Exam

STK 802

Connor McDonald
u16040725

Behavioural Analytics

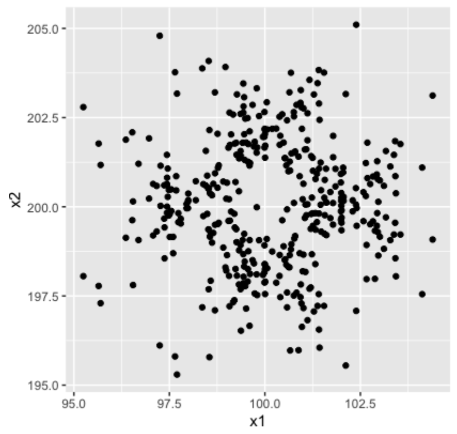
UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

Department of Computer Science
University of Pretoria
Date: 30 November 2021

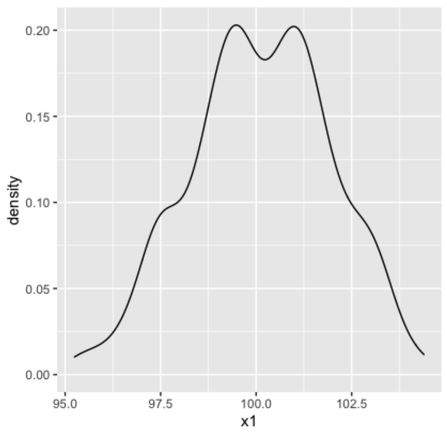
1

1.a Exploratory Analysis

When conducting the exploratory analysis, the first analysis conducted was a simple scatter plot paired with a density plot of the independent variable to visualise the relationship between the x1 and x2 variables. In figures 1a and 2a there appears to be two sets of parallel lines in the data, looking at the distributions of the x1 we see two distinct peaks near the center of the distribution and two smaller peaks at the sides of the distribution. X2 shows a distinct peak at the center and a smaller less prominent peak to the left side of the data.

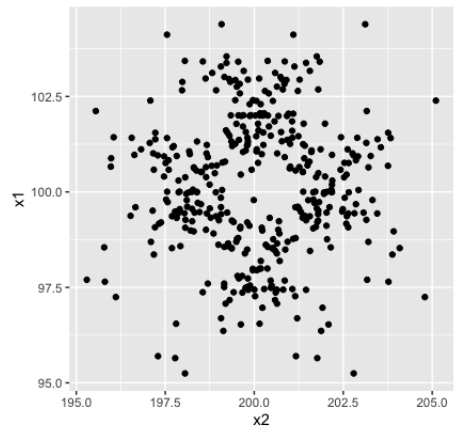


(a)

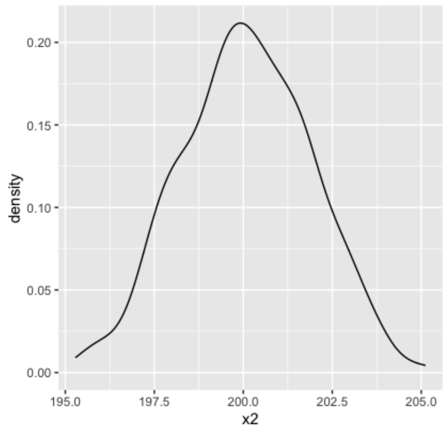


(b)

Figure 1



(a)



(b)

Figure 2

To get a better understanding of the data, we can plot both variables together in a contour plot, this is done in figure 3 below. This plot shows five regions which could be either local minimums or local maximums, a heat map of this contour plot was subsequently created (figure 4) to distinguish between local minimums and local maximums. Figure 4 thus implies that their are four peaks in the multivariate distribution of the data and thus the middle region in figure 3 can be ruled out as a potential mode.

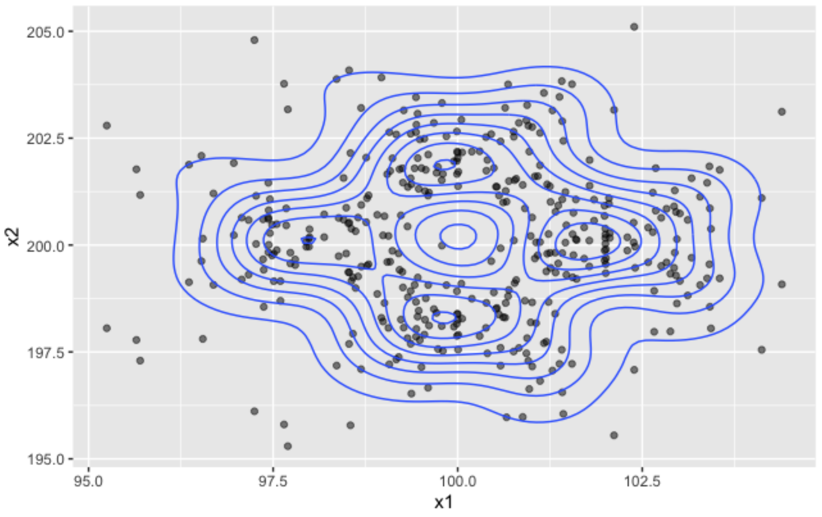


Figure 3: Contour Plot of X1 and X2

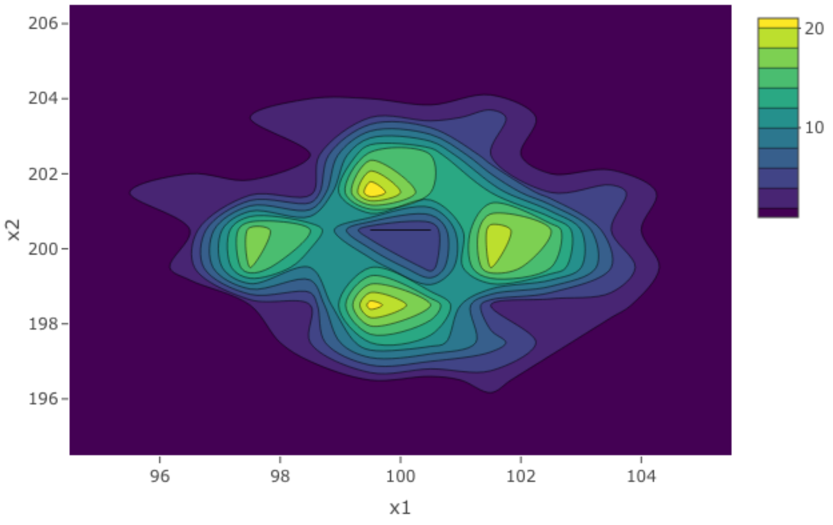


Figure 4: Heatmap of X1 and X2

Lastly, we can confirm the existence of four modes by plotting the multivariate Gaussian distribution of the data in figure 5.

