9950153     0.8959673
```

Specific variances when no rotation is done:

```
     Calories       Protein           Fat        Sodium         Fiber
    0.1689147     0.3063902     0.4899276     0.5988647     0.2340993
Carbohydrates         Sugar     Potassium
    0.0050000     0.0050000     0.1433619
```

Specific variances when orthogonal rotation is done:

```
     Calories       Protein           Fat        Sodium         Fiber
    0.1689147     0.3063902     0.4899276     0.5988647     0.2340993
Carbohydrates         Sugar     Potassium
    0.0050000     0.0050000     0.1433619
```

Now we make a scatter plot of the first and second loadings and try to interpret the factors;

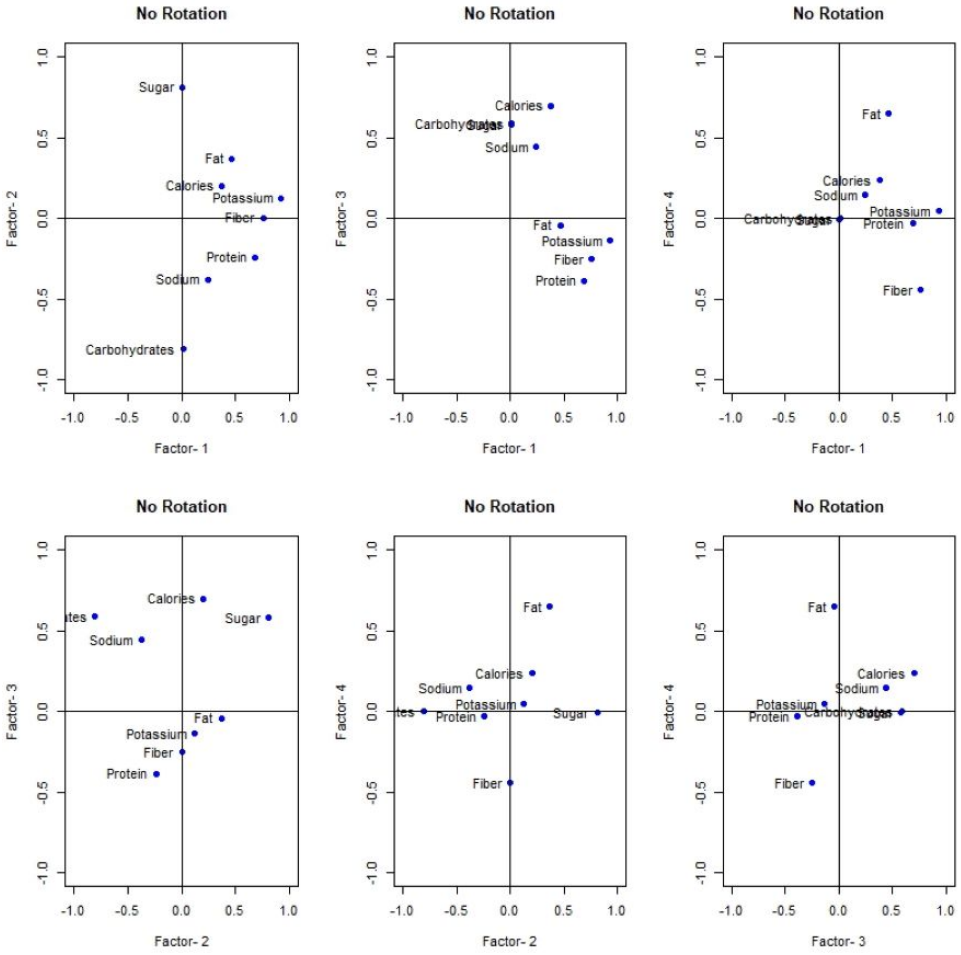

Figure 17: Scatter Plot of first and second loading when no rotation is done

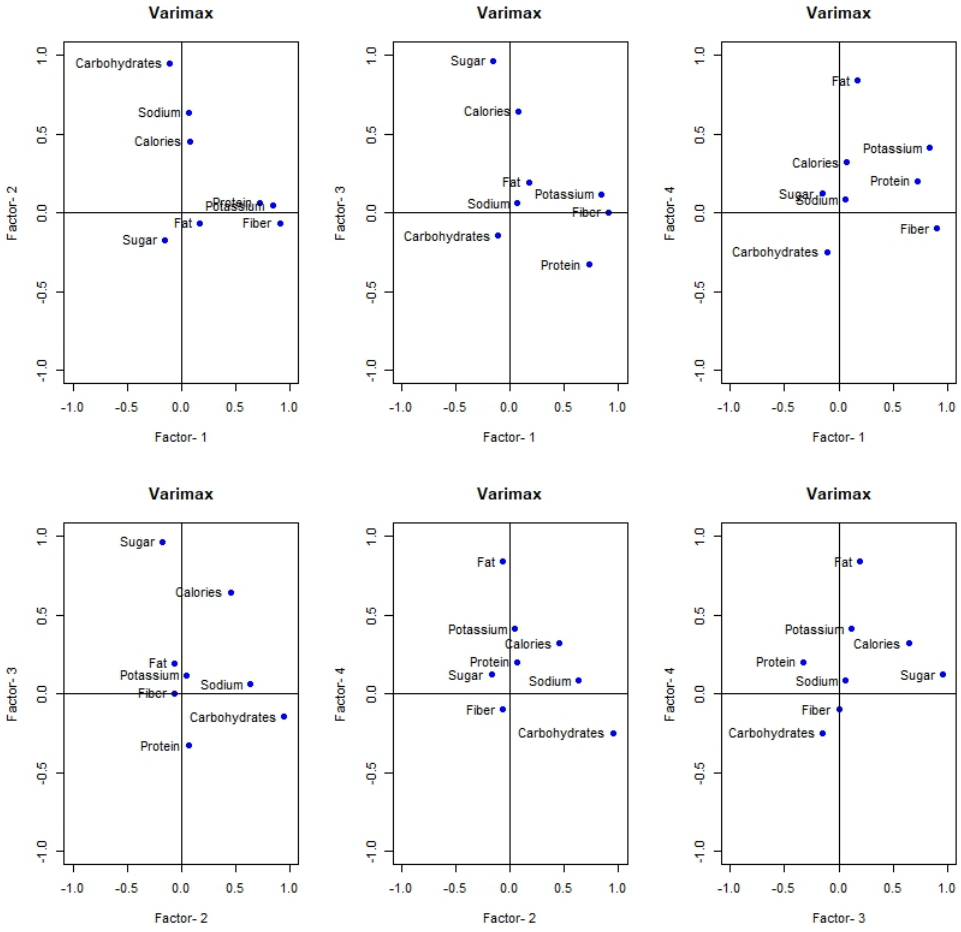

Figure 18: Scatter Plot of first and second loading when varimax rotation is done

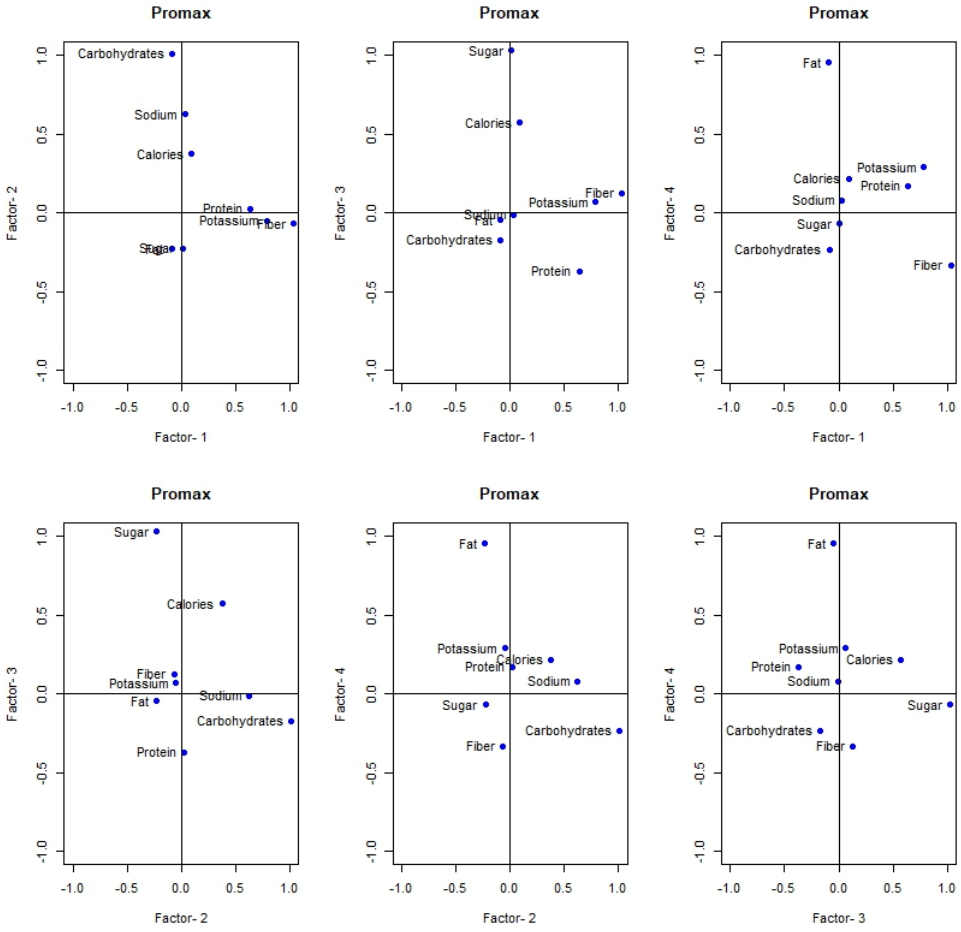

Figure 19: Scatter Plot of first and second loading when promax rotation is done

Interpretation of the factors:

If two variables both have large loadings for the same factor, then we know they have something in common. As a researcher we have to understand the data and its meaning in order to give a name to that common ground. In the plots above we can see that there are variables having the same loading for a particular factor that means both the variables if present in high content in a cereal has an impact or vice versa.

6 Confidence Ellipsoids

Constructing Bonferroni Simultaneous intervals and comparing them with T^2 interval:

Here our main motive is to draw a confidence ellipsoid for a bivariate normal distribution and construct the simultaneous confidence interval and T^2 interval and make a comparison among them.

