# INF0613 – Aprendizado de Máquina Não Supervisionado

Trabalho 3 - Técnicas de Agrupamento

Daniel Noriaki Kurosawa  Eric Uyemura Suda  Fernando Shigueru Wakabayashi

O objetivo deste trabalho é exercitar o uso de algoritmos de agrupamento. Neste trabalho, vamos analisar diferentes atributos de carros com o objetivo de verificar se seus atributos são suficientes para indicar um valor de risco de seguro. O conjunto de dados já apresenta o risco calculado no campo `symboling` indicado na Tabela 1. Quanto mais próximo de 3, maior o risco. O conjunto de dados que deve ser usado está disponível na página do Moodle com o nome `imports-85.data`.

## Atividade 0 – Configurando o ambiente

Antes de começar a implementação do seu trabalho configure o *workspace* e importe todos os pacotes e execute o preprocessamento da base:

```
# Adicione os pacotes usados neste trabalho:
#install packages
#install.packages("apcluster")
#install.packages("fclust")
#install.packages("ppclust")
#install.packages("dbscan")
#install.packages("gridExtra")
#install.packages("fpc")
#install.packages("NbClust")
#install.packages("factoextra")
#install.packages("NbClust")

library(dplyr)
library(ggplot2)
library(factoextra)
library(NbClust)
library(fpc)
library(gridExtra)
library(dbscan)
library(ppclust)
library(fclust)
library(apcluster)
library ( NbClust )
library(reshape2)


#Funções extras
onehot_features <- function(dataset, feature){
    cats <- unique(dataset[, feature])
    for (cat in cats){
```

```
        dataset[cat] <- as.numeric(dataset[,feature]==cat)
    }
    return(dataset)
}

min_max <- function(df, columns){
  min_features <- apply(df[,colnames(df) %in% columns], 2, min); min_features
  max_features <- apply(df[,colnames(df) %in% columns], 2, max); max_features
  diff <- max_features - min_features; diff
  df[,colnames(df) %in% columns] <- sweep(df[,colnames(df) %in% columns], 2, min_features, "-")
  df[,colnames(df) %in% columns] <- sweep(df[,colnames(df) %in% columns], 2, diff, "/")
  return(df)
}

normalization <- function(df, columns){
  mean_features <- apply(df[,colnames(df) %in% columns], 2, mean);mean_features
  sd_features <- apply(df[,colnames(df) %in% columns], 2, sd); sd_features
  df[,colnames(df) %in% columns] <- sweep(df[,colnames(df) %in% columns], 2, mean_features, "-")
  df[,colnames(df) %in% columns] <- sweep(df[,colnames(df) %in% columns], 2, sd_features, "/")
  return(df)
}

`%notin%` <- Negate(`%in%`)

#trainSet[,2:ncol(trainSet)] <- sweep(trainSet[,2:ncol(trainSet)], 2, mean_features, "-")
#trainSet[,2:ncol(trainSet)] <- sweep(trainSet[,2:ncol(trainSet)], 2, sd_features, "/")


# Configure ambiente de trabalho na mesma pasta
# onde colocou a base de dados:
# setwd("")
# setwd("C:\\Users\\Eric\\Documents\\GitHub\\mineiracao_dados_complexos\\Aprendizado de Maquina Nao Sup
# setwd("/Users/nkuros/Documents/mineiracao_dados_complexos/Aprendizado de Maquina Nao Supervisionado/T
```

## Atividade 1 – Análise e Preparação dos Dados

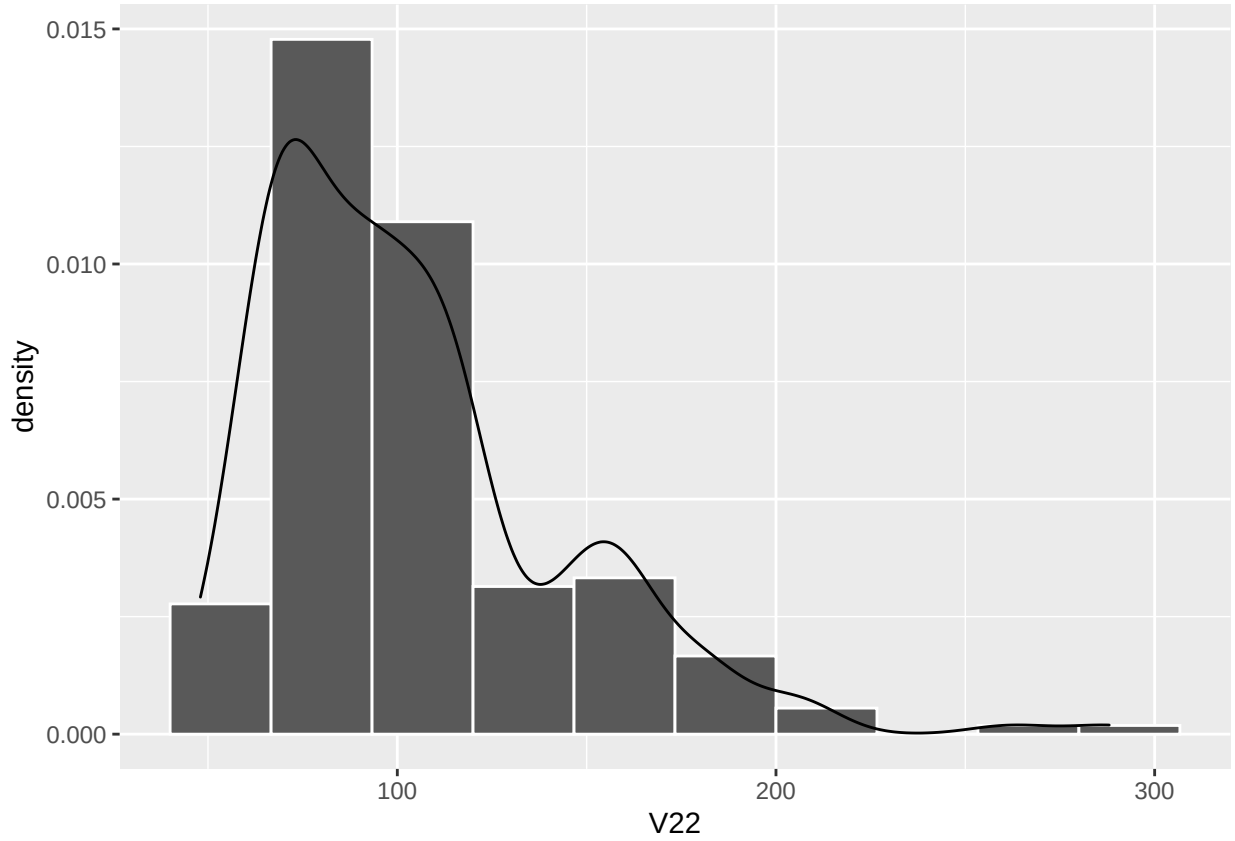
O conjunto de dados é composto por 205 amostras com 26 atributos cada descritos na Tabela 1. Os atributos são dos tipos `factor`, `integer` ou `numeric`. O objetivo desta etapa é a análise e preparação desses dados de forma a ser possível agrupá-los nas próximas atividades.