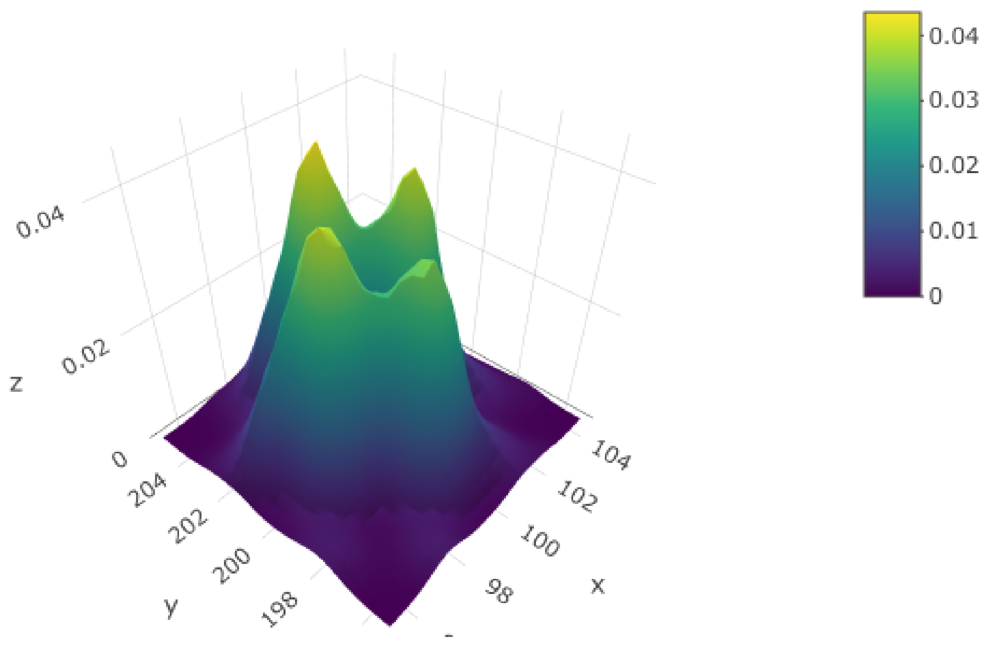
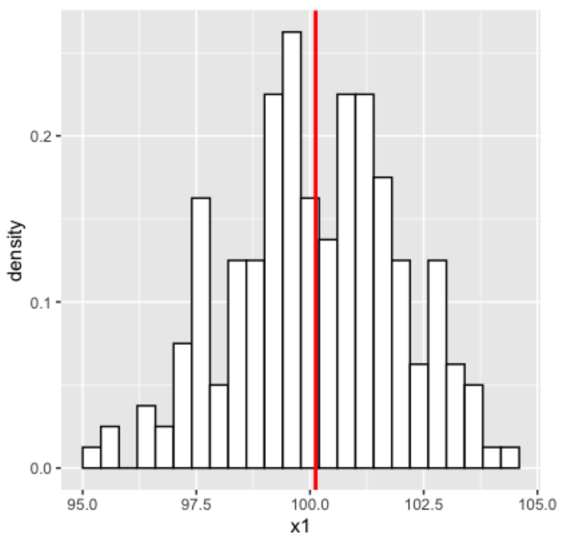


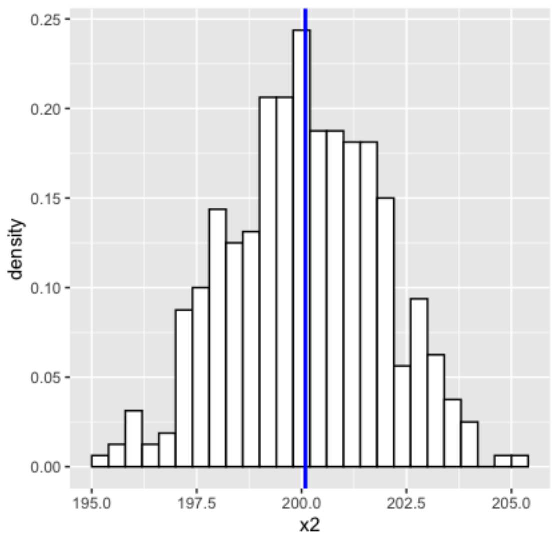
Figure 5: Multivariate Gaussian distribution

1.b Distribution Analysis

To analyse the distributions of each variable, figure 6 was generated. In terms of variance, both variables exhibited similar standard deviation around 1.8. However, the variables differed significantly in terms of their means, where x1 had a mean around 100 and x2 had a mean around 200. Both variables hint at 3 possible modes, with the modes in variable x1 being more clearly defined. When we look at the multivariate Gaussian distribution from the perspective of x1 and x2 respectively in figure 7, we can see how these three modes are achieved in each distribution, although this may not be in proportion as the middle modes in both figures are essentially summed to get the density shown in the histograms.

