INF0613 – Aprendizado de Máquina Não Supervisionado

Trabalho 3 - Técnicas de Agrupamento

Kaleb Roncatti de Souza Nelson Gomes Brasil Junior

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:
library(dplyr)
library(corrplot)
library(caret)
library(factoextra)
library(dbscan)
library (cluster)
library(NbClust)
# Configure ambiente de trabalho na mesma pasta
# onde colocou a base de dados:
# setwd("")
set.seed(30)
```

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).

```r
# Leitura da base
cars <- read.csv("imports-85.data", header = FALSE)
summary(cars)
```

```
##        V1              V2                 V3                 V4
##  Min.   :-2.000   Length:205         Length:205         Length:205
##  1st Qu.: 0.000   Class :character   Class :character   Class :character
##  Median : 1.000   Mode  :character   Mode  :character   Mode  :character
##  Mean   : 0.834
##  3rd Qu.: 2.000
##  Max.   : 3.000
##       V5                 V6                 V7                 V8
##  Length:205         Length:205         Length:205         Length:205
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##       V9                 V10              V11            V12             V13
##  Length:205         Min.   : 86.6   Min.   :141   Min.   :60.3   Min.   :47.8
##  Class :character   1st Qu.: 94.5   1st Qu.:166   1st Qu.:64.1   1st Qu.:52.0
##  Mode  :character   Median : 97.0   Median :173   Median :65.5   Median :54.1
##                     Mean   : 98.8   Mean   :174   Mean   :65.9   Mean   :53.7
##                     3rd Qu.:102.4   3rd Qu.:183   3rd Qu.:66.9   3rd Qu.:55.5
##                     Max.   :120.9   Max.   :208   Max.   :72.3   Max.   :59.8
##       V14            V15                V16                 V17
##  Min.   :1488   Length:205         Length:205         Min.   : 61
##  1st Qu.:2145   Class :character   Class :character   1st Qu.: 97
##  Median :2414   Mode  :character   Mode  :character   Median :120
##  Mean   :2556                                         Mean   :127
##  3rd Qu.:2935                                         3rd Qu.:141
##  Max.   :4066                                         Max.   :326
##      V18                V19                V20                 V21
##  Length:205         Length:205         Length:205         Min.   : 7.0
##  Class :character   Class :character   Class :character   1st Qu.: 8.6
##  Mode  :character   Mode  :character   Mode  :character   Median : 9.0
##                                                           Mean   :10.1
##                                                           3rd Qu.: 9.4
##                                                           Max.   :23.0
##      V22                V23                V24             V25
##  Length:205         Length:205         Min.   :13.0   Min.   :16.0
##  Class :character   Class :character   1st Qu.:19.0   1st Qu.:25.0
##  Mode  :character   Mode  :character   Median :24.0   Median :30.0
##                                        Mean   :25.2   Mean   :30.8
##                                        3rd Qu.:30.0   3rd Qu.:34.0
##                                        Max.   :49.0   Max.   :54.0
##      V26
##  Length:205
##  Class :character
##  Mode  :character
##
##
##
```

```r
# Tratamento de dados faltantes

# Visualizando elementos básicos
dim(cars)
```

```
## [1] 205  26
```

```r
head(cars, 5)
```

```
##   V1  V2          V3  V4  V5   V6          V7  V8    V9  V10 V11  V12  V13  V14
## 1  3   ? alfa-romero gas std  two convertible rwd front 88.6 169 64.1 48.8 2548
## 2  3   ? alfa-romero gas std  two convertible rwd front 88.6 169 64.1 48.8 2548
## 3  1   ? alfa-romero gas std  two   hatchback rwd front 94.5 171 65.5 52.4 2823
## 4  2 164        audi gas std four       sedan fwd front 99.8 177 66.2 54.3 2337
## 5  2 164        audi gas std four       sedan 4wd front 99.4 177 66.4 54.3 2824
##    V15  V16 V17  V18  V19  V20 V21 V22  V23 V24 V25   V26
## 1 dohc four 130 mpfi 3.47 2.68   9 111 5000  21  27 13495
## 2 dohc four 130 mpfi 3.47 2.68   9 111 5000  21  27 16500
## 3 ohcv  six 152 mpfi 2.68 3.47   9 154 5000  19  26 16500
## 4  ohc four 109 mpfi 3.19 3.40  10 102 5500  24  30 13950
## 5  ohc five 136 mpfi 3.19 3.40   8 115 5500  18  22 17450
```