**Implementações:** Nos itens a seguir você implementará a leitura da base e aplicará tratamentos básicos.

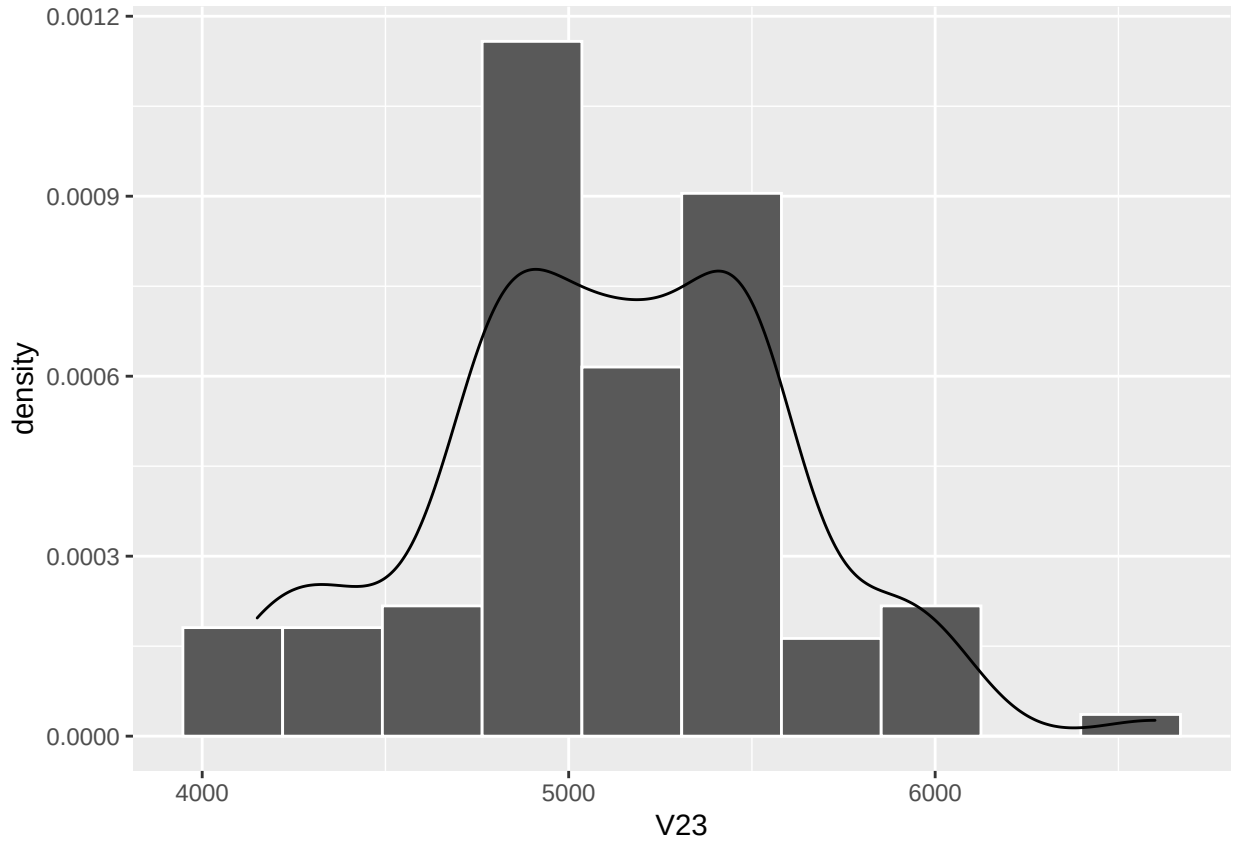
a) *Tratamento de dados Incompletos:* Amostras incompletas deverão ser tratadas, e você deve escolher a forma que achar mais adequada. Considere como uma amostra incompleta uma linha na qual faltam dados em alguma das colunas selecionadas anteriormente. Note que, dados faltantes nas amostras podem causar uma conversão do tipo do atributo de todas as amostras e isso pode impactar no item b).

```
# Leitura da base
df <- read.table('imports-85.data',sep=',')
set.seed (12345)
# Tratamento de dados faltantes
```

```r
# Avaliacao do dataset
#summary(df)
#head(df)

# Avalaiacao dos valores unicos de cada coluna
#for(col in colnames(df)){
#  print(col)
#  print(unique(df[,col]))
#  print('\n')
#}

#-------------------------------------------------------------------------
# Tratando "?" que apresenta nas bases para poder transformar em numerico
df$V2 <- as.numeric(gsub(pattern='[?]', replacement=NA, df$V2))
df$V6 <- gsub(pattern='[?]', replacement=NA, df$V6)
df$V19 <- as.numeric(gsub(pattern='[?]', replacement=NA, df$V19))
df$V20 <- as.numeric(gsub(pattern='[?]', replacement=NA, df$V20))
df$V22 <- as.numeric(gsub(pattern='[?]', replacement=NA, df$V22))
df$V23 <- as.numeric(gsub(pattern='[?]', replacement=NA, df$V23))
df$V26 <- as.numeric(gsub(pattern='[?]', replacement=NA, df$V26))

#unique(df$V6)
#unique(df$V18)

#----------------------------------------
# Data set de teste retirando todos os NA
df2 <- na.omit(df)
#summary(df2)
#----------------------------------------

# NA de V19 e V20 sao para a mesma marca mazda
#df[is.na(df$V19),]
#df[df$V3=='mazda',]
# Agrupando os valores para obter a m[edia de V19 e V20]
linhas = df$V3=='mazda' & !is.na(df$V19)
colunas = c('V3', 'V19', 'V20')
media_mazda = df[linhas,colunas] %>% group_by(V3) %>% summarise(media_v19=median(V19), media_V20=median
# Substituindo V19 e V20 que estao NA pela media
df[df$V3=='mazda' & is.na(df$V19),'V19'] = media_mazda$media_v19
df[df$V3=='mazda' & is.na(df$V20),'V20'] = media_mazda$media_V20

#-----------------------------------------------------------------------------------------
# Como temos apenas 2 carros da marca renault, vamos observar a distribuicao de V22 e V23
#df[is.na(df$V22),]
#df[df$V3=='renault',]

# Distribuicoes nao normais e outliers, vamos pela mediana
ggplot(df, aes(x=V22, y= ..density..)) +
  geom_histogram(color='White', bins=10) +
  geom_density()
```




```
ggplot(df, aes(x=V23, y= ..density..)) +
  geom_histogram(color='White', bins=10) +
  geom_density()
```

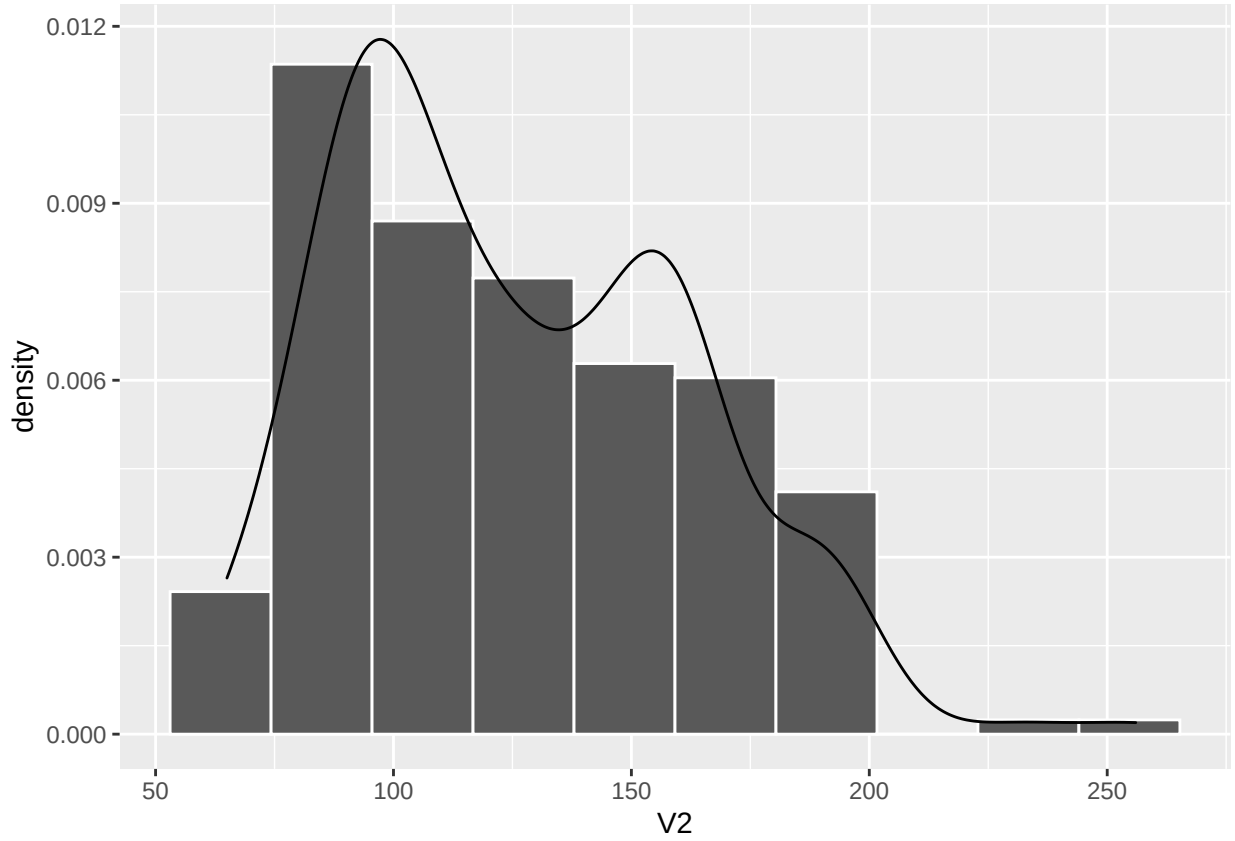



```
linhas <- !is.na(df$V22)
colunas <- c('V22', 'V23')
media_renault <- df[linhas,colunas] %>% summarise(mediana_v22=median(V22), mediana_V23=median(V23))

df[df$V3=='renault' & is.na(df$V22),'V22'] = media_renault$mediana_v22
df[df$V3=='renault' & is.na(df$V23),'V23'] = media_renault$mediana_V23

#---------------------------------------
# Verificacao dos veiculos com preco NA
#df[is.na(df$V26),]
# Marcas para verificar 'audi', 'isuzu', 'porsche'
#df[df$V3 %in% c('audi', 'isuzu', 'porsche'),]

#df[df$V3 %in% c('audi', 'isuzu', 'porsche') & !is.na(df$V26), c('V3', 'V26')]
#    %>% group_by(V3)
#    %>% summarise(max=max(V26), min=min(V26), mean=mean(V26), median=median(V26))

linhas <- df$V3 %in% c('audi', 'isuzu', 'porsche') & !is.na(df$V26)
colunas <- c('V3', 'V26')
median_marcas <- df[linhas, colunas] %>% group_by(V3) %>% summarise(median_26=median(V26))

for (marca in unique(median_marcas$V3)){
  df[df$V3==marca & is.na(df$V26),'V26'] <- median_marcas[median_marcas$V3==marca,'median_26']
}

#----------------------
```

```r
# Verificando coluna V2
teste <- df[df$V3 %in% unique(df[is.na(df$V2),'V3']),]
# teste

# Para as marcas que possuem mais carros com V2 nao nulo, tiramos a media
linhas <- df$V3 %in% unique(df[is.na(df$V2),'V3']) & !is.na(df$V2)
colunas <- c('V3', 'V2')
media_marcas <- df[linhas, colunas] %>% group_by(V3) %>% summarise(median_2=median(V2))
media_marcas
```

```
## # A tibble: 11 x 2
##    V3            median_2
##    <chr>            <dbl>
##  1 audi               161
##  2 bmw                190
##  3 jaguar             145
##  4 mazda              115
##  5 mercedes-benz       93
##  6 mitsubishi         153
##  7 peugot             161
##  8 plymouth           136.
##  9 porsche            186
## 10 toyota              91
## 11 volkswagen          94
```

```r
for (marca in unique(media_marcas$V3) ){
  df[df$V3==marca & is.na(df$V2),'V2'] <- media_marcas[media_marcas$V3==marca,'median_2']
}

# Para as marcas em que todos os carros apresentam V2 nulo, tiramos a mediana da base
ggplot(df, aes(x=V2, y= ..density..)) +
  geom_histogram(color='White', bins=10) +
  geom_density()
```




```
linhas <- !is.na(df$V2)
colunas <- 'V2'
mediana_marcas <- median(df[linhas, colunas])
mediana_marcas
```