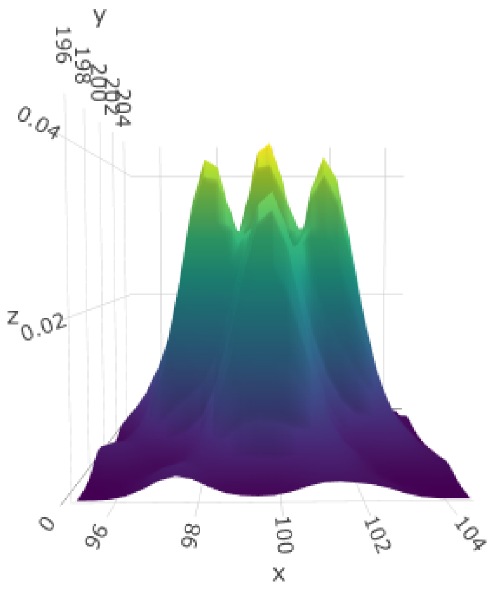


(a) $\mu = 100.12$, $\sigma = 1.82$

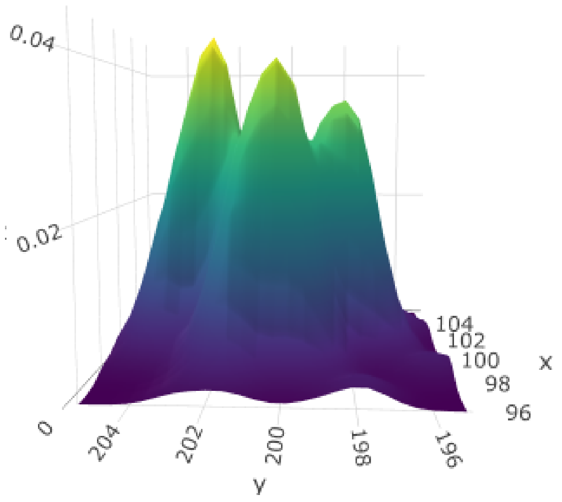


(b) $\mu = 200.1$, $\sigma = 1.81$

Figure 6: Density Distribution on Variables



(a) x1 perspective



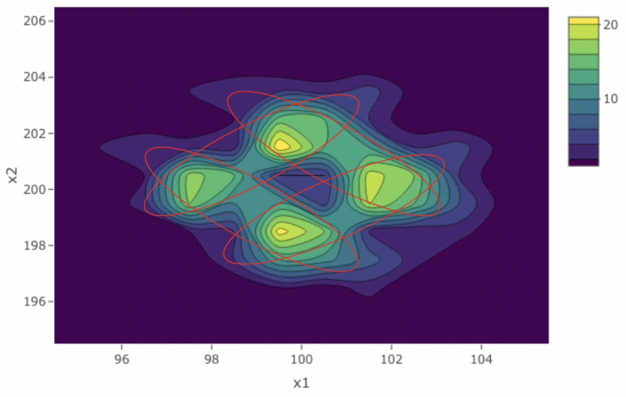
(b) x2 perspective

Figure 7: Multivariate Gaussian from different perspectives

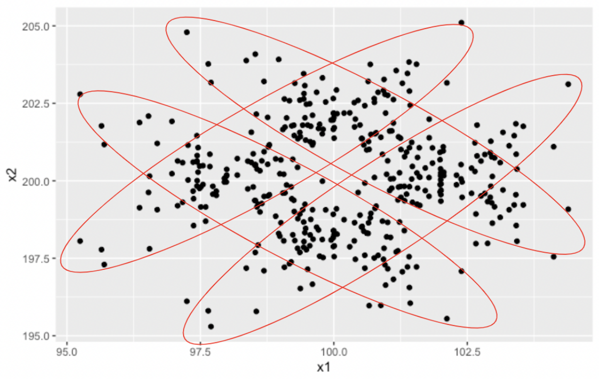
1.c Mixture Model

1.c.i Number of components

Looking at figure 8a one could argue that the data shows the four regions in which data can be clustered, the same circles can be plotted on figure 8b. Therefore, it is suggested that the mixture model requires four components.



(a) x1 perspective



(b) x2 perspective

Figure 8: Potential cluster regions

1.c.ii Model Parameters

Table 1: Model Paramters

Parameter/Component	Component 1	Component 2	Component 3	Component 4
x1 mean (μ_{x1})	99.91	101	101	99.1
x2 mean (μ_{x2})	199.0	199.0	201.0	201.2
variance (σ)	-2.01	2.14	-1.98	2.21
covariance	$\begin{bmatrix} 2.32 & -2.01 \\ -2.01 & 2.19 \end{bmatrix}$	$\begin{bmatrix} 2.34 & 2.14 \\ 2.14 & 2.29 \end{bmatrix}$	$\begin{bmatrix} 2.25 & -1.98 \\ -1.98 & 2.21 \end{bmatrix}$	$\begin{bmatrix} 2.36 & 2.21 \\ 2.21 & 2.35 \end{bmatrix}$
Mixing Probability (λ)	0.268	0.234	0.275	0.224

1.c.iii Graphical Representation

Figure 9 therefore shows that the data can be grouped into four distinct clusters with similar locations to those shown in figure 8.