```r
# Verificando se existem dados faltantes
cars[cars == "?"] <- NA
nas_count <- sapply(cars, function(x) sum(is.na(x))); nas_count
```

```
##  V1  V2  V3  V4  V5  V6  V7  V8  V9 V10 V11 V12 V13 V14 V15 V16 V17 V18 V19 V20
##   0  41   0   0   0   2   0   0   0   0   0   0   0   0   0   0   0   0   4   4
## V21 V22 V23 V24 V25 V26
##   0   2   2   0   0   4
```

```r
# Percebemos que uma das variáveis com dados faltantes é categórica
subset(cars, is.na(cars$V6))
```

```
##    V1   V2    V3     V4    V5   V6    V7  V8    V9  V10 V11  V12  V13  V14 V15
## 28  1  148 dodge    gas turbo <NA> sedan fwd front 93.7 157 63.8 50.6 2191 ohc
## 64  0 <NA> mazda diesel   std <NA> sedan fwd front 98.8 178 66.5 55.5 2443 ohc
##     V16 V17  V18  V19  V20  V21 V22  V23 V24 V25   V26
## 28 four  98 mpfi 3.03 3.39  7.6 102 5500  24  30  8558
## 64 four 122  idi 3.39 3.39 22.7  64 4650  36  42 10795
```

```r
# O conjunto é extremamente baixo, podemos remover os dados sem prejuízo
cars <- cars[-c(28, 64), ]
nrow(subset(cars, is.na(cars$V6)))
```

```
## [1] 0
```

```r
# Percebemos também que algumas features que são numéricas vieram como string
features_numeric_as_str <- c(2, 19, 20, 22, 23, 26)
cars[, features_numeric_as_str] <- sapply(cars[, features_numeric_as_str], as.numeric)
```

