Roll_no_08

Dinesh oli

2024-05-31

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

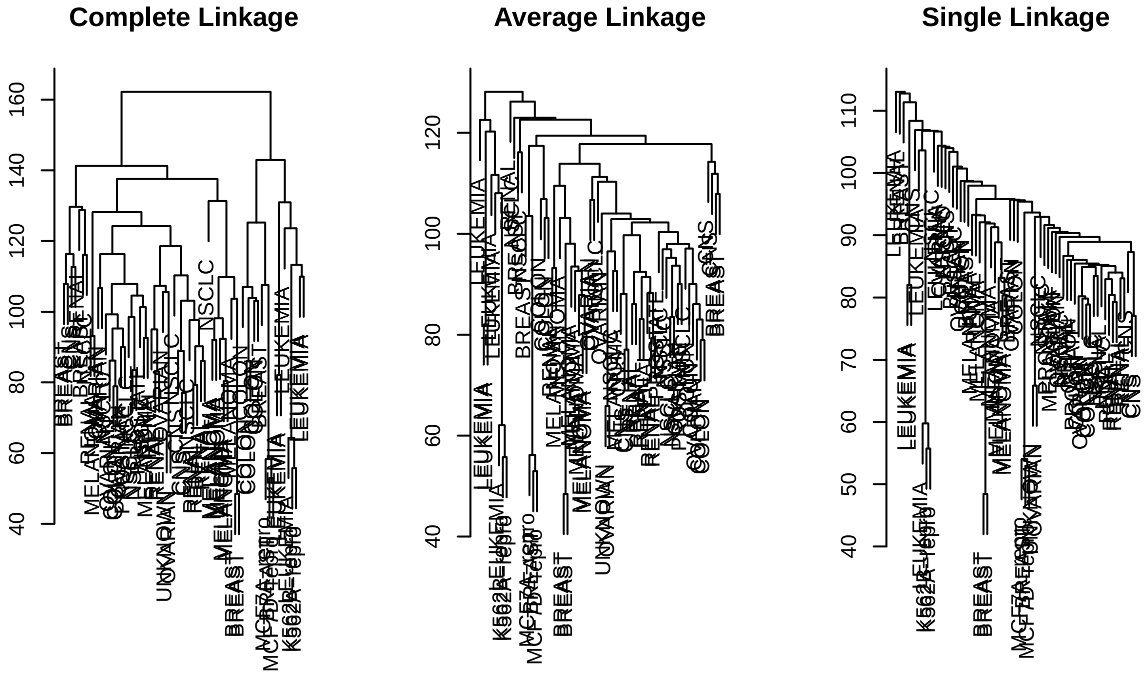
When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
summary(cars)
```

```
##      speed           dist       
##  Min.   : 4.0   Min.   :  2.00  
##  1st Qu.:12.0   1st Qu.: 26.00  
##  Median :15.0   Median : 36.00  
##  Mean   :15.4   Mean   : 42.98  
##  3rd Qu.:19.0   3rd Qu.: 56.00  
##  Max.   :25.0   Max.   :120.00
```

Ans 10

```
#Ans 10
# Question
# Scaling the nci.data as sd.data object
library(ISLR2)

nci.labs <- NCI60$labs
nci.data <- NCI60$data
sd.data <- scale(nci.data)

# Question
# Here we are fitting hierarchical clustering on the given dataset using
# various methods and showing the corresponding dendogram
par(mfrow = c(1,3))
data.dist <- dist(sd.data)
plot(hclust(data.dist), xlab = "", sub = "", ylab = "",
     labels = nci.labs, main = "Complete Linkage")
plot(hclust(data.dist, method = "average"), xlab = "", sub = "", ylab = "",
     labels = nci.labs, main = "Average Linkage")
plot(hclust(data.dist,method = "single"), xlab = "", sub = "", ylab = "",
     labels = nci.labs, main = "Single Linkage")
```