Through out our analysis on checking normality we can see that Fibre and Potassium are two variables which are showing a very high p value in MVN test, i.e from here we can infer that they are strongly normal. Further if we consider them together we see that the in MVN test multivariate normality is getting accepted so we consider them to be bivariate normal as there are only two variables with mean μ_1 and μ_2.
So we can construct a confidence ellipsoid for (μ_1, μ_2) by the equation

$$n(\underset{\sim}{\tilde{x}} - \underset{\sim}{\mu})' {S_u}^{-1} (\underset{\sim}{\tilde{x}} - \underset{\sim}{\mu}) \leq \frac{(n-1)pF_{p,n-p}}{n-p}$$

$$Where, \underset{\sim}{\mu} = (\mu_1, \mu_2)'$$

and plot it in R using the confidence Ellipse function in

`MVQuickGraphs`

package. Further we also plot the Bonferroni simultaneous confidence intervals forμ_1 and μ_2 given by,

$$\tilde{x_1} - \sqrt{\frac{s_{11}}{n}} t_{n-1,\frac{\alpha}{2}} \leq \mu_1 \leq \tilde{x_1} + \sqrt{\frac{s_{11}}{n}} t_{n-1,\frac{\alpha}{2}}$$
$$\tilde{x_2} - \sqrt{\frac{s_{22}}{n}} t_{n-1,\frac{\alpha}{2}} \leq \mu_2 \leq \tilde{x_2} + \sqrt{\frac{s_{22}}{n}} t_{n-1,\frac{\alpha}{2}}$$

and also plot the T^2 intervals given by

$$\tilde{x_1} - \sqrt{\frac{s_{11}}{n}} t_{n-1,\alpha} \leq \mu_1 leq \tilde{x_1} + \sqrt{\frac{s_{11}}{n}} t_{n-1,\alpha}$$
$$\tilde{x_2} - \sqrt{\frac{s_{22}}{n}} t_{n-1,\alpha} \leq \mu_2 leq \tilde{x_2} + \sqrt{\frac{s_{22}}{n}} t_{n-1,\alpha}$$

to draw a comparison.

In the plot we have variable Fibre in the X axis and Potassium in the Y axis. The bold black horizontal lines are the confidence interval for μ_2 and the dashed lines are the confidence interval for μ_1 .