```r
# Preenchendo os dados faltantes com a média dos valores que possuímos
cars <- lapply(cars, function(x) ifelse(is.na(x), mean(x, na.rm = TRUE), x))
cars <- as.data.frame(cars)
# Verificando que de fato não temos mais NAs
nas_count <- sapply(cars, function(x) sum(is.na(x))); nas_count
```

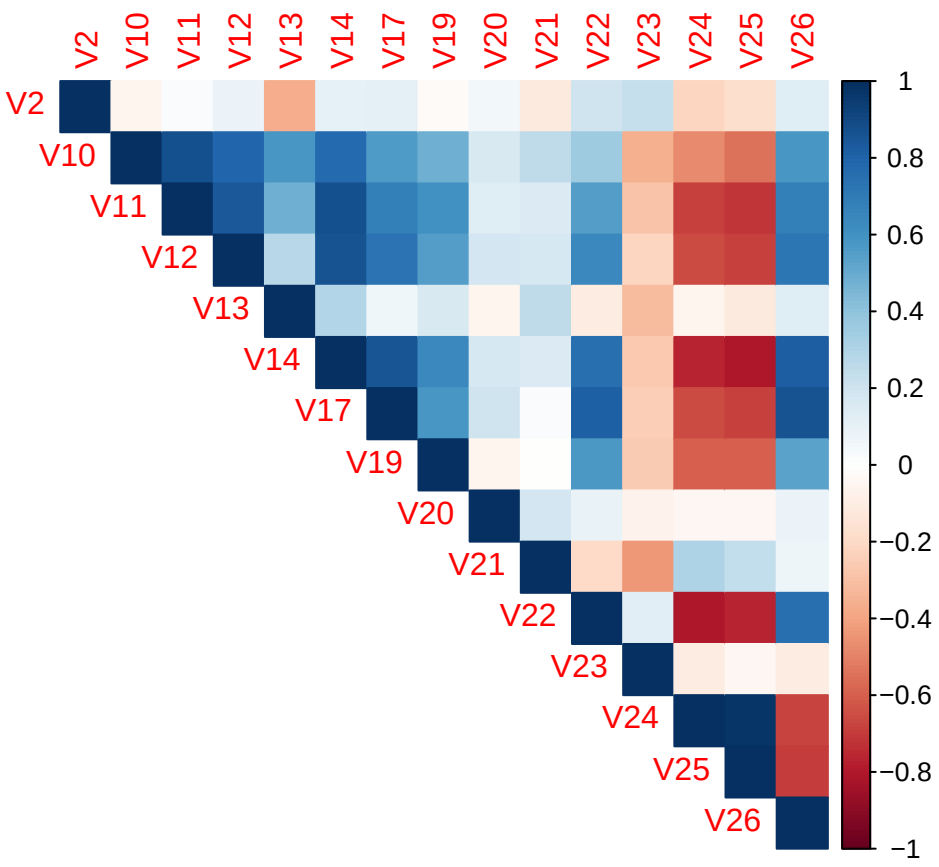
```
##  V1  V2  V3  V4  V5  V6  V7  V8  V9 V10 V11 V12 V13 V14 V15 V16 V17 V18 V19 V20
##   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
## V21 V22 V23 V24 V25 V26
##   0   0   0   0   0   0
```

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.

```r
# Seleção de atributos
head(cars, 5)
```

```
##   V1  V2          V3  V4  V5   V6          V7  V8    V9  V10 V11  V12  V13  V14
## 1  3 122 alfa-romero gas std  two convertible rwd front 88.6 169 64.1 48.8 2548
## 2  3 122 alfa-romero gas std  two convertible rwd front 88.6 169 64.1 48.8 2548
## 3  1 122 alfa-romero gas std  two   hatchback rwd front 94.5 171 65.5 52.4 2823
## 4  2 164        audi gas std four       sedan fwd front 99.8 177 66.2 54.3 2337
## 5  2 164        audi gas std four       sedan 4wd front 99.4 177 66.4 54.3 2824
##    V15  V16 V17  V18  V19  V20 V21 V22  V23 V24 V25   V26
## 1 dohc four 130 mpfi 3.47 2.68   9 111 5000  21  27 13495
## 2 dohc four 130 mpfi 3.47 2.68   9 111 5000  21  27 16500
## 3 ohcv  six 152 mpfi 2.68 3.47   9 154 5000  19  26 16500
## 4  ohc four 109 mpfi 3.19 3.40  10 102 5500  24  30 13950
## 5  ohc five 136 mpfi 3.19 3.40   8 115 5500  18  22 17450
```

```r
cars_target <- cars[, 1]

# Olhando para as features numéricas EXCETO target
num_features <- unlist(lapply(cars, is.numeric), use.names = FALSE)
cars_num <- cars[, num_features]

# Removendo o target
cars_num$V1 <- NULL

# Aplicando a correlação
correlacao <- cor(cars_num[, colnames(cars_num)])
corrplot(correlacao, method = "color", type = "upper")
```




```
# Encontrando elementos com correlação absoluta superior a um valor
correlacao_freq <- as.data.frame(as.table(correlacao))
relevant_corr <- subset(correlacao_freq, (abs(Freq) > 0.87 & abs(Freq) != 1.0))
relevant_corr <- relevant_corr[order(-relevant_corr$Freq),]
relevant_corr
```