```
hc.out <- hclust(dist(sd.data))
```

Ans 7

```
 #Ans 7
# using airquality data set
#Loading the ggplot2 library for graphing purposes
library(ggplot2)
#Loading the in-built airquality dataset
data(airquality)
aq <- airquality
#Calculating the number of intervals using Sturges' formula (a statistical tool/ formula)
num_intervals <- ceiling(log2(length(aq$Temp)) + 1)
#Creation of intervals using built-in cut() function
aq_intervals <- cut(aq$Temp, breaks = num_intervals)
#Calculation of frequency in each class using the table() function
aq_interval_freq <- table(aq_intervals)
aq_interval_freq
```

```
## aq_intervals
##    (56,60.6] (60.6,65.1] (65.1,69.7] (69.7,74.2] (74.2,78.8] (78.8,83.3]
##            8          10          14          16          26          35
## (83.3,87.9] (87.9,92.4]   (92.4,97]
##          22          15           7
```

```
#Calculation of cumulative frequencies
(cumulative_freq <- cumsum(aq_interval_freq))
```

```
##   (56,60.6] (60.6,65.1] (65.1,69.7] (69.7,74.2] (74.2,78.8] (78.8,83.3]
##           8          18          32          48          74         109
## (83.3,87.9] (87.9,92.4]   (92.4,97]
##         131         146         153
```

```
(cumulative_freq2 <- cumsum(rev(aq_interval_freq)))
```

```
##   (92.4,97] (87.9,92.4] (83.3,87.9] (78.8,83.3] (74.2,78.8] (69.7,74.2]
##           7          22          44          79         105         121
## (65.1,69.7] (60.6,65.1]   (56,60.6]
##         135         145         153
```

```
#Getting upper class boundaries using RegEx
(upper_boundaries <- as.numeric(sub("\\((.*),.*\\]", "\\1", names(cumulative_freq))))
```

```
## [1] 56.0 60.6 65.1 69.7 74.2 78.8 83.3 87.9 92.4
```

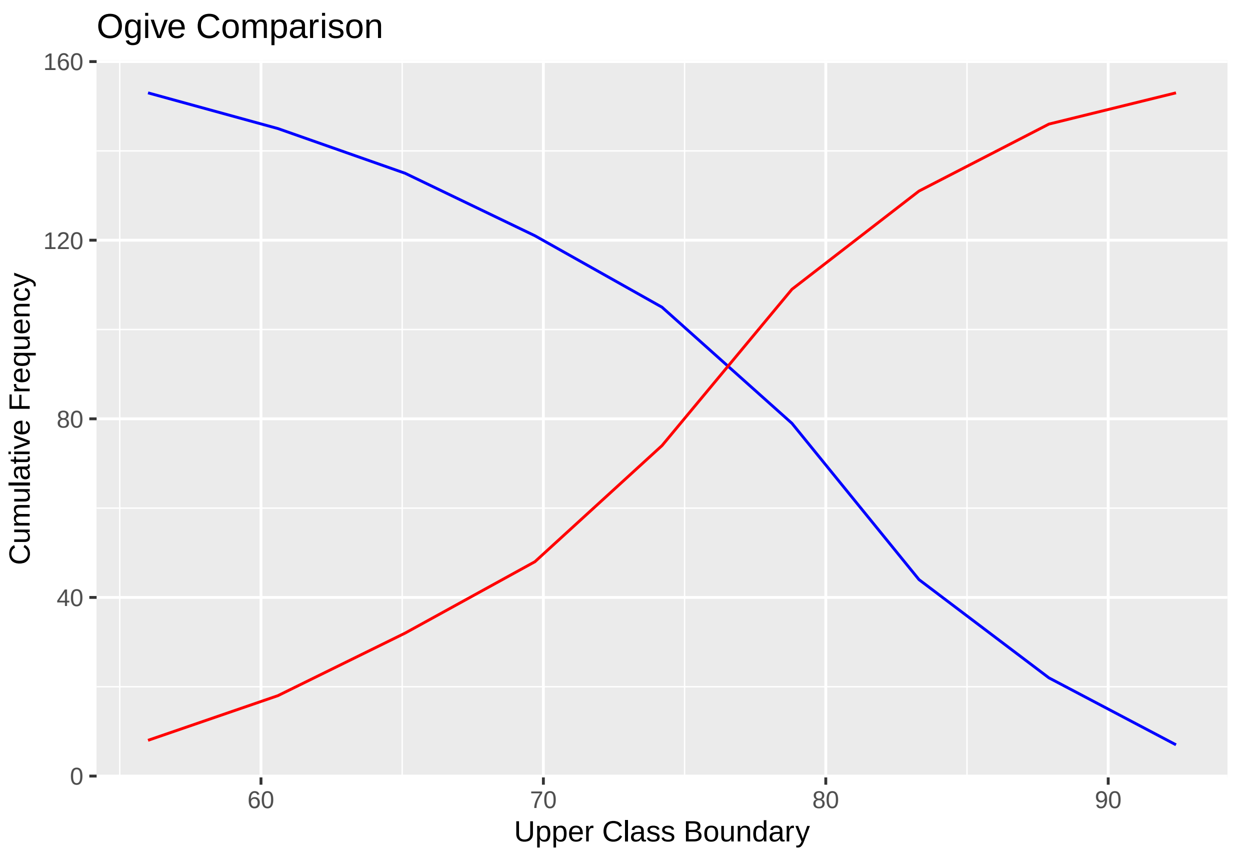
```
#Evaluating the maximum and minimum upper boundary
(max_upper_boundary <- max(upper_boundaries))
```