The red lines are the Bonferroni simultaneous intervals for μ_1 and μ_2 respectively. We see that the Bonferroni simultaneous intervals are much more precise than the T^2 intervals i.e it gives a much more precise range for the unknown parameters than the T^2 intervals.
Hotelling T^2 intervals for μ_1 is (0.1737169,1.786926) and μ_2 is (2.291618,2.677915)
Bonferroni corrected intervals for μ_1 is (0.5399971,1.42065) and μ_2 is (2.379327,2.59021)

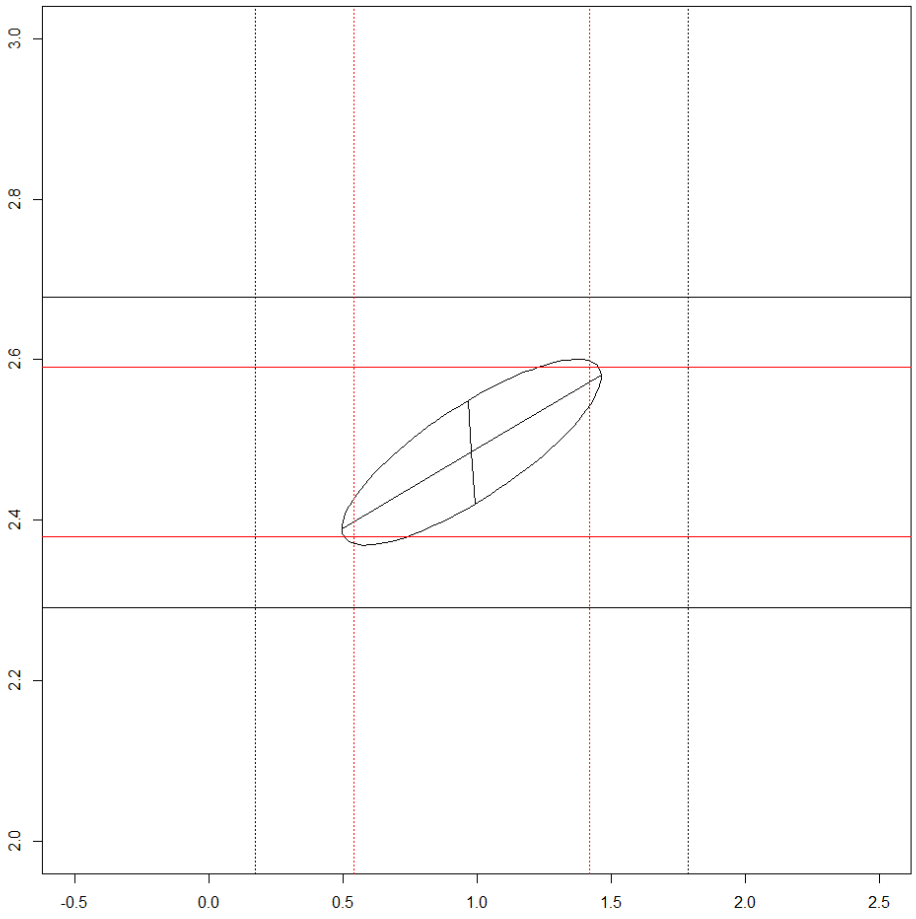

Figure 20: Confidence Interval

Note, the x axis is corresponding to μ_1 and the y axis is corresponding to μ_2. The red line indicates the Bonferroni Corrected Confidence Intervals for μ_1 and μ_2. The black line indicates the Individual Confidence Intervals for μ_1 and μ_2.

7 Acknowledgement

We are very grateful to Professor Dr. Swagata Nandi for guiding us throughout the project. She guided us with her valuable advice whenever we were stuck with the project. She provided us with references that helped us complete this project and also helped us gaining knowledge about different aspects of theoretical statistics and its application.

We would like to thank our Professor and our friends as well. They helped us understanding some of the concepts as well as codes whenever we got stuck.

This project has definitely helped us understanding how real life data actually is and how to analyse such data with all the theory we study in our class. Without a proper hands on application, theory is just a bunch of equations and literature, especially is statistics. This project helped us connecting the real life with the theory of statistics.

8 Appendix

Listing 1: R Code used for the Analysis