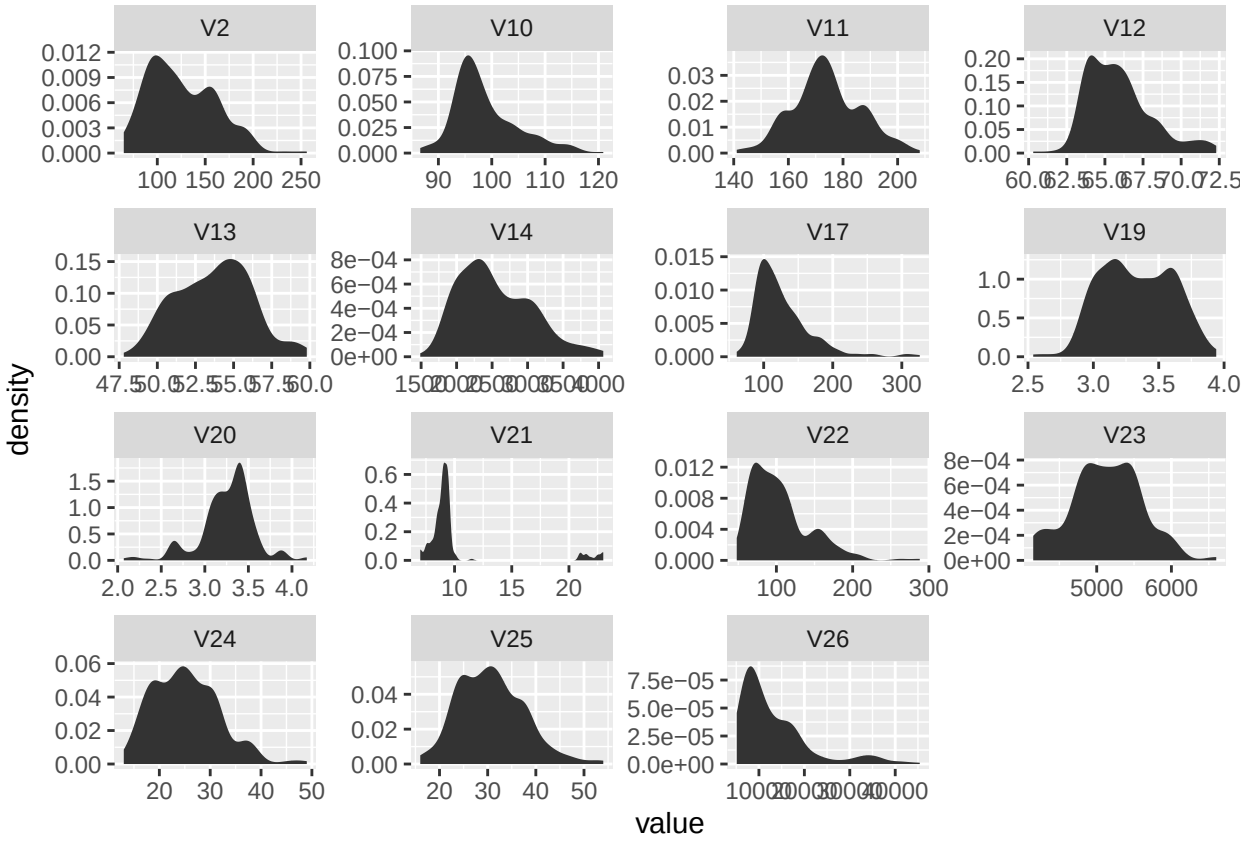
```
## [1] 119
```

```
for (marca in unique(df[is.na(df$V2), 'V3'])) {
  df[df$V3==marca & is.na(df$V2),'V2'] <- mediana_marcas
}
# retirando duas linhas
# df <- na.omit(df, cols='V6')

# Validacao do conteudo de cada coluna para ver se tratamos todos os casos de [?]
# for(col in colnames(df)){
#   print(col)
#   print(unique(df[,col]))
#   print('-----------------------------')
# }

# Construindo um gráfico com as distâncias intra-cluster
# colunas_porta = c("V2", "V10", "V11", "V12", "V13", "V14", "V17", "V19", "V20", "V21", "V22", "V23",
# colunas_tracao = c("V2", "V10", "V11", "V12", "V13", "V14", "V17", "V19", "V20", "V21", "V22", "V23",
# colunas_cilindro = c("V2", "V10", "V11", "V12", "V13", "V14", "V17", "V19", "V20", "V21", "V22", "V23
colunas_full = c("V2", "V10", "V11", "V12", "V13", "V14", "V17", "V19", "V20", "V21", "V22", "V23", "V2
```

```
#colunas = c("V2", "V10", "V11", "V12", "V13", "V14", "V17", "V19", "V20", "V21", "V22", "V23", "V24",
colunas = c("V2", "V10", "V11", "V12", "V13", "V14", "V17", "V19", "V20", "V21", "V22", "V23", "V24", "
numericas = c("V2", "V10", "V11", "V12", "V13", "V14", "V17", "V19", "V20", "V21", "V22", "V23", "V24",


# Normalizacao minmax das variaveis numericas
df_normalized <- min_max(df,numericas)

# df_normalized <- scale(df[, numericas])
# df_normalized <- cbind(df_normalized, df[,(colnames(df) %notin% numericas) ])

# df_normalized <- normalization(df, numericas)

# summary(df_normalized)
# colnames(df_normalized)
```

b) *Seleção de Atributos:* Atributos não-numéricos não podem ser usados com as técnicas agrupamento vistas em aula. Portanto, você deve selecionar um conjunto de atributos numéricos que serão usados para o agrupamento. Além disso você deve analisar se os atributos não-numéricos são descritivos para a realização dos agrupamentos. Caso um dos atributos não numéricos seja necessário, use a técnica do *one hot encoding* para transformá-lo em numérico. **Não** aplique essa técnica nos atributos `symboling` e `make` para os agrupamentos subsequentes, eles não devem fazer parte do agrupamento.

```
# Seleção de atributos

# Iremos criar o onehot encoding para todas as features categoricas, assim podemos testar a sua eficien

# Criando features Categoricas via One Hot Encoding
# df_normalized = onehot_features(df_normalized, 'V3')
df_normalized = onehot_features(df_normalized, 'V4')
df_normalized = onehot_features(df_normalized, 'V5')
# df_normalized = onehot_features(df_normalized, 'V6')
df_normalized = onehot_features(df_normalized, 'V7')
df_normalized = onehot_features(df_normalized, 'V8')
df_normalized = onehot_features(df_normalized, 'V9')
df_normalized = onehot_features(df_normalized, 'V15')
df_normalized = onehot_features(df_normalized, 'V16')
df_normalized = onehot_features(df_normalized, 'V18')

melt_train_set <- df[, numericas]
melt_train_set <- melt(melt_train_set)

p <- ggplot(data=melt_train_set, aes(x=value))+
    stat_density()+
    facet_wrap(~variable, scales='free'); p
```




```
numericas_reduzido <- c("V2", "V10", "V11", "V12", "V13", "V14", "V19", "V23", "V24", "V25", "V26")

#-----------------------------------------------
# Teste com PCA para variaveis categoricas
# Codigo comentado para nao plotar no relatorio
#-----------------------------------------------
# categoricas_reduzido <- c("gas", "turbo", "convertible", "hatchback", "sedan", "wagon", "4wd", "fwd",
# pc <- prcomp(df_normalized[, c(numericas_reduzido,categoricas_reduzido)], center = TRUE,scale. = TRUE
# en <- cumsum(pc$sdev / sum(pc$sdev)); en

# z <- pc$x[, 1:25]
# z <- scale(z)
# z <- min_max(z, colnames(z))

# nb <- NbClust(z, distance="euclidean", min.nc=2, max.nc=30, method="complete", index="all")

# fviz_nbclust(nb) + theme_minimal()

# summary(df_normalized)
```

## Análises

Após as implementações escreva uma análise da base de dados. Em especial, descreva o conjunto de dados inicial, relate como foi realizado o tratamento, liste quais os atributos escolhidos para manter na base e descreva a base de dados após os tratamentos listados. Explique todos os passos executados, mas sem

copiar códigos na análise. Além disso justifique suas escolhas de tratamento nos dados faltantes e seleção de atributos.

**Resposta:** Primeiramente avaliamos como estava a qualidade da informação na base de dados através da função summary onde observamos que muitas variáveis que eram para ser numéricas foram convertidas para classe de caracteres por causa dos valores "[?]" que vieram no lugar de NA. Logo substituimos estes valores errados por NA e convertemos as colunas em numéricas. Em seguida avaliamos a quantidade de NAs presente nas colunas numéricas (V2, V19, V20, V22, V23, V26), seguimos a seguinte metodologia para estas variáveis, aonde havia a informação de marca tiravamos a mediana dentro da própria marca da variável e quando havia apenas um exemplo daquela marca ou todos os exemplos daquela marca estavam com a variável marcada como NA tiramos a mediana do dataset inteiro. Preferimos utilizar a mediana para diminuir a influência dos outliers. Após o tratamento dos NAs normalizamos as variáveis numéricas com a técnica min-max e transformamos também as variáveis catergóricas através do one-hot encoding e testamos em um data set apartado a transformação via PCA (após os testes com os modelos decidimos seguir apenas com as variáveis continuas). Por fim, após analisar a distribuição de todas as variáveis numéricas, optamos por remover também as variáveis que visualmente apresentava uma distribuição muito diferente da normal e com outliers.