```
## [1] 92.4
```

```
(min_upper_boundary<-min(upper_boundaries))
```

```
## [1] 56
```

```
#More than frequency data for more than ogive
rev_upper_boundaries<-rev(upper_boundaries)
rev_cumulative_freq<-rev(cumulative_freq2)

#Creation of data frame, plotting of ogives, and determining the intersection point
df_more_than <- data.frame(Upper_Class_Boundary = upper_boundaries, Cumulative_Frequency = rev_cumulativ
df_less_than <- data.frame(Upper_Class_Boundary = upper_boundaries, Cumulative_Frequency = cumulative_fr

common_xlim <- c(min(min_upper_boundary, rev_upper_boundaries), max(max_upper_boundary, upper_boundaries

ggplot() +
  geom_line(data = df_more_than, aes(x = Upper_Class_Boundary, y = Cumulative_Frequency), color = "blue
  geom_line(data = df_less_than, aes(x = Upper_Class_Boundary, y = Cumulative_Frequency), color = "red")
  labs(x = "Upper Class Boundary", y = "Cumulative Frequency",
       title = "Ogive Comparison") +
  xlim(common_xlim)
```



```r
#Calculation of median using the median() function of R
median(aq$Temp)
```

```
## [1] 79
```

```r
# WORK 2
# Plotting histogram of 'Temp' variable
hist(aq$Temp, xlab = "Temperature", ylab = "Frequency", main = "Histogram of Temperature")

hist_data <- hist(aq$Temp, plot = FALSE)
#Calculation of the bin index that has the highest frequency
(max_freq_index <- which.max(hist_data$counts))
```