```
##     Var1 Var2  Freq
## 194  V25  V24 0.971
## 208  V24  V25 0.971
## 36   V14  V11 0.879
## 78   V11  V14 0.879
## 18   V11  V10 0.875
## 32   V10  V11 0.875
```

```
# Removendo features com alta correlação entre si, trariam possíveis redundâncias para o modelo
cars_num$V25 <- NULL
cars_num$V11 <- NULL

# Usando one-hot encoding para algumas variáveis categóricas
dummy <- dummyVars(~ V4 + V5 + V7 + V8 + V9 + V16, data = cars)
cars_cat <- predict(dummy, newdata = cars)

# Features que serão utilizadas
cars_mut_features <- cbind(cars_num, cars_cat)
dim(cars_mut_features)
```

```
## [1] 203  34
```

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: Começamos todo o tratamento através de uma observação das entradas do conjunto de dados. Percebemos que, ao invés de `NA (Not a Number)` para os dados faltantes, tínhamos elementos com pontos de interrogação `(?)` para tais casos. Fizemos a conversão das interrogações para o tipo `(NA)` e contabilizamos a quantidade de elementos faltantes por feature. Para apenas uma delas (Coluna `(V6)`), tínhamos dados não-númericos, e apenas dois dados sem preenchimento. Neste caso, optamos por remover tais dados.

Percebemos também que, devido às interrogações observadas já no início, algumas colunas com dados do tipo `numeric` vieram do carregamento inicial como `String` e fizemos então a conversão para conseguirmos prosseguir de maneira coerente (colunas `V2, V19, V20, V22, V23 e V26`. Logo após, optamos por preencher todos os dados faltantes das features numéricas com o valor da média de tal feature, técnica explicada e utilizada em aula para não termos que descartar um conjunto bastante relevante de dados. Sendo assim, finalizamos o tratamento inicial da base de dados tendo removido alguns dados faltantes e tendo preenchido outros com a métrica de média, também modificando o tipo de dados para o tipo adequado.