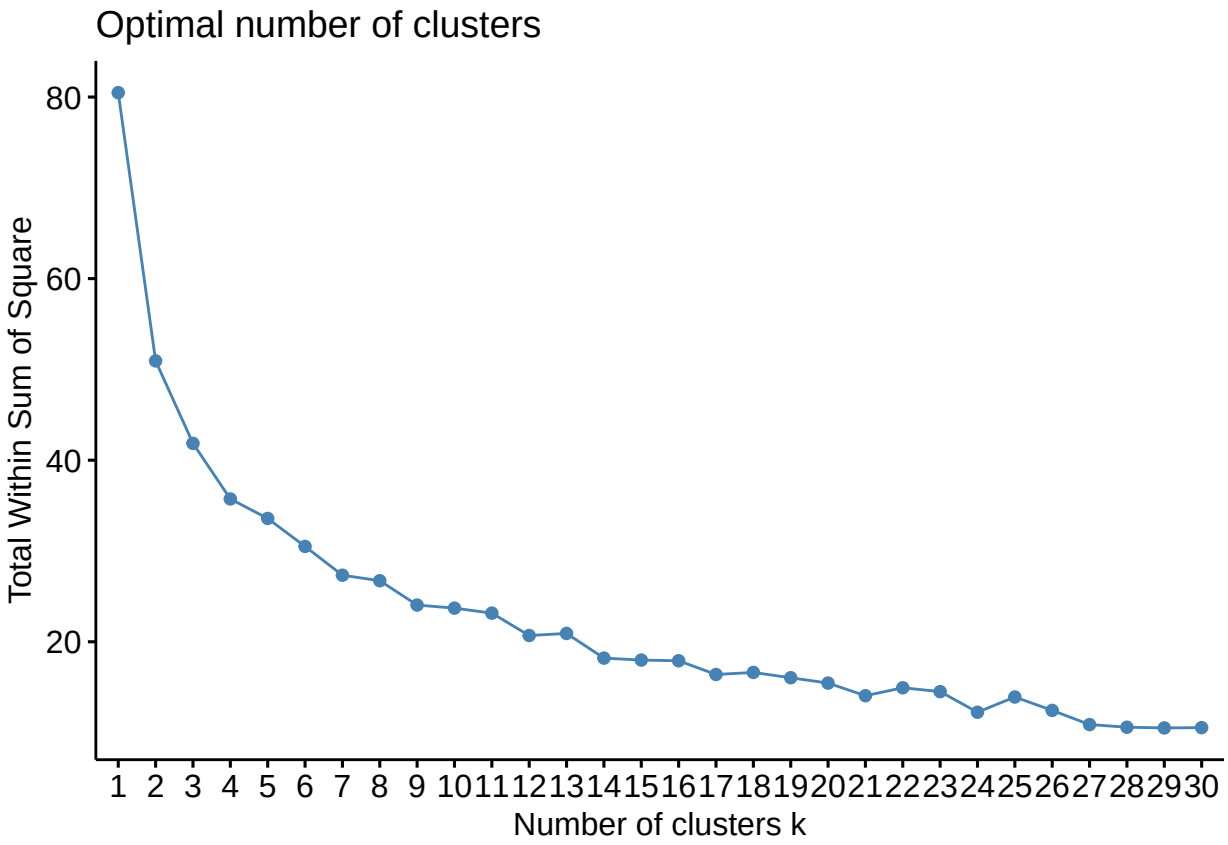
## Atividade 2 – Agrupamento com o $K$-*means*

Nesta atividade, você deverá agrupar os dados com o algoritmo $K$-*means* e utilizará duas métricas básicas para a escolha do melhor $K$: a soma de distâncias intra-cluster e o coeficiente de silhueta.

**Implementações:** Nos itens a seguir você implementará a geração de gráficos para a análise das distâncias intra-cluster e do coeficiente de silhueta. Em seguida, você implementará o agrupamento dos dados processados na atividade anterior com o algoritmo $K$-*means* utilizando o valor de $K$ escolhido.

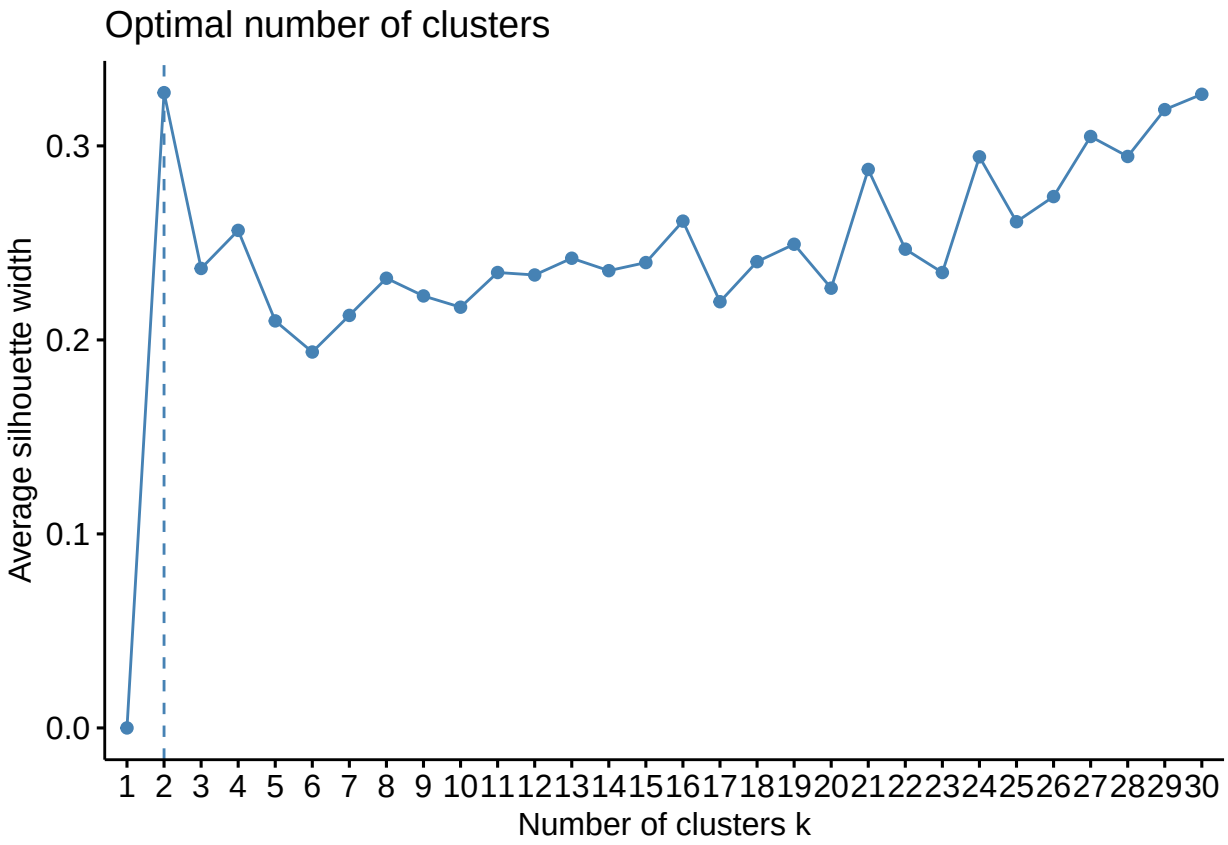
a) *Gráfico Elbow Curve:* Construa um gráfico com a soma das distâncias intra-cluster para $K$ variando de 2 a 30.

```
base_1 = df_normalized[, numericas_reduzido]

fviz_nbclust(base_1, kmeans, method="wss",k.max=30)
```



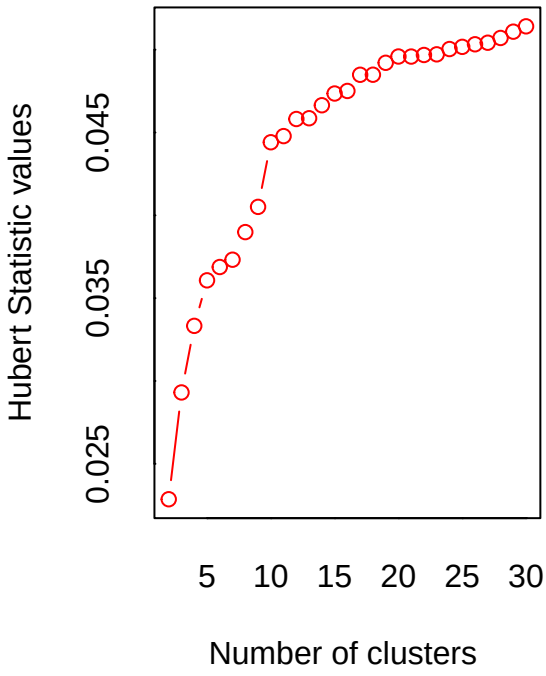

b) *Gráfico da Silhueta:* Construa um gráfico com o valor da silhueta para $K$ variando de 2 a 30.