```
## [1] 6
```

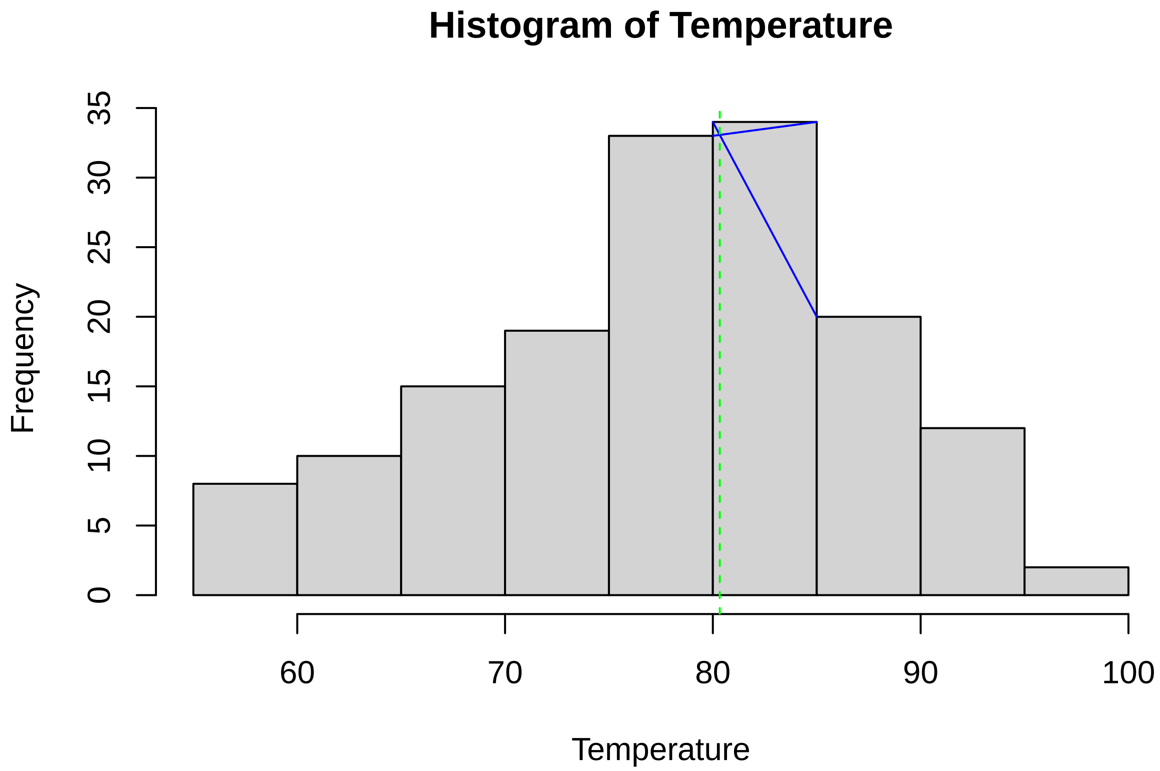
```r
#Determining the left and right bin adjacent to the bin with highest frequency
(left_bin <- max_freq_index - 1)
```

```
## [1] 5
```

```r
(right_bin <- max_freq_index + 1)
```

```
## [1] 7
```

```r
#Coordinates' calculation for the diagonal line from the edge of the largest bar to the tip of the oppo
x_left1 <- hist_data$breaks[left_bin + 1]
x_right1 <- hist_data$breaks[max_freq_index + 1]
y_max <- max(hist_data$counts)
y_left1 <- hist_data$counts[left_bin]
y_right1 <- hist_data$counts[max_freq_index]

#Plotting the diagonal line
segments(x_left1, y_left1, x_right1, y_right1, col = "blue")

#Drawing another diagonal line
x_left2 <- hist_data$breaks[max_freq_index]
x_right2 <- hist_data$breaks[right_bin]
y_left2 <- hist_data$counts[max_freq_index]
y_right2 <- hist_data$counts[right_bin]

#Plotting another diagonal line
segments(x_left2, y_left2, x_right2, y_right2, col = "blue")

#Slopes of the two lines
slope1 <- (y_right1 - y_left1) / (x_right1 - x_left1)
slope2 <- (y_right2 - y_left2) / (x_right2 - x_left2)

#Calculation of intercept of the lines
intercept1 <- y_left1 - slope1 * x_left1
intercept2 <- y_left2 - slope2 * x_left2

#Calculation of intersection point
intersection_x <- (intercept2 - intercept1) / (slope1 - slope2)

#Plot vertical line at the intersection point
abline(v = intersection_x, col = "green", lty = 2)
```



```
#Calculation of mode using the statistical method built into R
mode(aq$Temp)
```

```
## [1] "numeric"
```

```
freq_table <- table(aq$Temp)
mode <- as.numeric(names(freq_table)[freq_table == max(freq_table)])
mode
```