Desta forma, inicializamos o estudo de features pelas features exclusivamente numéricas. Plotamos o grau de correlação dentre todas as features (note que não incluímos o target `V1 - symboling`), observamos visualmente, e decidimos definir um threshold para deixarmos de lado apenas algumas features que apresentam altíssimo grau de correlação entre si. Nesta situação, utilizamos `0.87` de correlação absoluta para as features numéricas como valor de trigger para deixarmos features de lado. Percebemos que o maior grau de correlação disparado foi entre as colunas `V24 (city-mpg) e V25 (highway-mpg)`, métricas que dizem respeito ao consumo de combustível dos veículos na cidade e na estrada, respectivamente. Optamos então por desprezar a feature `V25 (highway-mpg)`. Também com um grau de correlação alto temos as features `V11 (length) e V14 (curb-weight)`, das quais optamos por desprezar a feature `V11 (length)`, dado que a mesma também possue alto grau de correlação com a feature `V10 (wheel-base)`.

No que diz respeito às features categóricas, aplicamos o método de one-hot encoding em algumas features categóricas escolhidas, dentre elas: `V4, V5, V7, V8, V9 e V16`. Infelizmente, não tínhamos acesso à um especialista no ramo para realizarmos um inquérito no que diz respeito às features mais importantes para a aplicação de modelos de clusterização. Desta forma, testamos algumas combinações manualmente (também se baseando na intuição para quais variáveis teriam a maior influência para a previsão de risco de um seguro) e chegamos à conclusão que as features acima se demonstraram mais relevantes para a aplicação dos métodos subsequentes. Uma possível melhoria à nossa análise seria sistematizar a realização de combinações de features e da clusterização, observando-se, por exemplo, a soma das distâncias intra-cluster para `K` fixo conforme varíamos as features utilizadas. Desta maneira, partimos de um conjunto com 205 dados e 25 features e finalizamos com 203 dados e 37 features.

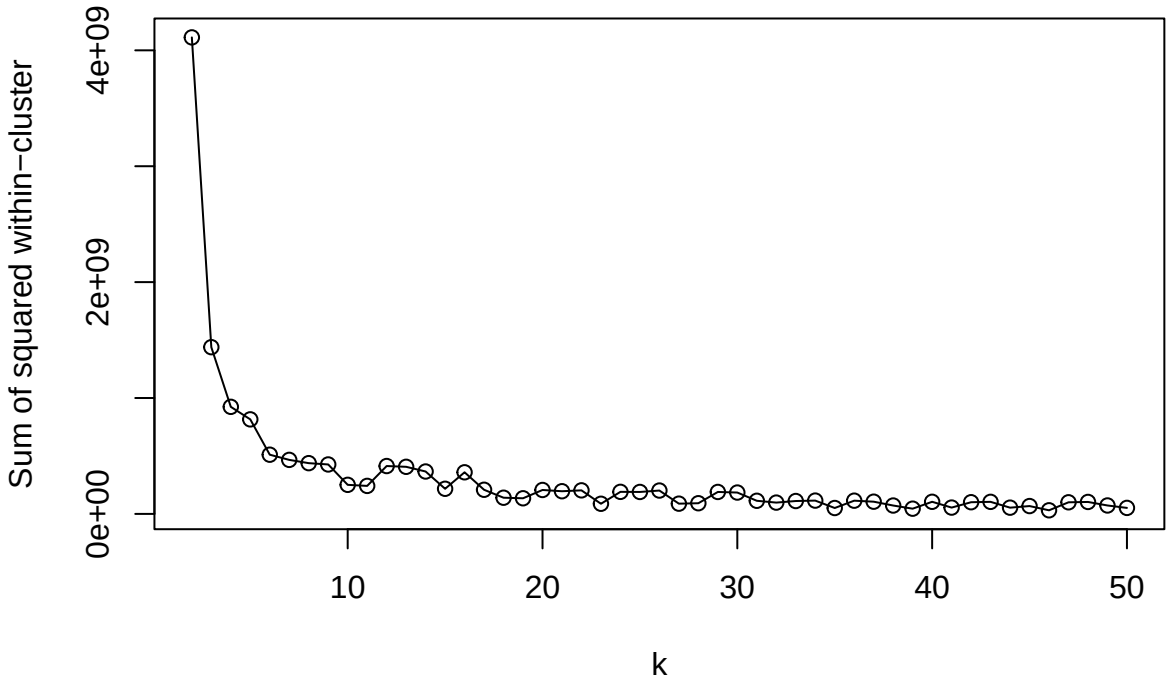
Atividade 2 – Agrupamento com o K-*means*

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.