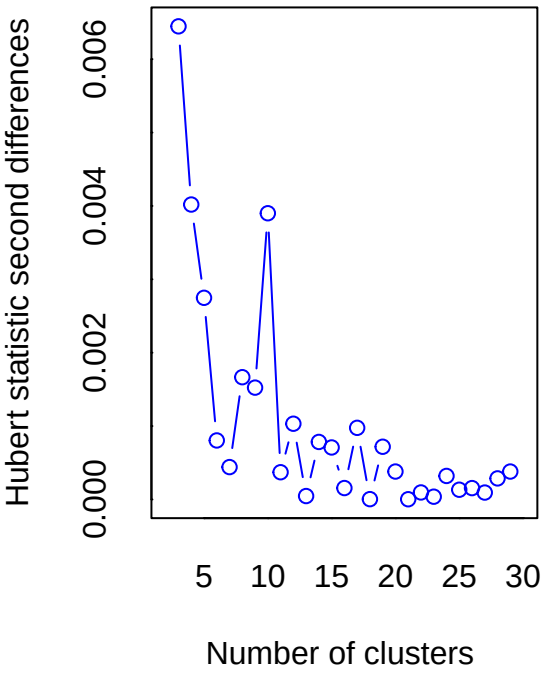
```
# Construindo um gráfico com os valores da silhueta

fviz_nbclust(base_1, kmeans, method="silhouette", k.max=30)
```



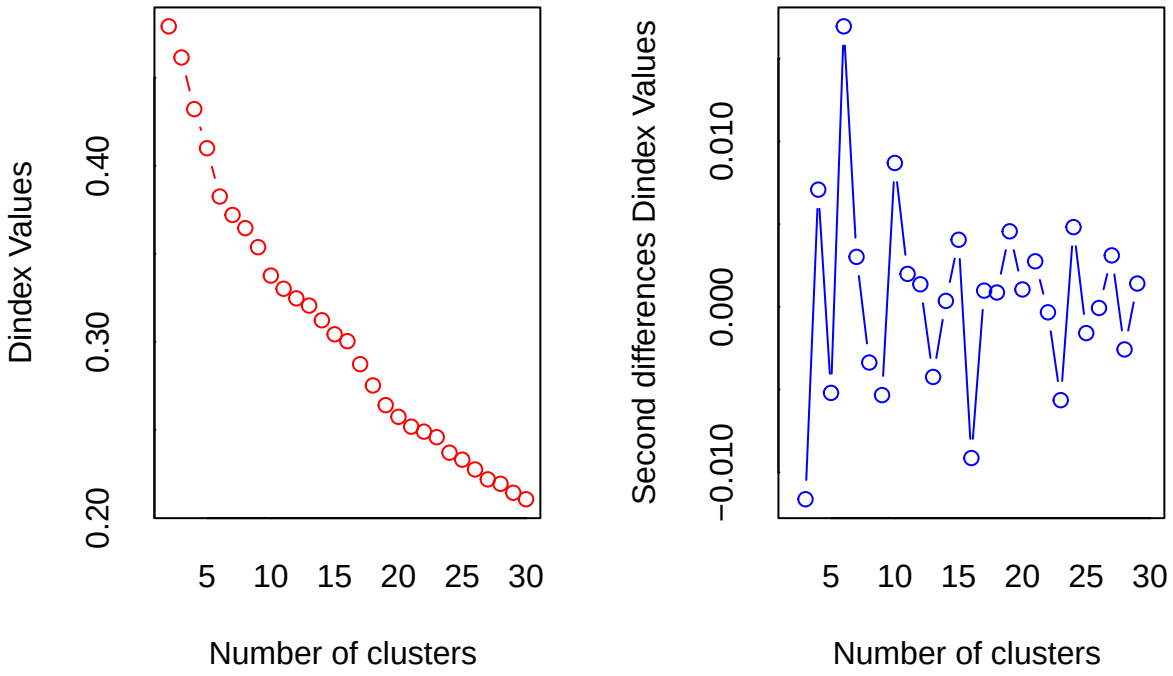

c) *Escolha do K:* Avalie os gráficos gerados nos itens anteriores e escolha o melhor valor de $K$ com base nas informações desses gráficos e na sua análise. Se desejar, use também a função `NbClust` para ajudar nas análises. Com o valor de $K$ definido, utilize o rótulo obtido para cada amostra, indicando o grupo ao qual ela pertence, para gerar um gráfico de dispersão (atribuindo cores diferentes para cada grupo).

```
# Algoritmo de avaliacao do melhor K a ser escolhido
# K = 2 escolhido como K otimo
nb <- NbClust ( base_1 , distance ="euclidean",min.nc =2 , max.nc =30 , method ="complete",index ="all"
```



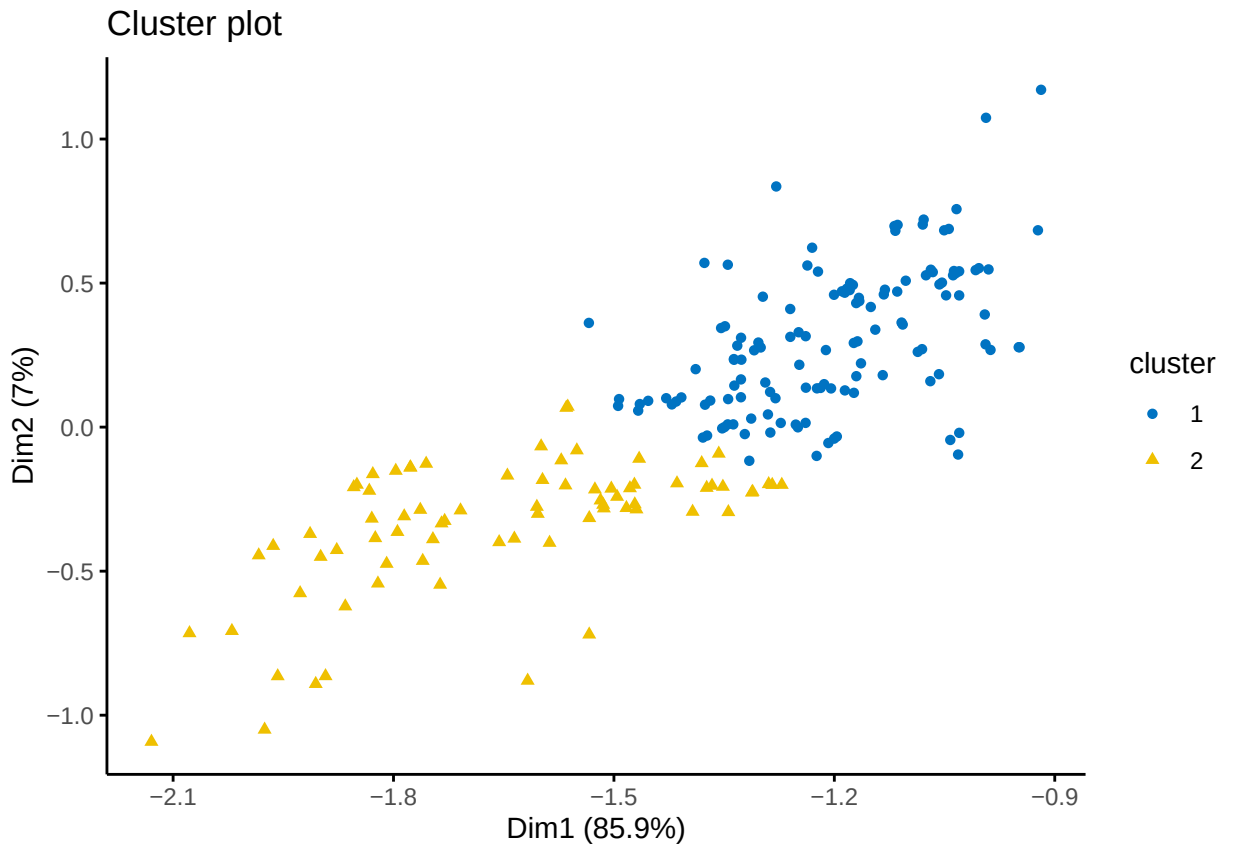

```
## *** : The Hubert index is a graphical method of determining the number of clusters.
##                 In the plot of Hubert index, we seek a significant knee that corresponds to a
##                 significant increase of the value of the measure i.e the significant peak in Hubert
##                 index second differences plot.
##
```




```
## *** : The D index is a graphical method of determining the number of clusters.
##                 In the plot of D index, we seek a significant knee (the significant peak in Dindex
##                 second differences plot) that corresponds to a significant increase of the value of
##                 the measure.
##
## *******************************************************************
## * Among all indices:
## * 7 proposed 2 as the best number of clusters
## * 3 proposed 3 as the best number of clusters
## * 3 proposed 4 as the best number of clusters
## * 2 proposed 6 as the best number of clusters
## * 1 proposed 24 as the best number of clusters
## * 1 proposed 29 as the best number of clusters
## * 6 proposed 30 as the best number of clusters
##
##                     ***** Conclusion *****
##
## * According to the majority rule, the best number of clusters is  2
##
##
## *******************************************************************
```

```r
# Aplicando o k-means com o k escolhido
km <- eclust(base_1, "kmeans", k=2,nstart=25, graph=FALSE)
```

```
# Construindo um gráfico de dispersão
fviz_cluster(km, data=base_1, stand=FALSE,
             ellipse=FALSE, show.clust.cent=FALSE,
             geom="point", palette="jco",
             ggtheme=theme_classic())
```




```
# Teste dos labels vs clusters gerados
base_kmeans_test <- kmeans(base_1 , 2, nstart = 25)
test_df <- as.data.frame.matrix(table(base_kmeans_test$cluster, df_normalized$V1));test_df
```

```
##   -2 -1  0  1  2  3
## 1  3 14 28  8  7 16
## 2  0  8 39 46 25 11
```

```
test_df2 <- as.data.frame.matrix(table(base_kmeans_test$cluster, df_normalized$V3));test_df2
```

```
##   alfa-romero audi bmw chevrolet dodge honda isuzu jaguar mazda mercedes-benz
## 1           0    6   8         0     1     0     0      3     1             8
## 2           3    1   0         3     8    13     4      0    16             0
##   mercury mitsubishi nissan peugot plymouth porsche renault saab subaru toyota
## 1       1          3      6     11        1       5       0    5      0      5
## 2       0         10     12      0        6       0       2    1     12     27
##   volkswagen volvo
## 1          1    11
## 2         11     0
```

### Análises

Descreva cada um dos gráficos gerados nos itens acima e analise-os. Inclua na sua análise as informações mais importantes que podemos retirar desses gráficos. Discuta sobre a escolha do valor $K$ e sobre a apresentação dos dados no gráfico de dispersão.