```
## [1] 81
```

```
intersection_x
```

```
## [1] 80.33333
```

Ans 9

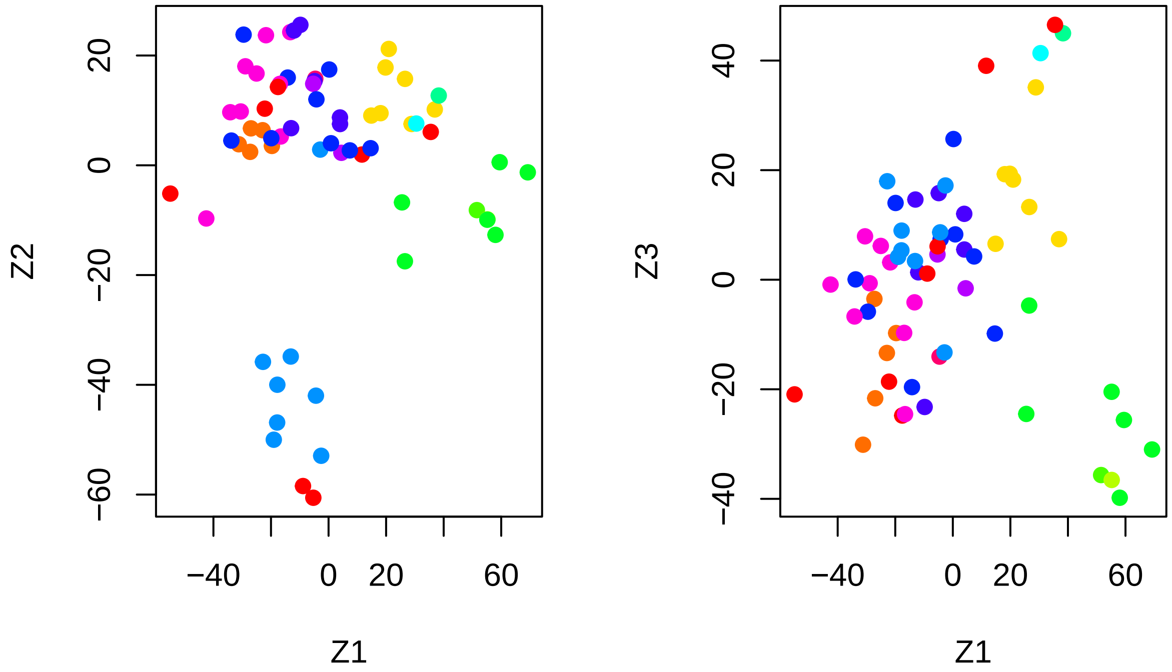
```
#ANs 9
# Qution a
library(ISLR2)
# Here we have defined nci labels (NCI$labs) as nci.labs and nci data (NCI$data) and nci.data
nci.labs <- NCI60$labs
nci.data <- NCI60$data


dim(nci.data)
```

```
## [1]   64 6830
```

```
# From the code below, we can see that the first four flower types are CNS, CNS, CNS, and RENAL.
nci.labs[1:4]
```

```
## [1] "CNS"   "CNS"   "CNS"   "RENAL"
```

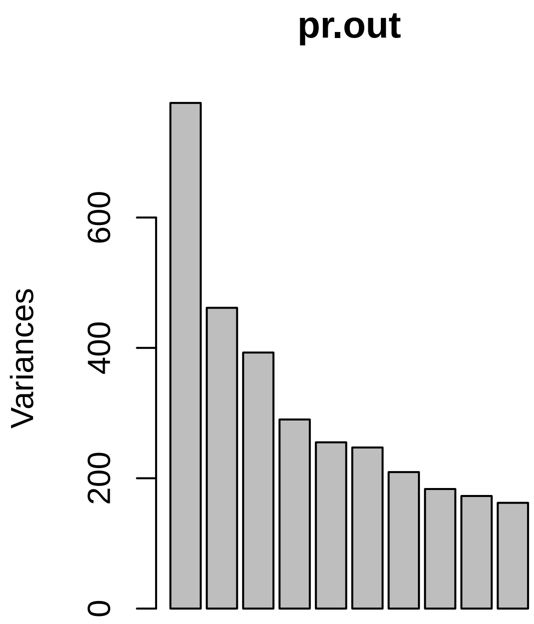
```
# This code conducts PCA on the given dataset after standardizing the variables,
# and explores for underlying patterns and relationships within the
# data through its principal components.
pr.out <- prcomp(nci.data, scale = TRUE)