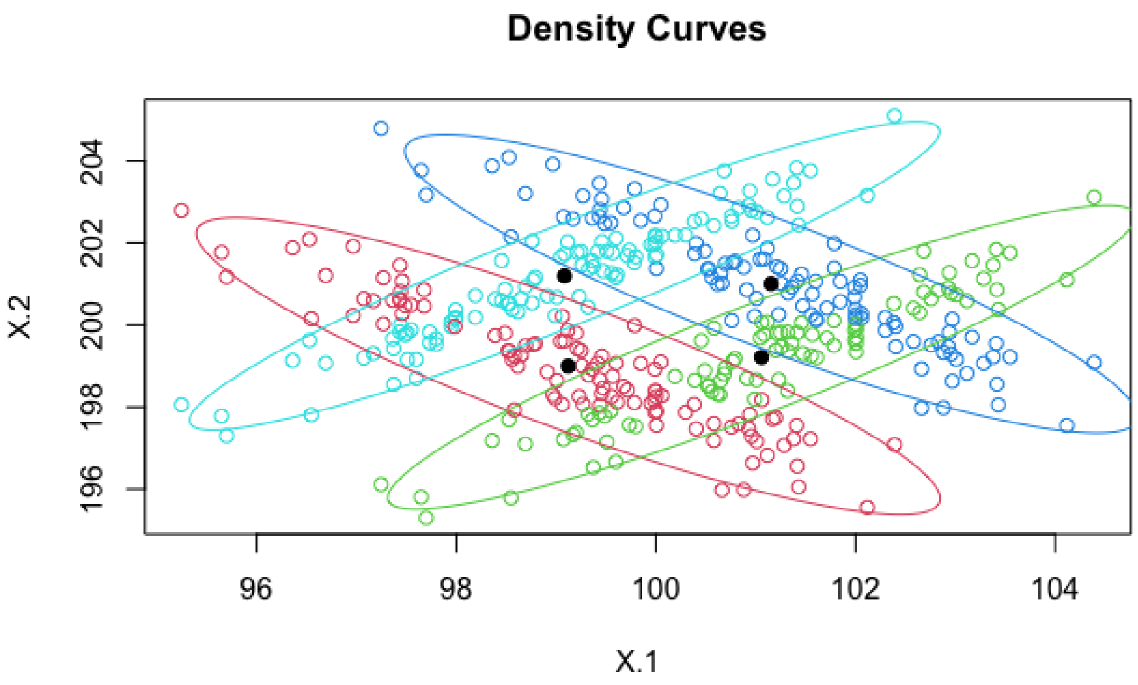


Figure 9: Clustering solution

1.c.iv Clustering Ability of Mixture Model

When looking at the clustering ability of the model it was observed that the model showed high probabilities regarding points that fell far from the overlapping regions. However, the model would often show two relatively high probabilities when attempting to cluster points in overlapping regions. This was expected as only two clusters can overlap at one time.

For example, when looking at figure 10, it is observed that point a falls far from an overlapping region, the posterior probability that this point belonged to component 2 was 100%, indicating that the model was extremely confident in this prediction. However, when looking at point b, which falls in an overlapping region, the posterior probability for this point was 0.48 for component 2 and 0.52 for component 3.

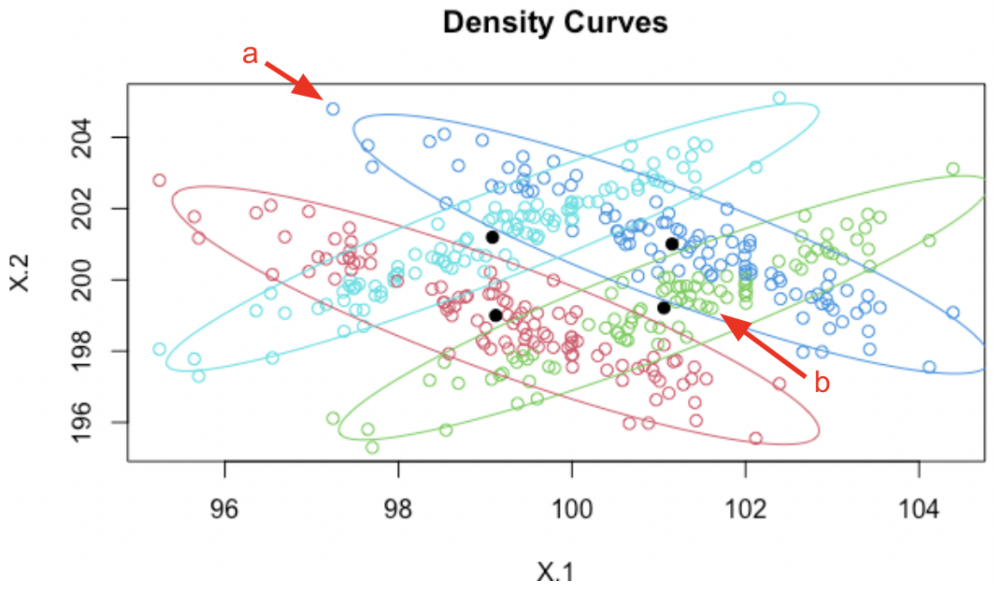
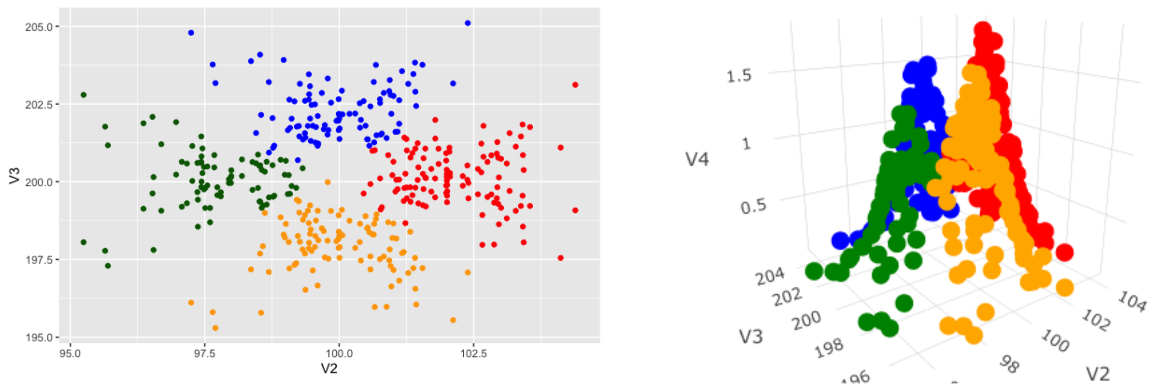


Figure 10: Clustering Ability

1.d k-NN Mode Seeking

Much like the Gaussian mixture regression model, the k-NN mode seeking algorithm was also able to identify four modes (when using a k-Value of 40). However, the clustering solution differed, as seen in figure 11a. The k-NN algorithm identified more circular clusters when compared to the clusters identified by mixture regression. This is expected as the k-NN mode seeking algorithm searches for a local density maximum when clustering.



(a) Cluster Solution

(b) k-NN density estimation

Figure 11: Potential cluster regions

2

2.a Exploratory Analysis

The first step of the exploratory analysis was to plot the data to see if any relationships could be observed between the response variable and the explanatory variable. This produced figure 12, which showed no linear relationships. However, the data appeared "circular" which can be modelled by hierarchical clustering. Using a euclidean distance and average method in a hierarchical clustering algorithm the dendrogram in figure 13 was produced. This subsequently indicated that the optimal number of clusters in the data was three.