**Resposta:** Observando a soma das distâncias intra cluster, supomos que o k ótimo se dá entre 2 e 10 aproximadamente (região de cotovelo). Ao gerar o grafico de silhueta, o valor k=2 foi apontado como melhor escolha (maior valor de largura de silhueta). Finalmente, tal valor foi corroborado pelo NbClust, em que 7 dos parametros indicaram 2 como o valor ótimo. Adicionalmente, podemos perceber pelo gráfico da derivada do D index gerado (grafico a direita) que o valor mais negativo da segunda derivada se dá em k=2 indicando ser o ponto de maior desaceleração da curva D (grafico à esquerda)

## Atividade 3 – Agrupamento com o *DBscan*

Nesta atividade, você deverá agrupar os dados com o algoritmo *DBscan*. Para isso será necessário experimentar com diferentes valores de *eps* e *minPts*.

a) *Ajuste de Parâmetros:* Experimente com valores diferentes para os parâmetros *eps* e *minPts*. Verifique o impacto dos diferentes valores nos agrupamentos.

```
# Experimento com valores de eps e minPts
 db <- dbscan :: dbscan ( base_1 , eps =0.5, minPts =2)
 print(db)
```

```
## DBSCAN clustering for 205 objects.
## Parameters: eps = 0.5, minPts = 2
## The clustering contains 5 cluster(s) and 2 noise points.
##
##   0   1   2   3   4   5
##   2 188   2   2   9   2
##
## Available fields: cluster, eps, minPts
```

```
 # Experimento com valores de eps e minPts
 db <- dbscan :: dbscan ( base_1 , eps =0.5, minPts =3)
 print(db)
```

```
## DBSCAN clustering for 205 objects.
## Parameters: eps = 0.5, minPts = 3
## The clustering contains 2 cluster(s) and 8 noise points.
##
##   0   1   2
##   8 188   9
##
## Available fields: cluster, eps, minPts
```

```
# Experimento com valores de eps e minPts
 db <- dbscan :: dbscan ( base_1 , eps =0.5, minPts =4)
 print(db)
```

```
## DBSCAN clustering for 205 objects.
## Parameters: eps = 0.5, minPts = 4
## The clustering contains 2 cluster(s) and 8 noise points.
##
##   0   1   2
##   8 188   9
##
## Available fields: cluster, eps, minPts
```

```
# Experimento com valores de eps e minPts
 db <- dbscan :: dbscan ( base_1 , eps =0.5, minPts =5)
 print(db)
```

```
## DBSCAN clustering for 205 objects.
## Parameters: eps = 0.5, minPts = 5
## The clustering contains 2 cluster(s) and 10 noise points.
##
##   0   1   2
##  10 186   9
##
## Available fields: cluster, eps, minPts
```

```
 # Experimento com valores de eps e minPts
 db <- dbscan :: dbscan ( base_1 , eps=1 , minPts=4)
 print(db)
```

```
## DBSCAN clustering for 205 objects.
## Parameters: eps = 1, minPts = 4
## The clustering contains 1 cluster(s) and 0 noise points.
##
##   1
## 205
##
## Available fields: cluster, eps, minPts
```

```
 # Experimento com valores de eps e minPts
 db <- dbscan :: dbscan ( base_1 , eps =0.5, minPts =4)
 print(db)
```

```
## DBSCAN clustering for 205 objects.
## Parameters: eps = 0.5, minPts = 4
## The clustering contains 2 cluster(s) and 8 noise points.
##
##   0   1   2
##   8 188   9
##
## Available fields: cluster, eps, minPts
```

```
# Experimento com valores de eps e minPts
 db <- dbscan :: dbscan ( base_1 , eps =0.25, minPts =4)
 print(db)
```

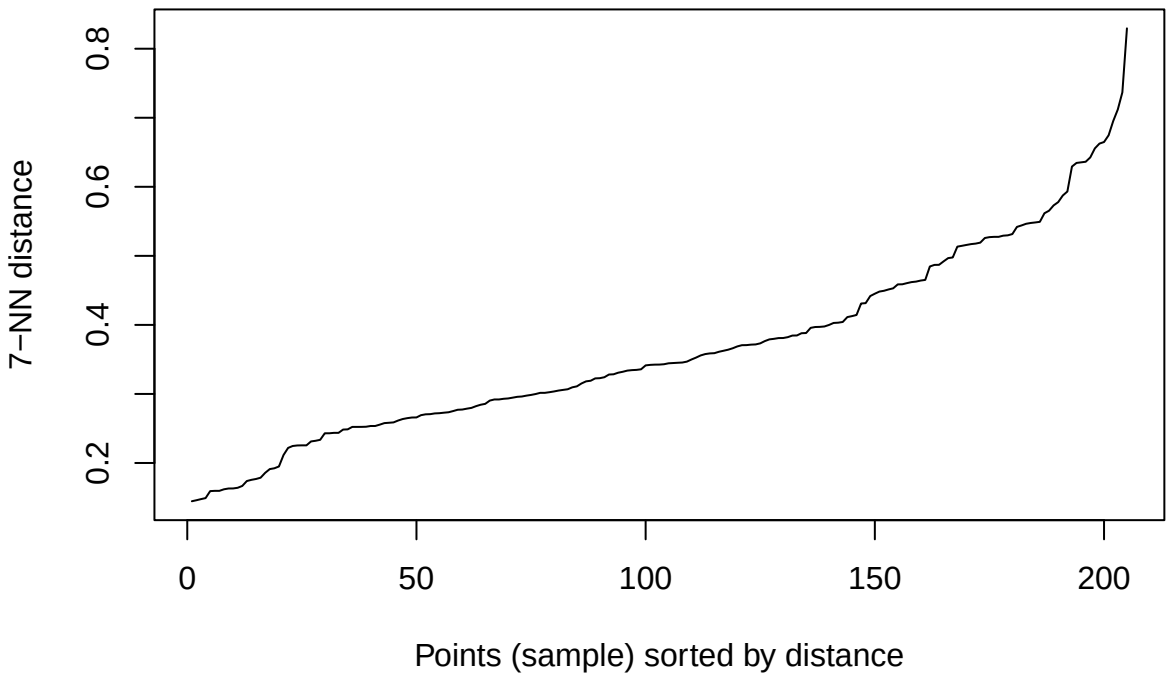
```
## DBSCAN clustering for 205 objects.
## Parameters: eps = 0.25, minPts = 4
## The clustering contains 10 cluster(s) and 86 noise points.
##
##  0  1  2  3  4  5  6  7  8  9 10
## 86 17  4  5 39 24  4  4  9  5  8
##
## Available fields: cluster, eps, minPts
```

```
# Experimento com valores de eps e minPts
 db <- dbscan :: dbscan ( base_1 , eps = 0.7, minPts =3)
 print(db)
```

```
## DBSCAN clustering for 205 objects.
## Parameters: eps = 0.7, minPts = 3
## The clustering contains 1 cluster(s) and 0 noise points.
##
##   1
## 205
##
## Available fields: cluster, eps, minPts
```

b) *Determinando Ruídos:* Escolha o valor de *minPts* que obteve o melhor resultado no item anterior e use a função `kNNdistplot` do pacote `dbscan` para determinar o melhor valor de *eps* para esse valor de *minPts*. Lembre-se que o objetivo não é remover todos os ruídos.

```
# Encontrando o melhor eps com o kNNdistplot
dbscan :: kNNdistplot (base_1 , k = 7)
```

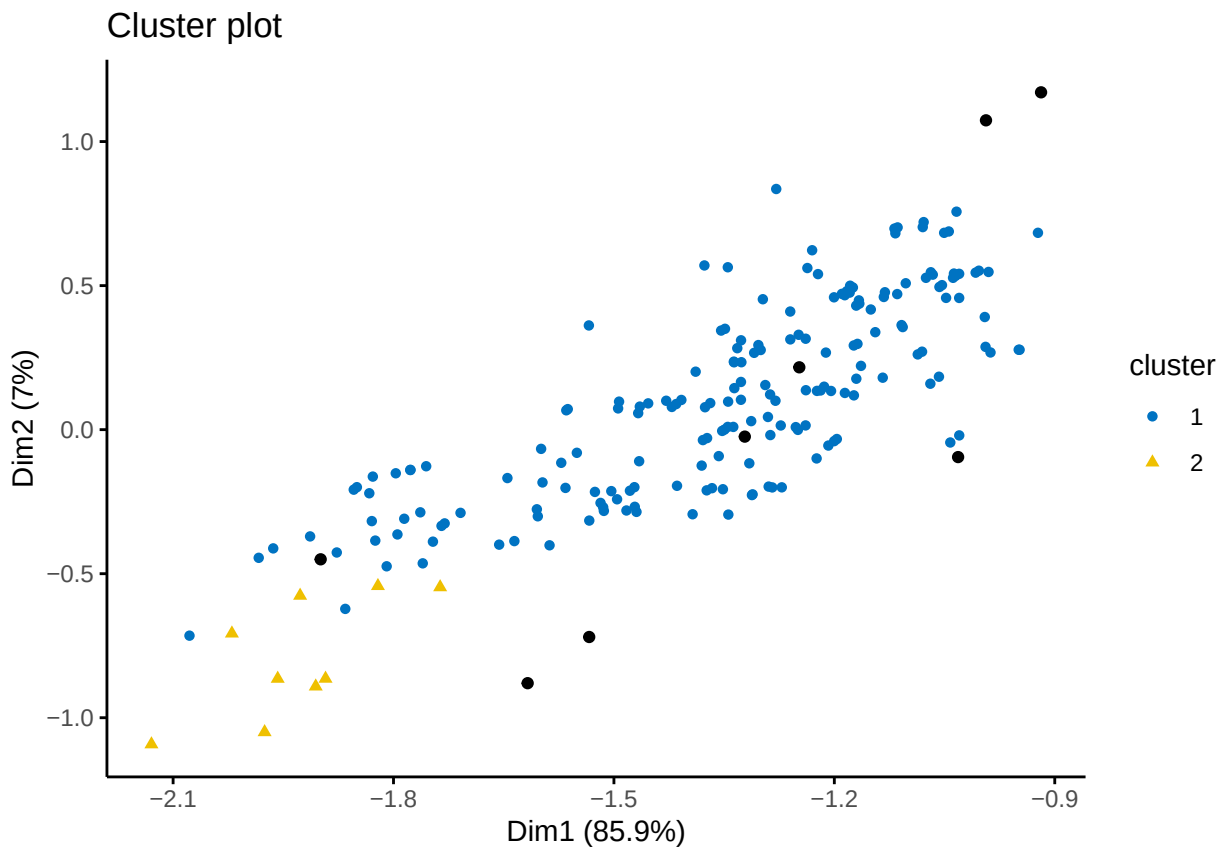



c) *Visualizando os Grupos:* Após a escolha dos parâmetros *eps* e *minPts*, utilize o rótulo obtido para cada amostra, indicando o grupo ao qual ela pertence, para gerar um gráfico de dispersão (atribuindo cores diferentes para cada grupo).

```
# Aplicando o DBscan com os parâmetros escolhidos
 db <- dbscan :: dbscan ( base_1 , eps =0.5 , minPts =4)
 print(db)
```

```
## DBSCAN clustering for 205 objects.
## Parameters: eps = 0.5, minPts = 4
## The clustering contains 2 cluster(s) and 8 noise points.
##
##   0   1   2
##   8 188   9
##
## Available fields: cluster, eps, minPts
```

```
# Construindo um gráfico de dispersão
fviz_cluster(db, data=base_1, stand=FALSE,
             ellipse=FALSE, show.clust.cent=FALSE,
             geom="point", palette="jco",
             ggtheme=theme_classic())
```




```
#analise matriz cruzada

#base_kmeans_test <- kmeans(base_1 , 3, nstart = 20)
analise <- as.data.frame.matrix(table(db$cluster, df_normalized$V1));analise
```

```
##   -2 -1  0  1  2  3
## 0  0  0  1  2  2  3
## 1  3 18 62 51 30 24
## 2  0  4  4  1  0  0
```