```
# Construindo um gráfico com as distâncias intra-cluster
df_intra_cluster <- data.frame(matrix(ncol = 2, nrow = 0))
colnames(df_intra_cluster) <- c("k", "Sum of squared within-cluster")

row_num <- 1
for (k in 2:50){
  km <- kmeans(cars_mut_features, centers = k)

  df_intra_cluster[row_num,] <- c(k, sum(km$withinss))
  row_num <- row_num + 1
}

plot(df_intra_cluster) + lines(df_intra_cluster)
```

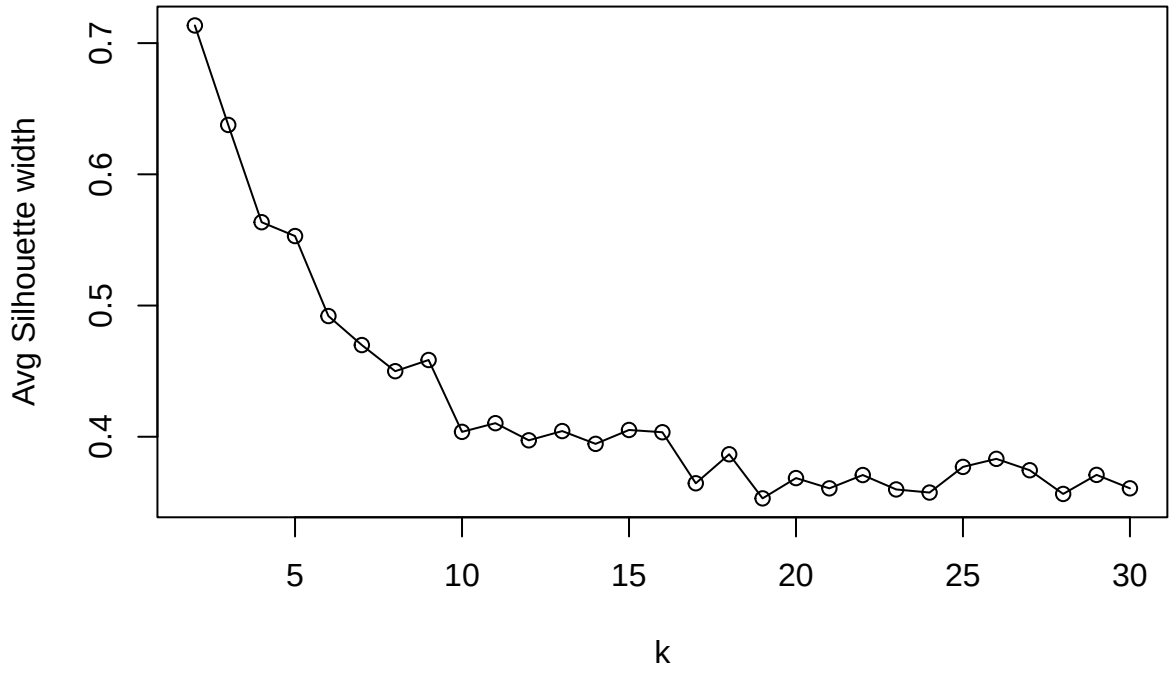



```
## integer(0)
```

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
df_sl <- data.frame(matrix(ncol = 2, nrow = 0))
colnames(df_sl) <- c("k", "Avg Silhouette width")
diss_features <- daisy(cars_mut_features)
```