```
rm(list=ls())
set.seed(seed=2023)
## Calling packages-----------------
library(plotly)
library(lattice)
library(MASS)
library(ggplot2)
library(latticeExtra)
library(gridExtra)
library(car)
library(tactile)
library(EnvStats)
library(dplyr)
library(reshape)
library(RVAideMemoire)
library(heplots)
library(qqplotr)
library(MVN)
library(biotools)
library(forecast)
library(MVQuickGraphs)
```

```
22 library(qqplotr)
23
24 ## calling the dataset---------------
25 setwd("D:/ISI/Sem-2/Swagata Nandi")
26 B=as.data.frame(read.csv(file="cereal.csv"))
27 attach(B)
28
29 ## EDA of the dataset----------------------
30 # 1. boxplot
31 eda1 = list()
32 for(i in 1:8)
33 {
34   eda1[[i]] = bwplot(as.matrix(B[,-c(1,2)][i]),
35                      xlab=paste(colnames(B)[i+2]))
36 }
37
38 do.call(grid.arrange, c(eda1, ncol = 4,nrow=2,
39                         top='Boxplot of Individual
                               Variable') )
40 # 2. Summary
41 summary(B[,-c(1,2)])
42
43 # 3. Correlation heatmap
44 corr = data.matrix(cor(B[,-c(1,2)]) )
45 mel = melt(corr)
46 ggplot(mel, aes(X1,X2))+geom_tile(aes(fill=value)) +
```

```
47   geom_text(aes(label = round(value, 1)))+
48   scale_fill_gradient2(low='red',mid = 'white' ,high='
       maroon')+
49   labs(title="Correlation Heatmap")
50
51 ## Normality Checking-------------------
52 # Adding some jitter to the variables as there are many
      repeatations
53 cal.jit=B[,3]+runif(43,0,7.8)
54 fat.jit=B[,5]+runif(43,.1,.5)
55 protein.jit=B[,4]+runif(43,0,.5)
56 fiber.jit=B[,7]+runif(43,0,2)
57 sugar.jit=B[,9]+runif(43,0,4)
58 sodium.jit=B[,6]+runif(43,0,5) #for 0 values
59
60 # original data frame modified by adding jitter
61 A=data.frame(Brand.of.cereal=B[,1],Manufacturer=B[,2],
62              Calories=cal.jit,Protein=protein.jit,Fat=
                   fat.jit,
63              Sodium=sodium.jit,Fiber=fiber.jit,
                   Carbohydrates=B[,8],
64              Sugar=sugar.jit,Potassium=B[,10])
65 A
66 attach(A)
67
```

```
68 # Density plot of individual variable after adding noise
       to the variables(not considering the grouping)
69 plist.j = list()
70 for(i in 1:8)
71 {
72   plist.j[[i]] = densityplot(~scale(A[,i+2]),data=A,
73                              xlab=paste(colnames(A)[i
                                  +2]),plot.points=F,rev=F
                                  )+
74     layer(panel.curve(dnorm(x),lty=2))
75 }
76 do.call(grid.arrange, c(plist.j, ncol = 4,nrow=2,
77                         top='Density Plot of Indviidual
                               Variable after adding jitter
                               (Standardised)'))
78
79 # QQplot of individual variables
80 plist_qq.j = list()
81 for(i in 1:8){
82   plist_qq.j[[i]]=qqmath(~ scale(A[,i+2]),data = A,
83                          prepanel = prepanel.qqmathline,
84                          panel = function(x, ...) {
85                            panel.qqmathci(x, ...)
86                            panel.qqmathline(x, ...)
87                            panel.qqmath(x, ...)
```