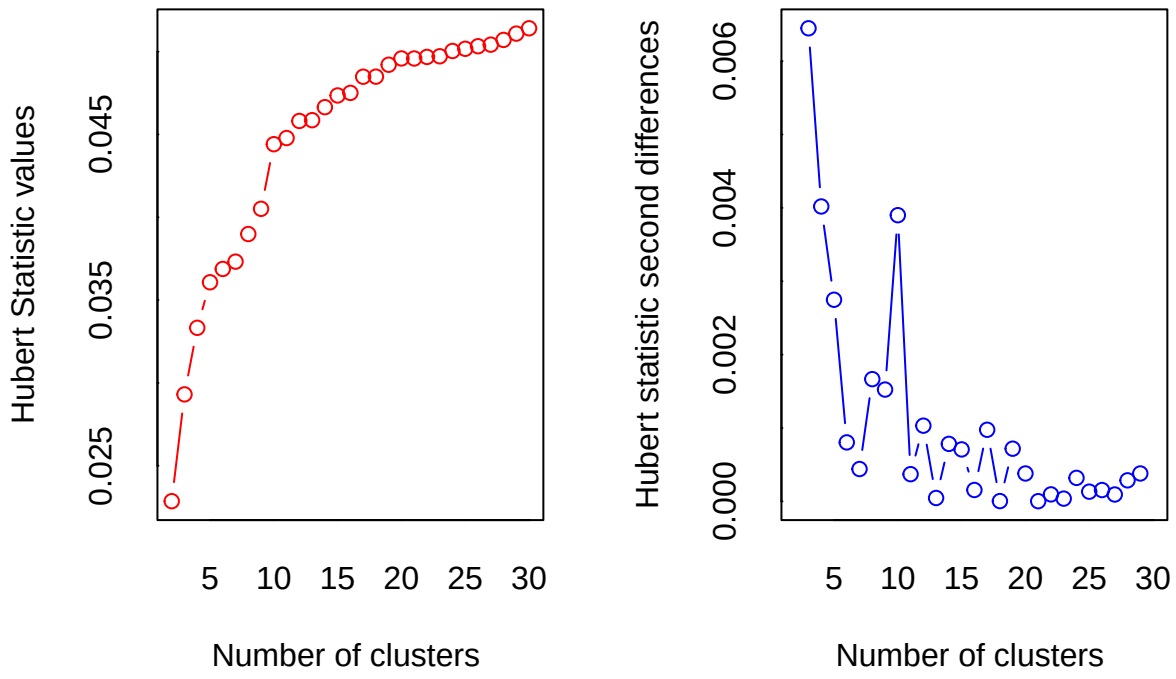
### Análises

Descreva os experimentos feitos para a escolha dos parâmetros *eps* e *minPts*. Inclua na sua análise as informações mais importantes que podemos retirar dos gráficos gerados. Justifique a escolha dos valores dos parâmetros e analise a apresentação dos dados no gráfico de dispersão.

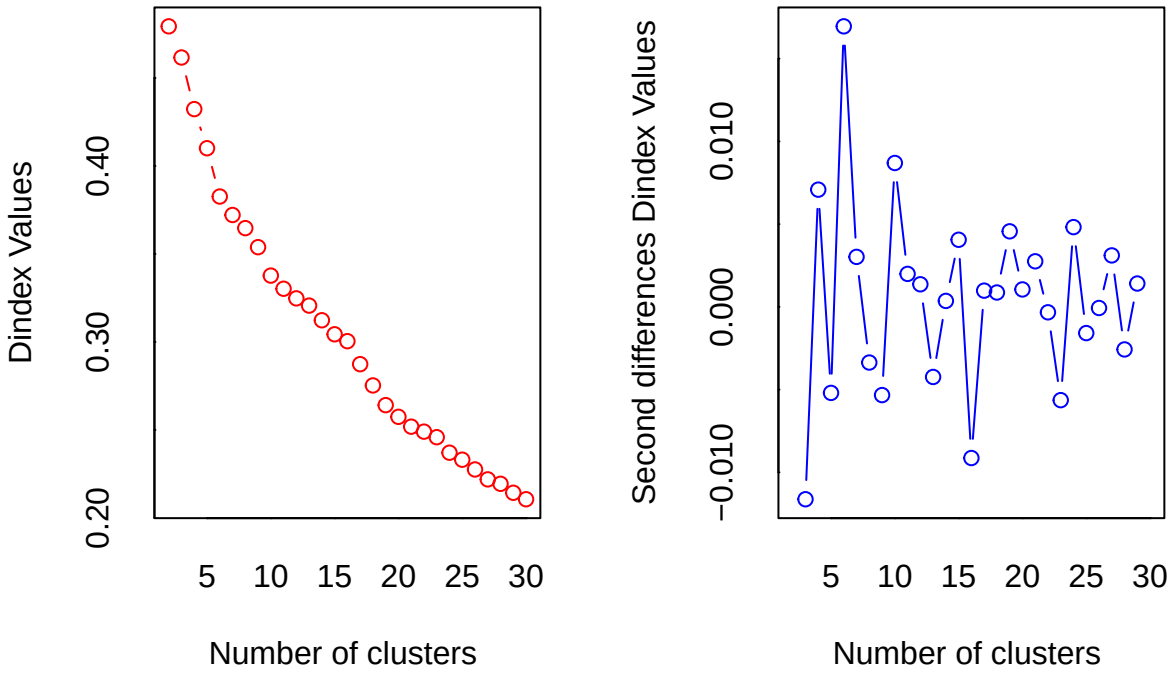
**Resposta:** Primeiramente, fixamos eps para escolha de um valorde minPts adequado. Feito isso, fixamos minPts no melhor valor obtidoe variamos eps para encontrar um valor ótimo Feitas essas escolhas, executamos kNNdistplot e verificamos o valor de cotovelo, em que a rejeição

## Atividade 4 – Comparando os Algoritmos

```
# Algoritmo de avaliacao do melhor K a ser escolhido
# K = 2 escolhido como K otimo
nb <- NbClust ( base_1 , distance ="euclidean",min.nc =2 , max.nc =30 , method ="complete",index ="all")
```