# The code visualizes the principal component scores obtained from a PCA analysis of the dataset.
# The use of colors helps distinguish different classes or groups in the data.
# This aids in the interpretation of patterns or clusters in the principal component space.
Cols<- function(vec){
  cols<- rainbow(length(unique(vec)))
  return(cols[as.numeric(as.factor(vec))])
}
par(mfrow = c(1, 2))
plot(pr.out$x[, 1:2], col = Cols(nci.labs), pch = 19,
     xlab = "Z1", ylab = "Z2")
plot(pr.out$x[, c(1, 3)], col = Cols(nci.labs), pch = 19,
     xlab = "Z1", ylab = "Z3")
```



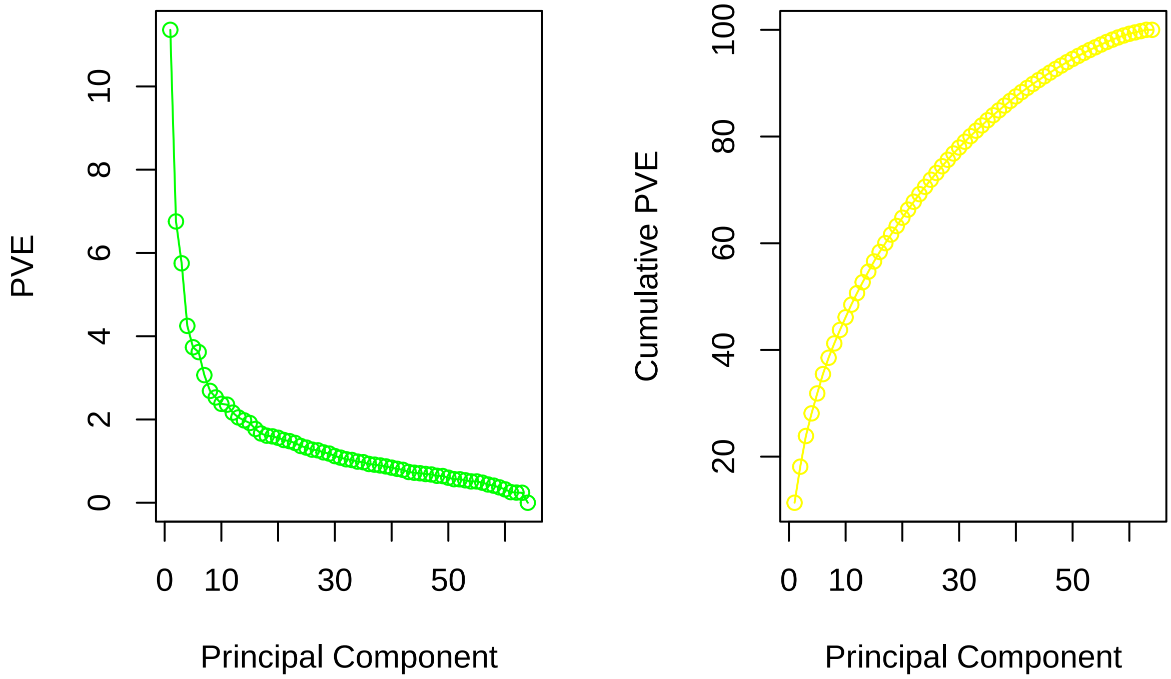
```r
# The summary of(pr.out) function in R provides a summary of the results obtained from the Principal Co
summary(pr.out)
```