```
88                          },pch=19,xlab="Theortical
                               Quantile of N(0,1)",
89                          ylab="Observed Quantiles",
90                          main=paste(colnames(A)[i+2]))
91 }
92
93 do.call(grid.arrange, c(plist_qq.j, ncol = 4,nrow=2,
94                         top="QQPlot of Individual
                               Variables after adding jitter
                                (Standardised) vs N(0,1)") )
95
96 # Shapiro test----------------------
97 Shap1=matrix(0,nrow=8,ncol=3)
98 Shap1[,1]=colnames(B)[3:10]
99 for(i in 1:8)
100 {
101   Shap1[i,2]=shapiro.test(A[,i+2])$p
102 }
103 Shap1[,3]=ifelse(as.numeric(Shap1[,2])<0.01,"Reject","
      Accept")
104 colnames(Shap1)=c("Variable","p value","Decision")
105 Shap1
106
107
108 ## Transformation of variables (not group wise)
      ---------------------
```

```
# Boxcox Transformation
bc <- function(data, var, optimize = FALSE){
  var <- as.character(substitute(var))
  fmla <- reformulate("1", var)
  lmvar <- lm(formula = fmla, data = as.data.frame(data)
     , x = TRUE, y = TRUE)
  boxcox(lmvar, optimize = optimize)
}

l=array(0)
l[1]=bc(A,Calories,optimize = T)$lambda
l[2]=bc(A,Protein,optimize = T)$lambda
l[3]=bc(A,Fat,optimize = T)$lambda
l[4]=bc(A,Sodium,optimize = T)$lambda
l[5]=bc(A,Fiber,optimize = T)$lambda
l[6]=bc(A,Carbohydrates,optimize = T)$lambda
l[7]=bc(A,Sugar,optimize = T)$lambda
l[8]=bc(A,Potassium,optimize = T)$lambda
l

trans=function(X,lambda)
{
  Y=as.data.frame(matrix(0,nrow=43,ncol=10))
  Y[,1]=X[,1]
  Y[,2]=X[,2]
  for(i in 1:8)
```

```
  {
    Y[,i+2]=(X[,i+2] ^ lambda[i] - 1) / lambda[i]
  }
  colnames(Y)=colnames(X)
  return(Y)
}
D=trans(A,1)
D

## Normality Checking------------------------------
# qqplot
plist_tq0 = list()
for(i in 1:8){
  plist_tq0[[i]]=qqmath(~ scale(D[,i+2]),data = D,
                        prepanel = prepanel.qqmathline,
                        panel = function(x, ...) {
                          panel.qqmathci(x, ...)
                          panel.qqmathline(x, ...)
                          panel.qqmath(x, ...)
                        },pch=19,xlab="Theortical
                           Quantile of N(0,1)",
                        ylab="Observed Quantiles",
                        main=paste(colnames(D)[i+2]))
}

do.call(grid.arrange, c(plist_tq0, ncol = 4,nrow=2,
```

```
159                          top="QQPlot of Transformed
                                Individual Variables (
                                Standardised) vs N(0,1)") )
160
161 # shapiro test
162 shap.wilk=function(X)
163 {
164   Result=matrix(0,nrow=8,ncol=3)
165   Result[,1]=colnames(X)[3:10]
166   for(i in 1:8)
167   {
168     Result[i,2]=shapiro.test(X[,i+2])$p
169     Result[i,3]=ifelse(as.numeric(Result[i,2])<0.01,"
          Reject","Accept")
170   }
171   return(Result)
172 }
173 shap.wilk(D)
174
175 D.new= D[,-c(1,2)]
176 D.new
177
178 ## PCA on the original data frame-------------------
179 d=B[,-c(1,2)]
180 p=prcomp(d,center=T,scale.=T)
181 p$rotation
```

```
182 p$x
183 biplot(p,scale=0,main="Biplot of Principle Component 1
       and 2")
184
185 # calculate total variance explained by each principal
       component
186 var_explained=p$sdev^2/sum(p$sdev^2)
187 var_explained
188 # checking how many pc are required to explain 95%
       variability
189 prop=((cumsum((p$sdev)^2))/(sum((p$sdev)^2)))*100
190 prop
191 #create scree plot
192 qplot(c(1:8),var_explained)+
193   geom_line()+
194   xlab("Principal Component")+
195   ylab("Variance Explained")+
196   ggtitle("Scree Plot")+
197   ylim(0,1)
198
199 ## Clubbing of two groups------------------------
200 data1=B[,-1]
201 data1$Manufacturer=as.factor(data1$Manufacturer)
202 data1
203 attach(data1)
204
```

```
# just an exploratory approach to see which groups should be clustered----------
# variables: Potassium, Fiber, Protein (from PCA)

# 3d plot----------
plot_ly(x=Potassium, y=Fiber, z=Protein, type="scatter3d",mode="markers",
        color=as.factor(Manufacturer),
        title="3D Scatterplot of Potassium,Fiber and Protein")%>%
  layout(title = 'Scatter plot of top 3 variables that explains the variation of the dataset most',
         scene = list(xaxis=list(title='Potassium'),
                      yaxis=list(title='Fiber'),
                      zaxis=list(title='Protein')),
         legend = list(title=list(text='Types of Manufacturer')))
# add group2 and group3.