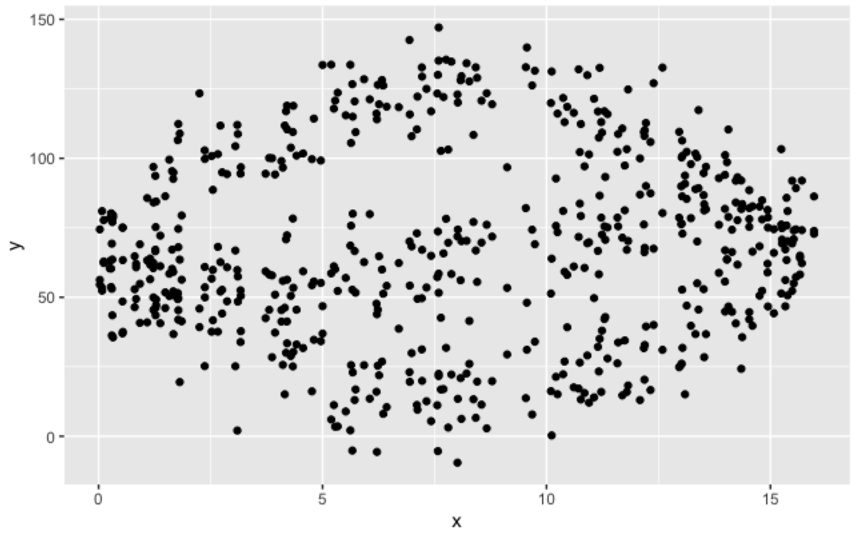


Figure 12: Relationship Between X and Y

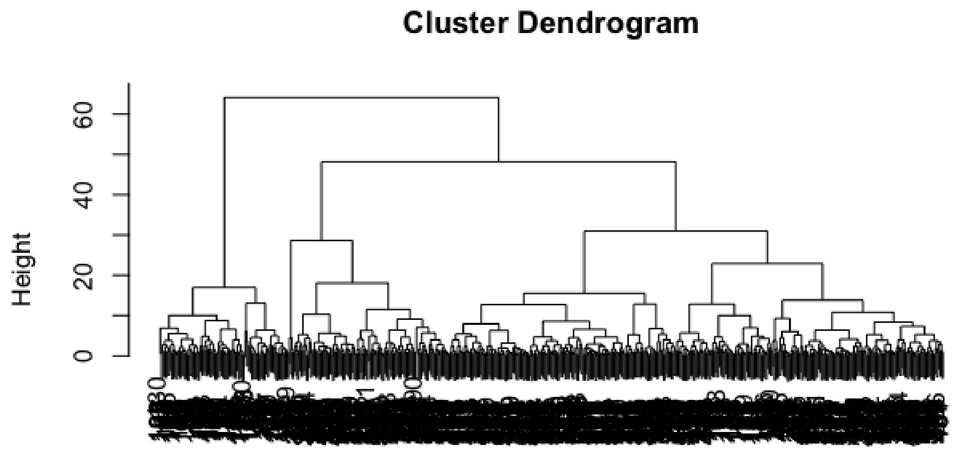


Figure 13: Dendrogram of the data

Using this information, the data was clustered in figure 14 and clearly shows the three clusters observed in the data. Lastly, a Gaussian mixture model of the response variable can also be constructed with 3 components.

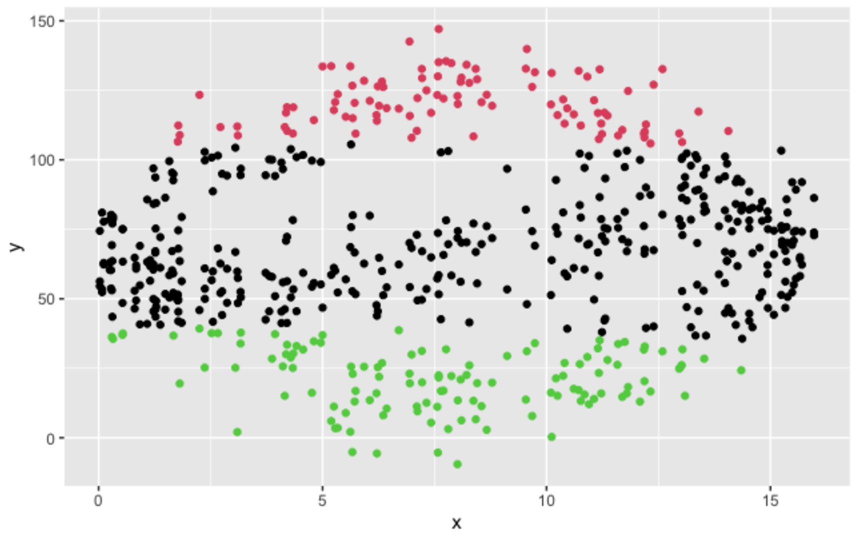


Figure 14: Dendrogram of the data

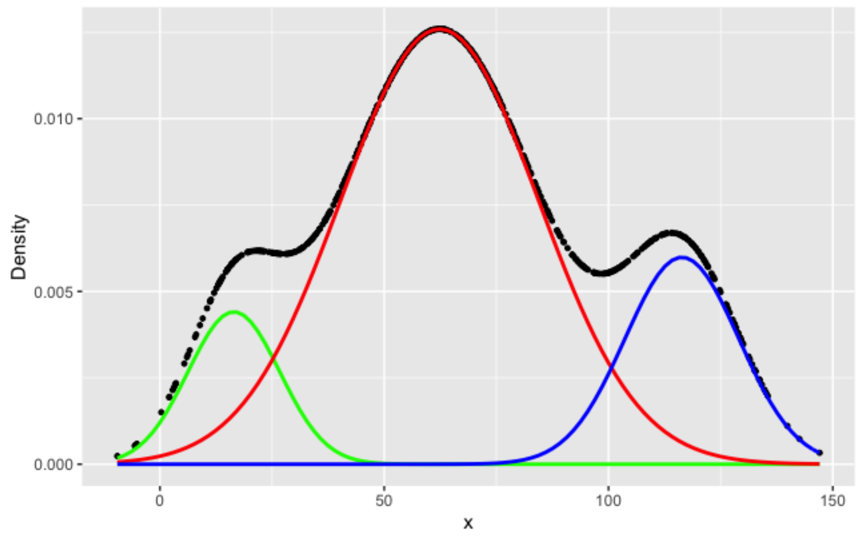


Figure 15: Gaussian Mixture Model of the Response Variable

2.b Polynomial Gaussian Regression Model

2.b.i Model Parameters

Using the flexmix package in R, a mixture regression model was created with the parameters shown in table 2. The functional form used by the model is represented in equation 1. This reason for this degree of polynomial was so that the data could be generalised well enough without being overfit.