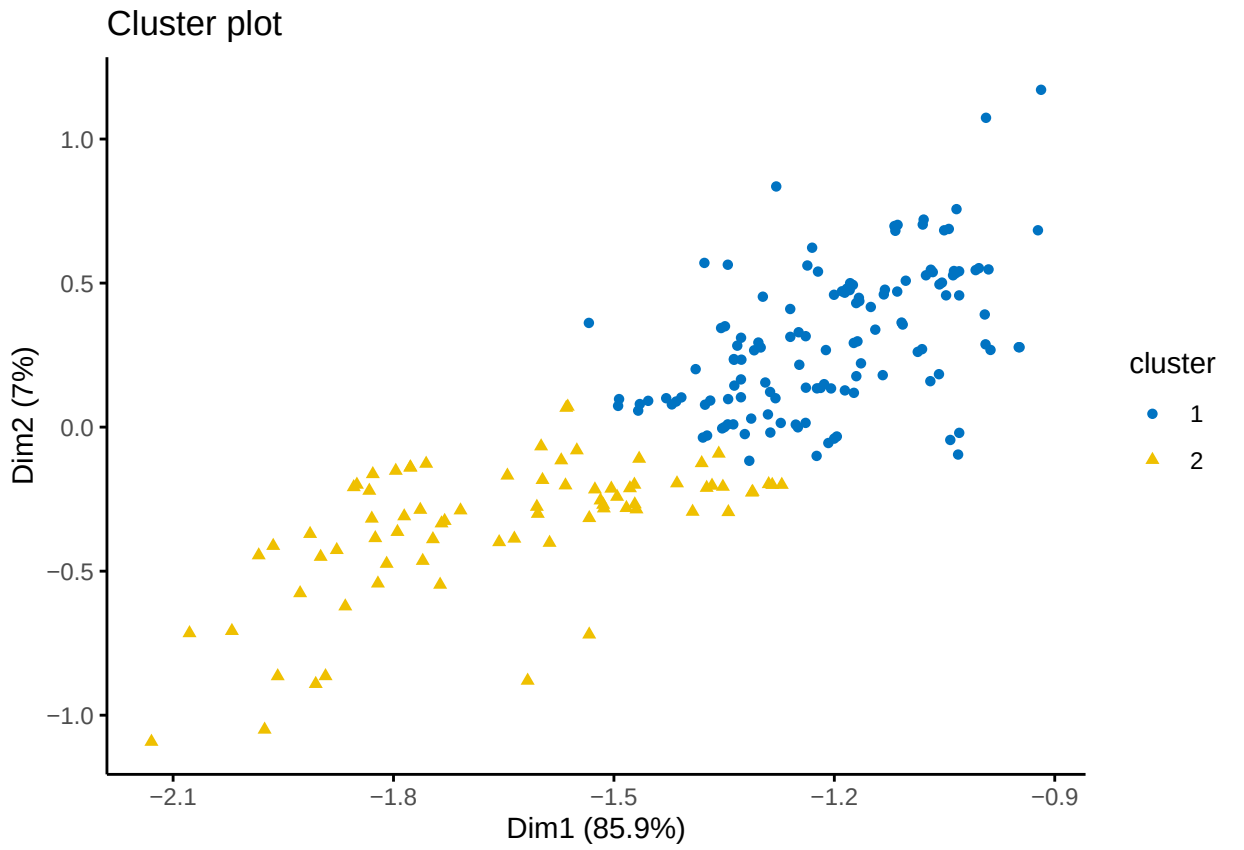

```
## *** : The Hubert index is a graphical method of determining the number of clusters.
##                 In the plot of Hubert index, we seek a significant knee that corresponds to a
##                 significant increase of the value of the measure i.e the significant peak in Hubert
##                 index second differences plot.
##
```




```
## *** : The D index is a graphical method of determining the number of clusters.
##                 In the plot of D index, we seek a significant knee (the significant peak in Dindex
##                 second differences plot) that corresponds to a significant increase of the value of
##                 the measure.
##
## *******************************************************************
## * Among all indices:
## * 7 proposed 2 as the best number of clusters
## * 3 proposed 3 as the best number of clusters
## * 3 proposed 4 as the best number of clusters
## * 2 proposed 6 as the best number of clusters
## * 1 proposed 24 as the best number of clusters
## * 1 proposed 29 as the best number of clusters
## * 6 proposed 30 as the best number of clusters
##
##                     ***** Conclusion *****
##
## * According to the majority rule, the best number of clusters is  2
##
##
## *******************************************************************
```

```r
# Aplicando o k-means com o k escolhido
km <- eclust(base_1, "kmeans", k=2,nstart=25, graph=FALSE)
```

```
# Construindo um gráfico de dispersão
fviz_cluster(km, data=base_1, stand=FALSE,
             ellipse=FALSE, show.clust.cent=FALSE,
             geom="point", palette="jco",
             ggtheme=theme_classic())
```




```
# Teste dos labels vs clusters
base_kmeans_test <- kmeans(base_1 , 2, nstart = 25)
test_df <- as.data.frame.matrix(table(base_kmeans_test$cluster, df_normalized$V1));test_df
```

```
##   -2 -1  0  1  2  3
## 1  3 14 28  8  7 16
## 2  0  8 39 46 25 11
```

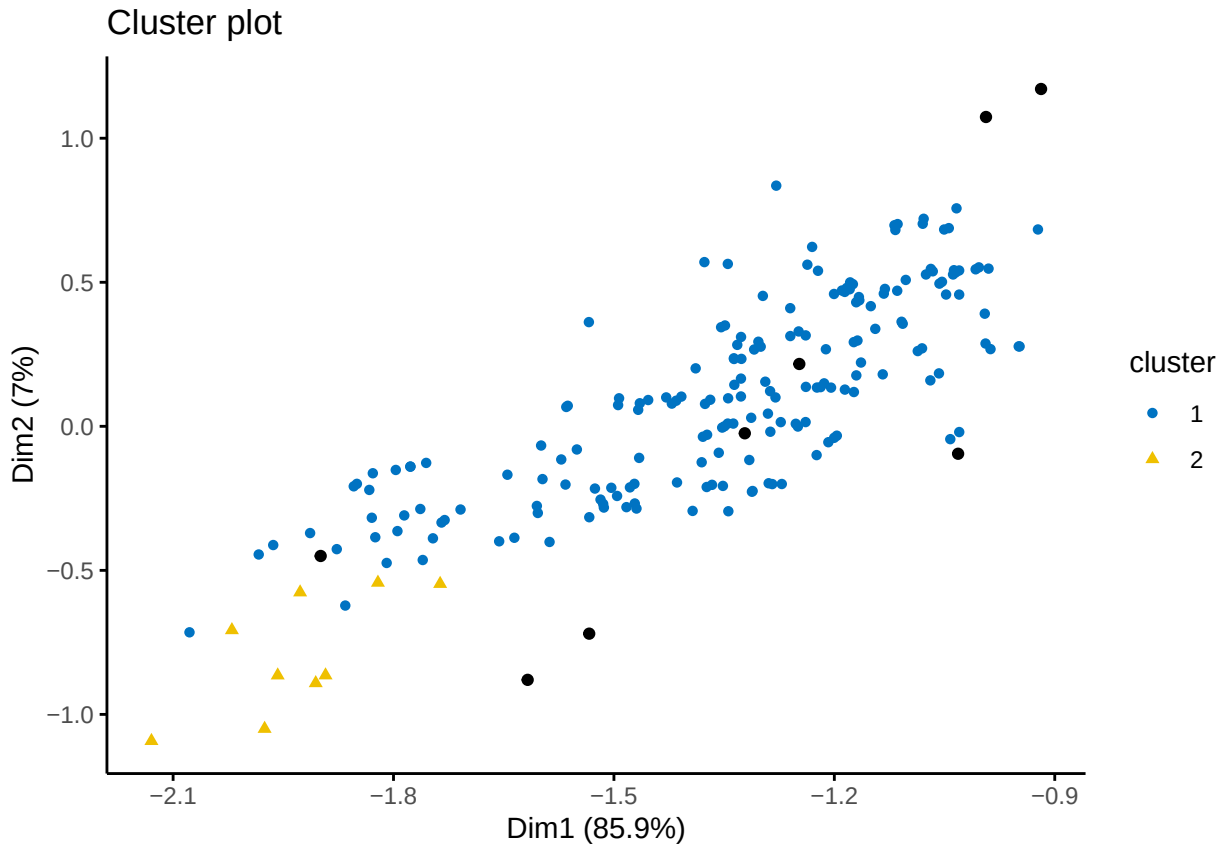
```
test_df2 <- as.data.frame.matrix(table(base_kmeans_test$cluster, df_normalized$V3));test_df2
```

```
##   alfa-romero audi bmw chevrolet dodge honda isuzu jaguar mazda mercedes-benz
## 1           0    6   8         0     1     0     0      3     1             8
## 2           3    1   0         3     8    13     4      0    16             0
##   mercury mitsubishi nissan peugot plymouth porsche renault saab subaru toyota
## 1       1          3      6     11        1       5       0    5      0      5
## 2       0         10     12      0        6       0       2    1     12     27
##   volkswagen volvo
## 1          1    11
## 2         11     0
```

```
# Aplicando o DBscan com os parâmetros escolhidos
 db <- dbscan :: dbscan ( base_1 , eps =0.5 , minPts =4)
 print(db)
```

```
## DBSCAN clustering for 205 objects.
## Parameters: eps = 0.5, minPts = 4
## The clustering contains 2 cluster(s) and 8 noise points.
##
##   0   1   2
##   8 188   9
##
## Available fields: cluster, eps, minPts
```

```
# Construindo um gráfico de dispersão
fviz_cluster(db, data=base_1, stand=FALSE,
             ellipse=FALSE, show.clust.cent=FALSE,
             geom="point", palette="jco",
             ggtheme=theme_classic())
```




```
#analise matriz cruzada

#base_kmeans_test <- kmeans(base_1 , 2, nstart = 20)
analise <- as.data.frame.matrix(table(db$cluster, df_normalized$V1));analise
```

```
##   -2 -1  0  1  2  3
```

```
## 0  0  0  1  2  2  3
## 1  3 18 62 51 30 24
## 2  0  4  4  1  0  0
```

```
analise2 <- as.data.frame.matrix(table(db$cluster, df_normalized$V3));analise2
```

```
##   alfa-romero audi bmw chevrolet dodge honda isuzu jaguar mazda mercedes-benz
## 0           1    0   0         1     0     1     0      1     0             1
## 1           2    7   8         2     9    12     4      0    17             0
## 2           0    0   0         0     0     0     0      2     0             7
##   mercury mitsubishi nissan peugot plymouth porsche renault saab subaru toyota
## 0       0          0      0      0        0       1       0    1      0      0
## 1       1         13     18     11        7       4       2    5     12     32
## 2       0          0      0      0        0       0       0    0      0      0
##   volkswagen volvo
## 0          1     0
## 1         11    11
## 2          0     0
```

Com base nas atividades anteriores, faça uma conclusão dos seus experimentos respondendo às seguintes perguntas:

a) Qual dos métodos apresentou melhores resultados? Justifique.

b) Quantos agrupamentos foram obtidos?

c) Analisando o campo `symboling` e o grupo designado para cada amostra, os agrupamentos conseguiram separar os níveis de risco?

d) Analisando o campo `make` que contém as marcas dos carros, os agrupamentos conseguiram separar as marcas?

**Respostas:** a) K-means pois apesar do menor número de agrupamentos, foi possível interpretar leve separação, ainda que grosseira, em symboling positivos e negativos, no caso de DBScan, praticamente todos os pontos eram agrupados em um único grupo independentemente do número de partiçoes.

b) Dois agrupamentos.

c) Para o K-Means o algoritmo não conseguiu separar de forma adequada os grupos pois para o nível de risco igual a 0, ele se distribuiu igualmente entre os dois clusters. Já para o DB Scan, a maior parte dos carros, independente do nível de risco, foi classificado em apenas um cluster.

d) Como utilizamos K = 2 pudemos observar que o algoritmo de K-Means separou aproximadamente os carros de maior valor (Audi, BMW, Mercedez, Porsche e Volvo) das marcas mais populares, já para o DBScan quase todos os carros foram agrupados em um único grupo.