# New data frame----------
data2=data1
data2$Manufacturer=as.factor(as.numeric(replace(data2$Manufacturer,which(data2$Manufacturer==3),2)))
data2

# Everything done now onwards considering 2 groups
```

```
225 # EDA of group-1-----------
226 #1. boxplot---------
227 eda.g1 = list()
228 for(i in 1:8)
229 {
230   eda.g1[[i]] = bwplot(as.matrix(data2[,-1])[1:17,i],
231                        xlab=paste(colnames(B)[i+2]))
232 }
233
234 do.call(grid.arrange, c(eda.g1, ncol = 4,nrow=2,
235                         top='Boxplot of Individual
                               Variable of Manufacturer=1')
                               )
236 #2. Summary-----------
237 summary(data2[,-1][1:17,])
238
239 #3. Correlation heatmap----
240 corr = data.matrix(cor(data2[,-1][1:17,]))
241 mel = melt(corr)
242 ggplot(mel, aes(X1,X2))+geom_tile(aes(fill=value)) +
243   geom_text(aes(label = round(value, 1)))+
244   scale_fill_gradient2(low='red',mid = 'white' ,high='
        maroon')+
245   labs(title="Correlation Heatmap of Manufacturer=1")
246
247 # EDA of group-2-----------
```

```
#1. boxplot---------
eda.g2 = list()
for(i in 1:8)
{
  eda.g2[[i]] = bwplot(as.matrix(data2[,-1])[18:43,i],
                       xlab=paste(colnames(B)[i+2]))
}

do.call(grid.arrange, c(eda.g2, ncol = 4,nrow=2,
                        top='Boxplot of Individual
                           Variable of Manufacturer=2')
                           )
#2. Summary-----------
summary(data2[,-1][18:43,])

#3. Correlation heatmap----
corr = data.matrix(cor(data2[,-1][18:43,]))
mel = melt(corr)
ggplot(mel, aes(X1,X2))+geom_tile(aes(fill=value)) +
  geom_text(aes(label = round(value, 1)))+
  scale_fill_gradient2(low='red',mid = 'white' ,high='
     maroon')+
  labs(title="Correlation Heatmap of Manufacturer=2")


## Univariate Normal---------------------
```

```
271 # Adding jitter-----------------------------
272 data3=data2
273 data3$Calories=data2$Calories+runif(43,0,7.8)
274 data3$Protein=data2$Protein+runif(43,0,.5)
275 data3$Fat=data2$Fat+runif(43,.1,.5)
276 data3$Fiber=data2$Fiber+runif(43,0,2)
277 data3$Sodium=data2$Sodium+runif(43,0.001,0.005) #
       because of 0
278 data3$Sugar=data2$Sugar+runif(43,0.001,0.005) #because
       of 0
279 data3
280
281 # QQplot of individual variables----------------
282
283 plist_qq1 = list()
284 for(i in 1:8){
285   plist_qq1[[i]]=qqmath(~ scale(data3[,i+1])|
        Manufacturer,data = data3,
286                         prepanel = prepanel.qqmathline,
287                         panel = function(x, ...) {
288                           panel.qqmathci(x, ...)
289                           panel.qqmathline(x, ...)
290                           panel.qqmath(x, ...)
291                         },pch=19,xlab="Theortical
                              Quantile of N(0,1)",ylab="
                              Observed Quantiles",
```

```
292                          main=paste(colnames(data3)[i+1])
                                ))
293 }
294
295 do.call(grid.arrange, c(plist_qq1, ncol = 4,nrow=2,
296                          top="QQPlot of Individual
                                Variables (Standardised) vs N
                                (0,1)") )
297
298
299 # MVN test----------------------
300 mvn(data3[1:17,-1],mvnTest = "royston",univariateTest="
       SW")
301 mvn(data3[18:43,-1],mvnTest = "royston",univariateTest =
       "SW")
302
303 # Boxcox Transformations--------------------
304 l=array(0)
305 l[1]=boxcox(lm(data3$Calories~data3$Manufacturer),
       optimize = T)$lambda
306 l[2]=boxcox(lm(data3$Protein~data3$Manufacturer),
       optimize = T)$lambda
307 l[3]=boxcox(lm(data3$Fat~data3$Manufacturer),optimize =
       T)$lambda
308 l[4]=boxcox(lm(data3$Fiber~data3$Manufacturer),optimize
       = T)$lambda
```

```
l[5]=boxcox(lm(data3$Potassium~data3$Manufacturer),
   optimize = T)$lambda
l