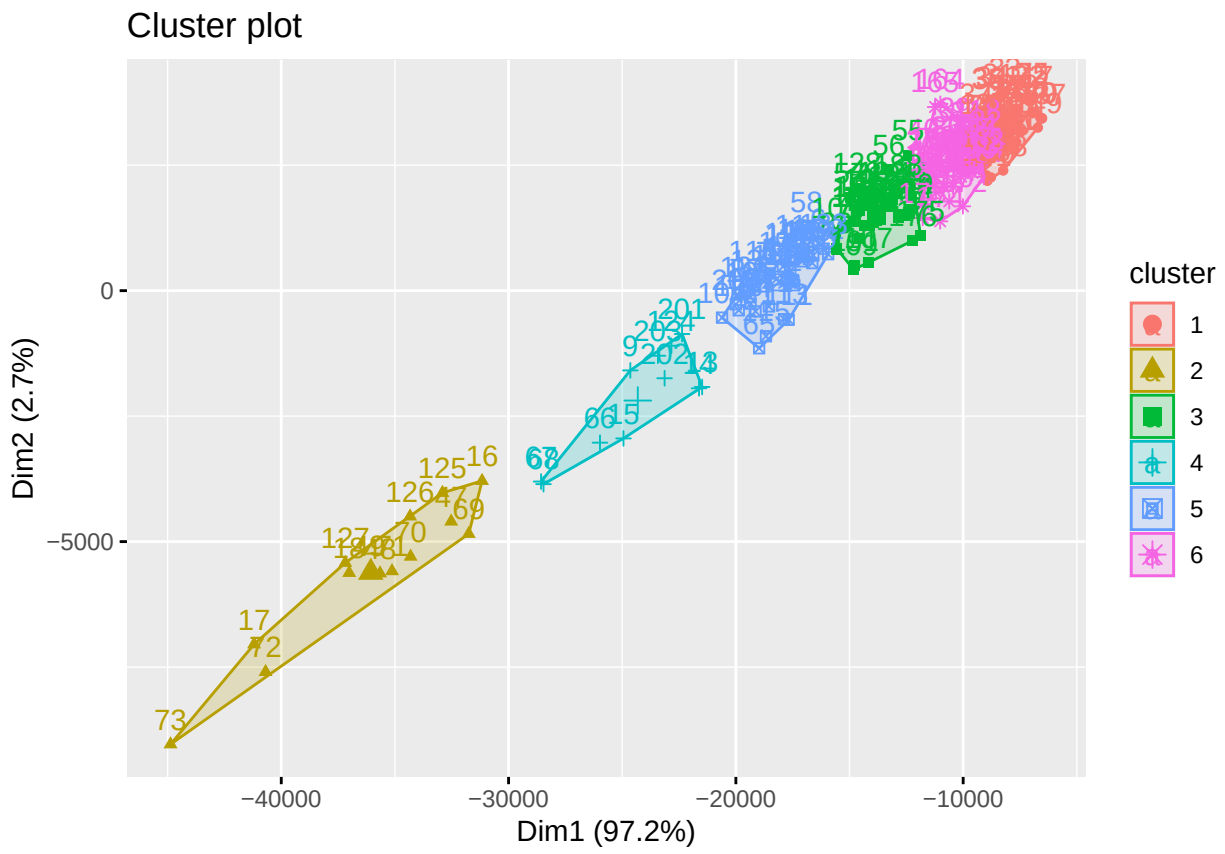
```
row_num <- 1
for (k in 2:30){
  km <- kmeans(cars_mut_features, centers = k)
  sil <-silhouette(km$cl, diss_features)
  # Média da width da silhouetta
  df_sl[row_num,] <- c(k, mean(sil[, 3]))
  row_num <- row_num + 1
}

plot(df_sl) + lines(df_sl)
```