```
## Importance of components:
##                            PC1      PC2      PC3      PC4      PC5      PC6
## Standard deviation     27.8535 21.48136 19.82046 17.03256 15.97181 15.72108
## Proportion of Variance  0.1136  0.06756  0.05752  0.04248  0.03735  0.03619
## Cumulative Proportion   0.1136  0.18115  0.23867  0.28115  0.31850  0.35468
##                             PC7      PC8      PC9     PC10     PC11     PC12
## Standard deviation     14.47145 13.54427 13.14400 12.73860 12.68672 12.15769
## Proportion of Variance  0.03066  0.02686  0.02529  0.02376  0.02357  0.02164
## Cumulative Proportion   0.38534  0.41220  0.43750  0.46126  0.48482  0.50646
##                            PC13     PC14     PC15     PC16     PC17     PC18
## Standard deviation     11.83019 11.62554 11.43779 11.00051 10.65666 10.48880
## Proportion of Variance  0.02049  0.01979  0.01915  0.01772  0.01663  0.01611
## Cumulative Proportion   0.52695  0.54674  0.56590  0.58361  0.60024  0.61635
##                            PC19    PC20     PC21    PC22    PC23    PC24
## Standard deviation     10.43518 10.3219 10.14608 10.0544 9.90265 9.64766
## Proportion of Variance  0.01594  0.0156  0.01507  0.0148 0.01436 0.01363
## Cumulative Proportion   0.63229  0.6479  0.66296  0.6778 0.69212 0.70575
##                           PC25    PC26    PC27   PC28    PC29    PC30    PC31
## Standard deviation     9.50764 9.33253 9.27320 9.0900 8.98117 8.75003 8.59962
## Proportion of Variance 0.01324 0.01275 0.01259 0.0121 0.01181 0.01121 0.01083
## Cumulative Proportion  0.71899 0.73174 0.74433 0.7564 0.76824 0.77945 0.79027
##                           PC32    PC33    PC34    PC35    PC36    PC37    PC38
## Standard deviation     8.44738 8.37305 8.21579 8.15731 7.97465 7.90446 7.82127
## Proportion of Variance 0.01045 0.01026 0.00988 0.00974 0.00931 0.00915 0.00896
## Cumulative Proportion  0.80072 0.81099 0.82087 0.83061 0.83992 0.84907 0.85803
##                           PC39    PC40    PC41   PC42    PC43   PC44    PC45
## Standard deviation     7.72156 7.58603 7.45619 7.3444 7.10449 7.0131 6.95839
## Proportion of Variance 0.00873 0.00843 0.00814 0.0079 0.00739 0.0072 0.00709
## Cumulative Proportion  0.86676 0.87518 0.88332 0.8912 0.89861 0.9058 0.91290
##                          PC46    PC47    PC48    PC49    PC50    PC51    PC52
## Standard deviation     6.8663 6.80744 6.64763 6.61607 6.40793 6.21984 6.20326
## Proportion of Variance 0.0069 0.00678 0.00647 0.00641 0.00601 0.00566 0.00563
## Cumulative Proportion  0.9198 0.92659 0.93306 0.93947 0.94548 0.95114 0.95678
##                           PC53    PC54    PC55    PC56    PC57   PC58    PC59
## Standard deviation     6.06706 5.91805 5.91233 5.73539 5.47261 5.2921 5.02117
## Proportion of Variance 0.00539 0.00513 0.00512 0.00482 0.00438 0.0041 0.00369
## Cumulative Proportion  0.96216 0.96729 0.97241 0.97723 0.98161 0.9857 0.98940
##                           PC60    PC61    PC62    PC63      PC64
## Standard deviation     4.68398 4.17567 4.08212 4.04124 1.951e-14
## Proportion of Variance 0.00321 0.00255 0.00244 0.00239 0.000e+00
## Cumulative Proportion  0.99262 0.99517 0.99761 1.00000 1.000e+00
```

```r
# These plots collectively provide insights into the structure of the data, the relationships between v
# and the overall effectiveness of the PCA in capturing the variability in the dataset.
plot(pr.out)



# The code present in the segment below calculates and visualizes the proportion of variance explained
# by each principal component and the cumulative proportion of variance explained.
pve <- 100 * pr.out$sdev^2 / sum(pr.out$sdev^2)
par(mfrow = c(1, 2))
```



```r
plot(pve, type = "o", ylab = "PVE",
     xlab = "Principal Component", col = "green")
plot(cumsum(pve), type = "o", ylab = "Cumulative PVE",
     xlab = "Principal Component", col = "yellow")
```



8

‘{r}

Ans 8

library(car) vif(model_lr)

Naive Bayes

install.packages(“e1071”)

library(e1071) set.seed(08)

model_nb<-naiveBayes(am~.,data=train)

predict_nb<-predict(model_nb,newdata=test) head(predict_nb) cm_nb<-table(test$am,predict_nb) cm_nb

“‘