# Transformations----------------
data4=data3
data4[,2]=(data3[,2] ^ l[1] - 1) / l[1]
data4[,3]=(data3[,3] ^ l[2] - 1) / l[2]
data4[,4]=(data3[,4] ^ l[3] - 1) / l[3]
data4[,6]=(data3[,6] ^ l[4] - 1) / l[4]
data4[,9]=(data3[,9] ^ l[5] - 1) / l[5]
data4

# Normality Test after transformation--------------
mvn(data4[1:17,-1],mvnTest = "royston",univariateTest =
   "SW")
mvn(data4[18:43,-1],mvnTest = "royston",univariateTest =
    "SW")


# gamma plot
d_i.sq.1=mahalanobis(data4[1:17,-1],colMeans(data4
   [1:17,-1]),cov(data4[1:17,-1]))
di<-"chisq"
dp<-list(df=8)
```

```
330 gg.1<-ggplot(data=as.data.frame(d_i.sq.1),mapping=aes(
       sample=rchisq(17,8)))+
331   stat_qq_band(distribution=di,dparams=dp)+
332   stat_qq_line(distribution=di,dparams=dp)+
333   stat_qq_point(distribution=di,dparams=dp)+
334   labs(x="Sample Mahalanobis distance for group-1",y="
         Theoritical quantiles of Chi square(n=17,df=8)")
335 gg.1
336
337 d_i.sq.2=mahalanobis(data4[18:43,-1],colMeans(data4
       [18:43,-1]),cov(data4[18:43,-1]))
338 gg.2<-ggplot(data=as.data.frame(d_i.sq.2),mapping=aes(
       sample=rchisq(26,8)))+
339   stat_qq_band(distribution=di,dparams=dp)+
340   stat_qq_line(distribution=di,dparams=dp)+
341   stat_qq_point(distribution=di,dparams=dp)+
342   labs(x="Sample Mahalanobis distance for group-2",y="
         Theoritical quantiles of Chi square(n=26,df=8)")
343 gg.2
344
345 #QQplot after transformation---------------
346 plist_qq2 = list()
347 for(i in 1:8){
348   plist_qq2[[i]]=qqmath(~ scale(data4[,i+1])|
         Manufacturer,data = data4,
349                         prepanel = prepanel.qqmathline,
```

```
350                          panel = function(x, ...) {
351                            panel.qqmathci(x, ...)
352                            panel.qqmathline(x, ...)
353                            panel.qqmath(x, ...)
354                          },pch=19,xlab="Theortical
                                Quantile of N(0,1)",ylab="
                                Observed Quantiles",
355                          main=paste(colnames(data3)[i+1])
                                )
356 }
357
358 do.call(grid.arrange, c(plist_qq2, ncol = 4,nrow=2,
359                          top="QQPlot of Individual
                                Variables (Standardised) vs N
                                (0,1)") )
360
361
362 # Box M--------------------
363 boxM(data4[,-1],data4$Manufacturer)
364
365 # QDA---------------------
366 model=qda(Manufacturer~.,data=data4,CV=T)
367 model
368
369 (1-(length(which(data4$Manufacturer==model$class))/43))*
      100  # APER
```

```
370 data.frame(original=data4$Manufacturer,predicted=model$
      class) # Prediction
371 confusionmatrix(data4$Manufacturer,model$class)
372
373 S1=(cov(data4[,-1][1:17,]))        # Covarinace matrix
      of Manufacturer-1
374 S2=(cov(data4[,-1][18:43,]))       # Covarinace matrix
      of Manufacturer-2
375 x1.bar=apply(data4[,-1][1:17,],2,mean)    # mean vector
      of Manufacturer-1
376 x2.bar=apply(data4[,-1][18:43,],2,mean)   # mean vector
      of Manufacturer-2
377 a=(solve(S1)-solve(S2))*(-0.5)            # Quadratic
      coefficient
378 b=(t(x1.bar)%*%solve(S1))-(t(x2.bar)%*%solve(S2))  #
      Linear Coefficient
379 S1
380 S2
381 x1.bar
382 x2.bar
383 a
384 b
385
386 sep=function(X)        # Separation function
387 {
388   v=array(0)   # vector of qda values
```

```
389
390   for(i in 1:43)
391   {
392     v[i]=t(as.vector(X[i,]))%*%a%*%as.vector(X[i,])+b%*%
          as.vector(X[i,])
393   }
394   Y=matrix(0,nrow=43,ncol=2)
395   Y[,1]=data4$Manufacturer
396   Y[,2]=v
397   colnames(Y)=c("Manufacturer","Value")
398   return(Y)
399 }
400
401 s=sep(as.matrix(data4[,-1]))    # values
402 s
403 p=c(17/43,26/43)  # prior probabilities
404 k=(0.5*log(det(S1)/det(S2)))+(0.5*((t(x1.bar)%*%solve(S1
    )%*%x1.bar)-(t(x2.bar)%*%solve(S2)%*%x2.bar)))+log(p
    [2]/p[1])
405
406 k                  # separation constant
407
408 p=ggplot(NULL, aes(x=data4$Manufacturer, y=as.vector(s
    [,2])))+
409   geom_point()+ylab("Value")+xlab("Manufacturer")
410
```