$$y = x_0 + b_1 x_1 + b_{2x} 1^2 + b_3 x_2^3 \tag{1}$$

Table 2: Model Paramters

Parameter/Component	Component 1	Component 2	Component 3
coef.(Intercept)	34.077	103.848	65.459
coef.poly(x1, 3)1	-23.28	-21.296	248.718
coef.poly(x1, 3)2	387.868	-434.171	20.306
coef.poly(x1, 3)3	-0.906	-32.146	-33.603
sigma	11.634	10.419	10.294

2.b.ii Chosen Value of k

Based on the the exploratory data analysis, the initial k value was chosen as 3, this value proved to be effective in modelling the data. Furthermore, the model appeared to cluster the data in a very similar manner to the hierarchical clustering algorithm (see figure 16) .

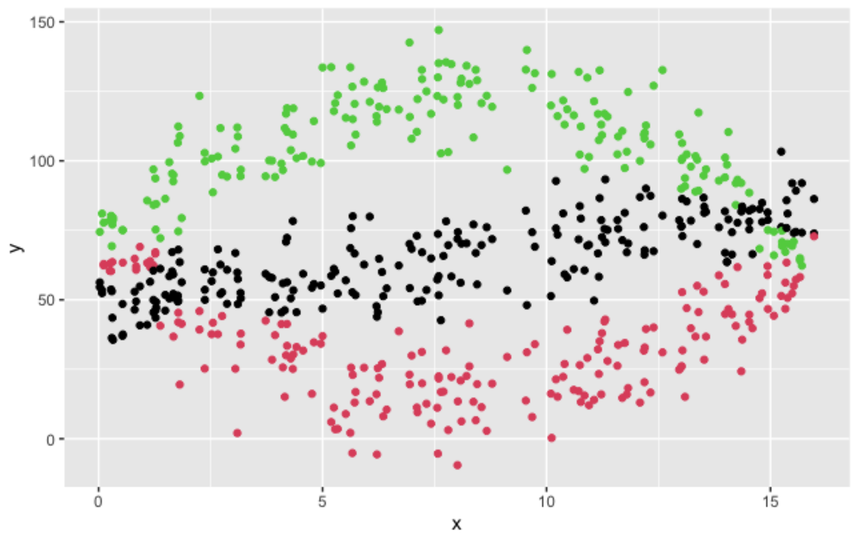


Figure 16: Mixture Model Cluster Solution

2.b.iii Visualisation of Mixture Regression Model

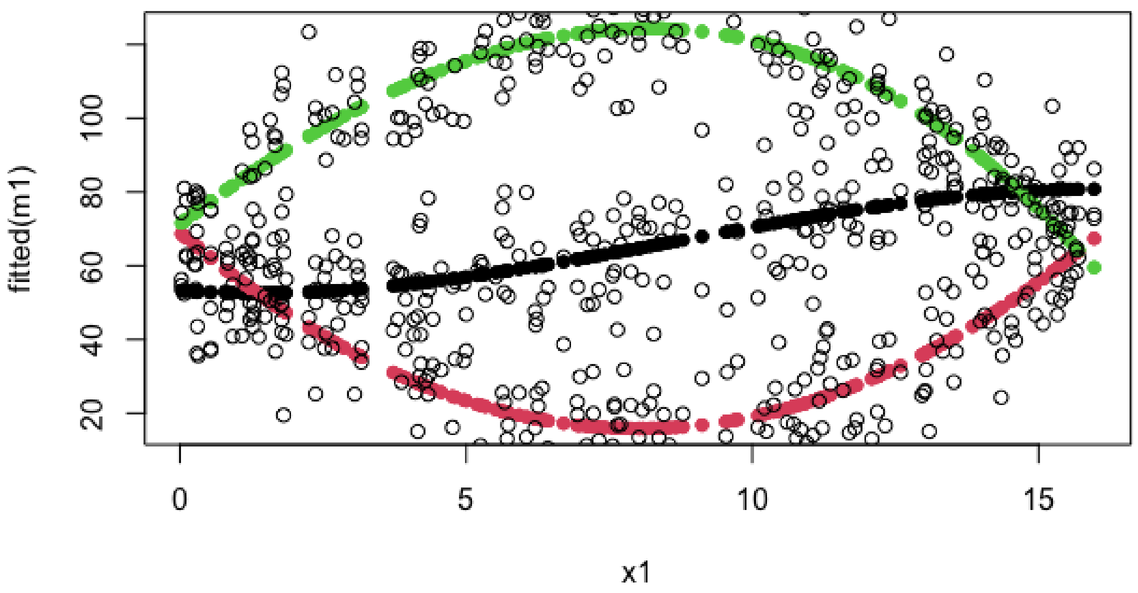


Figure 17: Mixture of Polynomial Gaussians Regression Model

2.b.iv Model Interpretation

Looking at figure 17 it was observed that 3 components fitted the data optimally, as each component showed relatively low variance, indicating a close fit to the regression model. The 3-component version of the model also had the lowest AIC and BIC of all models with components ranging from 2-7.