```
411 p + coord_flip()+geom_hline(yintercept=k[,1],col="red")
      +
412   ggtitle("Separation Region")    # Separation Region
413
414 ifelse(s[,2]>k[1,],"1","2")    # Prediction on training
      set
415
416 ## Factor Analysis-------------------------
417 #Factor Analysis
418 f1=factanal(data4[,-1],factors=4)
419 f1$loadings^2
420 c=apply(f1$loadings^2,1,sum)#...communality
421 lamda<-f1$loadings
422 f1$uniquenesses #uniquenesses
423 psi<-diag(f1$uniqueness)
424 s=f1$correlation
425 sigma=lamda%*%t(lamda)+psi
426 res=s-sigma #...residual matrix
427 writexl::write_xlsx(data.frame(res),'D:/ISI/Sem-2/
      Swagata Nandi/res.xlsx')
428
429 f.none=factanal(data4[,-1],factors=4,rotation="none",
      scores="Bartlett")
430 lamda.none<-f.none$loadings
431 psi.none<-diag(f.none$uniqueness)
432 s.none=f.none$correlation
```

```
433 sigma.none=lamda.none%*%t(lamda.none)+psi.none
434 res.none=s.none-sigma.none #...residual matrix
435 c.none=apply(f.none$loadings^2,1,sum)#communality when
       there is no rotation done
436 writexl::write_xlsx(data.frame(c.none),'D:/ISI/Sem-2/
       Swagata Nandi/resvar.xlsx')
437
438 #..specific variances or uniqueness when there is no
       rotation
439 f.var=factanal(data4[,-1],factors=4,rotation="varimax",
       scores="Bartlett")
440 lamda.var<-f.var$loadings
441 psi.var<-diag(f.var$uniqueness)
442 s.var=f.var$correlation
443 sigma.var=lamda.var%*%t(lamda.var)+psi.var
444 res.var=s.var-sigma.var #...residual matrix
445 writexl::write_xlsx(data.frame(res.var),'D:/ISI/Sem-2/
       Swagata Nandi/resvar.xlsx')
446 c.var=apply(f.var$loadings^2,1,sum)#communality when
       there is an orthogonal rotation done
447 #uniqueness when there is an orthogonal rotation done
448 f.promax=factanal(data4[,-1],factors=4,rotation="promax"
       ,scores="Bartlett")
449 fun<-function(x,main,col)
450 {
451   for(i in 1:3)
```

```
452   {
453     for(j in (i+1):4)
454     {
455       plot(x[,i],x[,j],xlab=paste('Factor-',i),ylab=
              paste('Factor-',j),
456            xlim=c(-1,1),ylim=c(-1,1),main=main,col=col,
                 pch=19)
457       abline(h=0,v=0)
458       text(x[,i], x[,j], labels=colnames(B)[3:10], pos
             =2)
459     }
460   }
461 }
462 par(mfrow=c(1,1))
463 par(mfrow=c(2,3))
464 fun(f.none$loadings,main="No Rotation",col="blue")
465 fun(f.var$loadings,main="Varimax",col="blue")
466 fun(f.promax$loadings,main="Promax",col="blue")
467
468
469 ## Confidence Interval-------------------------
470 #Simultaneous confidence interval
471 par(mfrow=c(1,1))
472 x=data.frame(x1=data4[,6],x2=data4[,9])
473 mean=c(colMeans(x))
474 sigma.est=cov(x)
```

```
475 round(solve(sigma.est),3)
476 c=(((43-1)*8)/(43-8))*qf(0.01,df1=8,df2=35,lower.tail=F)
477 t=qt((0.01/4),df=42,lower.tail=F)
478 confidenceEllipse(mean,eig=eigen(sigma.est),n=43,p=2,
       alpha=0.01,xl=c(-0.5,2.5),yl=c(2,3))
479 #T^2 intervals
480 mu1.upper=mean[1]+sqrt(c)*sqrt(sigma.est[1,1]/43)
481 mu1.lower=mean[1]-sqrt(c)*sqrt(sigma.est[1,1]/43)
482 mu2.upper=mean[2]+sqrt(c)*sqrt(sigma.est[2,2]/43)
483 mu2.lower=mean[2]-sqrt(c)*sqrt(sigma.est[2,2]/43)
484 #Bonferroni's simultaneous confidence intervals
485 mu1.upper.b=mean[1]+t*sqrt(sigma.est[1,1]/43)
486 mu1.lower.b=mean[1]-t*sqrt(sigma.est[1,1]/43)
487 mu2.lower.b=mean[2]-t*sqrt(sigma.est[2,2]/43)
488 mu2.upper.b=mean[2]+t*sqrt(sigma.est[2,2]/43)
489 abline(h=mu2.upper)
490 abline(h=mu2.lower)
491 abline(v=mu1.lower,lty=3)
492 abline(v=mu1.upper,lty=3)
493 abline(h=mu2.upper.b,col="red")
494 abline(h=mu2.lower.b,col="red")
495 abline(v=mu1.lower.b,lty=3,col="red")
496 abline(v=mu1.upper.b,lty=3,col="red")
```