Table 3: Model Summary

	prior	size	post 0	ratio
comp 1	0.323	192	368	0.522
comp 2	0.343	220	451	0.488
comp 3	0.334	188	412	0.456

log Likihood -2755.104 (df=17)
AIC: 5544.208
BIC: 5618.956

3 Topic Modelling

3.a

Topic modelling is an unsupervised machine learning method used to find natural groupings within textual data, such as finding the theme or topic in a corpus of documents and then classifying these documents to the discovered themes. The main objectives of topic modelling are as follows:

- Identifying themes or topics within a corpus of textual data
- Classifying documents to the identified themes/topics
- Using classifications to organise and search the documents.

3.b

In statistics, a mixture model is a probabilistic model for representing the presence of "sub-populations" within an overall population. This can be applied to topic modeling by representing the presence of topic or themes within a corpus of data. Mixture models work by determining a posterior probability of a specific observation belonging to a specific component or class. This posterior probability can be used to determine the probability of a specific word belonging to a specific topic. We can then determine the proportion of words in a document **D** that belong to a topic **T**, to determine the theme or topic of the document.

3.c

Latent Dirichlet Allocation (LDA) is a widely used topic modelling method where each document contains a set of words and each topic contains a set of words. The objective of LDA is to find a set of topics for which a document may belong to, based on the set of words in the document.

This algorithm works by randomly assigning each word in a document to one of K topics. Initially this is not a very accurate way to classify the topic of a document. A generative model is then used to go through each document **D** and each word **W** in **D**. In doing this, the posterior probability of words in **D** given that they are assigned to topic **T**. **W** is then reassigned to a new topic **T** where **T** is chosen by $P(\mathbf{T} \mid \mathbf{D}) \times P(\mathbf{W} \mid \mathbf{T})$. After repeating this process a large number of times, the algorithm will reach a steady state, where topic identification is reasonably accurate. These topics can then be used in a classification process by estimating topic mixtures within a document **D** and counting the proportion of words assigned to each topic for that document as well as the proportion of words assigned to each topic overall.

3.d

Preprocessing is essential to the success of an LDA model as textual data often contains a lot of redundant nuances that do not help in identifying the topic of a document. For example, as humans we understand that capitial letters do not change the meaning of a word, however a computer may interpret the same word as different words due to one version having a capital letter. Preprocessing of textual data typically consists of the following actions:

- Changing all letters to lowercase
- Removing all special characters such as quotes, commas and apostrophes
- Removing any remaining punctuation, numbers, or white space surrounding the words
- Accounting for regional differences in spelling, for example “colour” vs. “color”

3.e

Clustering of words is done by calculating the probability that a specific word belongs to a specific topic by calculating the proportion of given word in a given topic in relation to all topics. Words can then be sorted by descending probability, and the top n words can be used to represent a specific topic. Another method is to impose a threshold to filter out low probability words. Thus each topic essentially represents a cluster of words.

Table 4: Words

topic/word	word 1	word 2	word 3	word 4
topic 1	0.01	0.23	0.19	0.03
topic 2	0.21	0.07	0.48	0.02
topic 3	0.53	0.01	0.17	0.04

3.f

After words have been clustered, topics can be associated to specific clusters of words. Since the number of topics will inherently be less than the number of words, this process can be see as a form of dimensionality reduction. The topic space can subsequently be used to perform clustering of documents based on the words with in the document and how common they are to any given topic. This is not hard clustering where one topic is given as the final answer, but rather soft clustering, which provides probabilities that the document belongs to each topic, where the highest probability is then seen as the final classification. However, this leaves room for in very similar topics which may have similar probability scores.

3.g

Topic modelling algorithms are typically split into two parts, topic clustering and topic classification

Topic Clustering: This is the first phase of any topic modelling algorithm, this consists of unsupervised machine learning techniques which analyse word frequency, distance to other words, and the the k-nearest neighboring words. This can then be used to form word clusters which topics may be associated to.

Topic Classification: Unlike topic clustering, topic classification is a supervised machine learning technique which makes use of predetermined topics or classes to classify documents into topics.

Topic modelling in general has a wide variety of uses from sentiment analysis, search engine optimisation and lastly, understanding huge amounts of text based data in a very short amount of time.

A Question 1 Code

```
library(ggplot2)
library(GGally)
library(mixtools)
library(tidyverse)
library(mvtnorm)
library(plotly)
library(MASS)

# 1a
d = read.csv('q1.csv')

d %>%
  ggplot(aes(x = x1)) +
  geom_density() + theme(aspect.ratio=1)

d %>%
  ggplot(aes(x = x2)) +
  geom_density()+ theme(aspect.ratio=1)

ggplot(d, aes(x=x1, y=x2)) + geom_point()

ggplot(d, aes(x=x2, y=x1)) + geom_point()+ theme(aspect.ratio=1)

###########################################################################

p2 <- ggplot(d, aes(x = x1, y = x2)) +
  geom_point(alpha = .5) +
  geom_density_2d()

p2


(p <- plot_ly(d, x = ~x1, y = ~x2))
add_histogram2dcontour(p)



dens <- kde2d(d$x1, d$x2)

plot_ly(x = dens$x,
        y = dens$y,
        z = dens$z) %>% add_surface()



# 1b
d = read.csv('q1.csv')

x1<-as.matrix(d[,1])
x2<-as.matrix(d[,2])
data = data.frame(x1, x2)

m1<- mean(d$x1)
m2<- mean(d$x2)

s1<- sd(d$x1)
s2<- sd(d$x2)

p <- ggplot(data, aes(x = x1)) +
  geom_histogram(aes(x1, ..density..), binwidth = 0.4,
        colour = "black", fill = "white") +
  geom_vline(xintercept = m1, col = "red", size = 1)
  + theme(aspect.ratio=1)
p

p <- ggplot(data, aes(x = x2)) +
  geom_histogram(aes(x2, ..density..), binwidth = 0.4,
        colour = "black", fill = "white") +
  geom_vline(xintercept = m2, col = "blue", size = 1)
  + theme(aspect.ratio=1)
p

##########################################################

d = read.csv('q1.csv')
x <- c(d$x1,d$x2)
class <- c(rep('x1', 400), rep('xw', 400))
data <- data.frame(cbind(x=as.numeric(x), class=as.factor(class)))


p <- ggplot(data, aes(x = x)) +
  geom_histogram(aes(x, ..density..), binwidth = 1,
  colour = "black", fill = "white") +
  geom_vline(xintercept = m1, col = "red", size = 2) +
  geom_vline(xintercept = m2, col = "blue", size = 2)
p

library(mixtools)

d = read.csv('q1.csv')
set.seed(100)
clusters = mvnormalmixEM(d, k = 4,epsilon=1e-04)
plot(clusters,which=2, alpha = 0.05)

posterior_prob = clusters[["posterior"]]

#1d
xy<-read.csv("q1.csv",sep=",")
x<-as.matrix(xy)

n<-nrow(x)
n
k= 40


dx<-as.matrix(dist(xy,p=2))

knnde <- function(k,n) {
  fx<-0
  for(i in 1:n) {
    zz <- dx[,i]
    dv<-sort(t(zz))
    fx[i]<-1/(dv[k]^2)
  }
  #xfx
  xfx <- cbind(1:n,xy,fx)
  nm <- cbind("onum","x1","x2","fx")
  colnames(xfx) <- nm
  return(xfx)
}

xfx<-knnde(k,n)
xfxplot<-as.data.frame(xfx)
```

```
ms <- function(onum) {
  flag<-0
  nn<-onum
  kk<-0
  obs<-1
  while (flag==0) {
    kk<-kk+1
    zz <- cbind(dx[,nn],xfx)
    colnames(zz)<- c("d","nr","x1","x2","fx")
    #head(zz)
    dv<-zz[order(zz[,1]),]
    dvc <-  t(dv[1:k,5])
    #head(dvc)
    ss<-which.max(dvc)
    #ss
    nn<-dv[ss,2]
    #nn
    if(nn==obs){flag<-1}
    if(kk==n){flag<-1}
    obs<-nn
    #obs
  }
  pointval <- dv[ss,2]
  pval <- cbind(xfxplot[onum,],pointval)
  return(pval)
}

#eg. for observation 1
mode1 <- ms(1)
mode1

z<-matrix(,nrow = n,ncol = 5)
for(i in 1:n) {
  mode1 <- ms(i)
  z[i,1]<- mode1[1,1]
  z[i,2]<- mode1[1,2]
  z[i,3]<- mode1[1,3]
  z[i,4]<- mode1[1,4]
  z[i,5]<- mode1[1,5]
}
```

```
head(z)
pmodes<-unique(z[,5])
sort(pmodes)


xfx<-knnde(k,n)
xfxplot<-as.data.frame(z)

c = c()
for(i in 1:length(xfxplot$V1)){
  if(xfxplot$V5[i] == 122){
    c[i] = 'red'
  }else if(xfxplot$V5[i] == 283){
    c[i] = 'blue'
  }else if(xfxplot$V5[i] == 317){
    c[i] = 'darkgreen'
  }else{
    c[i] = 'orange'
  }
}


xfxplot$c = c

p4 <- plot_ly(xfxplot, x = ~ V2, y = ~ V3, z = ~ V4,
               marker = list(color = ~ c)) %>%
  add_markers()

p4

ggplot(xfxplot, aes(x=V2, y=V3)) + geom_point(col = xfxplot$c)
```

B Question 2 Code

```
#2a
d = read.csv('q2.csv')

ggplot(d, aes(x=x, y=y)) + geom_point()

clusters <- hclust(dist(d, method = 'euclidean'), method = 'average')
plot(clusters)

clusterCut <- cutree(clusters, 3)
d$class <-clusterCut

ggplot(d, aes(x=x, y=y)) + geom_point(col = d$class)

c = c()
for(i in 1:length(d$class)){
  if(d$class[i] == 1){
    c[i] = 'red'
  }else if(d$class[i] == 2){
    c[i] = 'blue'
  }else{
    c[i] = 'green'
  }
}

d$c = c



xy<-read.csv('q2.csv')
xy$c = c
x<-as.matrix(xy[,2])
y<-as.matrix(xy[,1])
c<-as.matrix(xy[,3])




# DOING THE GUASSIAN MIXTURE MODELLING
plot_mix_comps <- function(x, mu, sigma, lam) {
  lam * dnorm(x, mu, sigma)
}

set.seed(2021)

mixmdl <- normalmixEM(x, k = 3)

data.frame(x = mixmdl$x)
combined = mixmdl$lambda[1]*dnorm(x, mixmdl$mu[1],
mixmdl$sigma[1]) + mixmdl$lambda[2]*dnorm(x, mixmdl$mu[2],
mixmdl$sigma[2])+
  mixmdl$lambda[3]*dnorm(x, mixmdl$mu[3],
  mixmdl$sigma[3])
combined_df = data.frame(x, combined)


gaus_plot2 = ggplot() +
  geom_point(data=combined_df, aes(x = x, y = combined),
  color = "black", size = 1) +
  stat_function(geom = "line", fun = plot_mix_comps,
  args = list(mixmdl$mu[1], mixmdl$sigma[1], lam = mixmdl$lambda[1]),
  colour = "green", lwd = 1) +
  stat_function(geom = "line", fun = plot_mix_comps,
  args = list(mixmdl$mu[2], mixmdl$sigma[2], lam = mixmdl$lambda[2]),
  colour = "red", lwd = 1) +
  stat_function(geom = "line", fun = plot_mix_comps,
  args = list(mixmdl$mu[3], mixmdl$sigma[3], lam = mixmdl$lambda[3]),
  colour = "blue", lwd = 1) +
  ylab("Density")

gaus_plot2


#2b
library(ggplot2)
library(flexmix)

d = read.csv('q2.csv')

x1 = d$x
y = d$y
```

```
m1 <- flexmix(y ~ poly(x1,3), data = d, k = 3)
m1

c = clusters(m1)

par =parameters(m1)
par

matplot(x1, fitted(m1), pch = 16, type = "p")
points(x1, y)


ggplot(d, aes(x=x,  y=y)) +
  geom_point(color= c)


summary(m1)
```