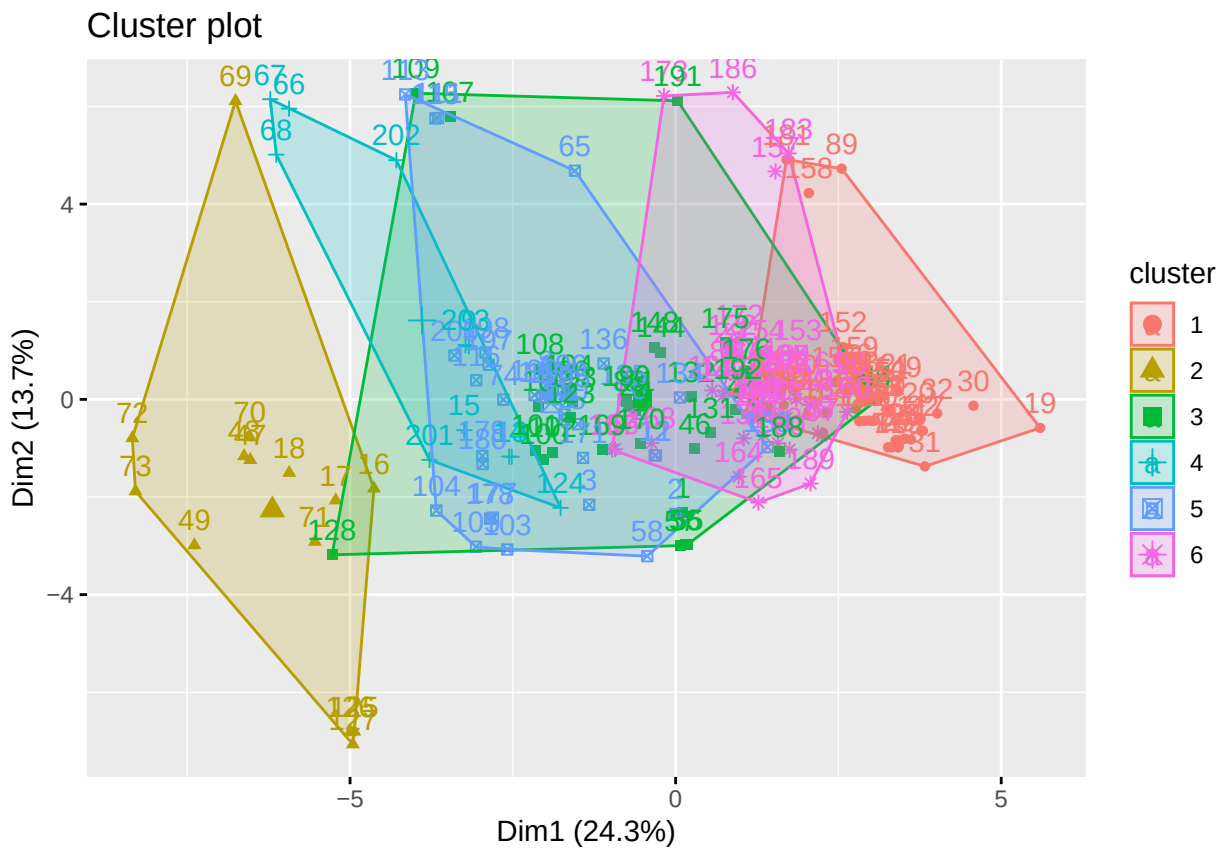

```
## integer(0)
```

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).

```
# Aplicando o k-means com o k escolhido
km_final <- kmeans(cars_mut_features, centers = 6)
# Construindo um gráfico de dispersão (para standardise = TRUE and FALSE)
fviz_cluster(km_final, data = cars_mut_features, stand = FALSE, ellipse = TRUE)
```




```
fviz_cluster(km_final, data = cars_mut_features, stand = TRUE, ellipse = TRUE)
```



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: No que diz respeito aos gráficos gerados:

- A soma dos quadrados da distância intra-cluster nos ajudou a orientar o ponto ótimo de clusters para que possamos prosseguir com os modelos. Nosso objetivo é definir clusters de modo que a variação total intra-cluster seja minimizada. No caso vigente, conseguimos visualizar que, em K = 6 temos um começo de estabilização da curva, ou seja, a partir de tal K, o valor da distância deixa de diminuir e temos o começo de uma assíntota.
- Para o caso da silhoueta, método para o qual tentamos determinar quão bem cada objeto se situa dentro do seu próprio cluster. Um alto valor de width indica uma boa clusterização. Para tal gráfico, percebemos que o K que escolhemos como ótimo dado o método anterior se mostra como o quarto/quinto valor mais elevado para tal width. De toda forma, optamos por focar/valorizar mais o primeiro método porque, mesmo para o K escolhido no método elbow, o valor de silhueta ainda se representa como um valor razoável.
- Apenas completando, para o gráfico de dispersão, observamos que a clusterização se demonstrou relativamente boa quando observamos a PCA não normalizada com detrimento à PCA normalizada. Com a PCA não normalizada, aproximadamente 99.9% da variância do conjunto é explicada pelas duas componentes principais, e para o caso da normalizada, menos de 40% da variância do conjunto é explicada. Visualmente falando, os clusters estão muito melhor separados quando observamos a PCA não

normalizada. Pelo menos a nível visual, a técnica K-means se demonstrou adequada para a separação do conjunto.

Para uma melhora no desempenho do método acima, poderíamos ter feito uma escolha mais rigorosa e sistemática de features que implicariam numa performance mais adequada para os conjuntos.

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_01 <- dbscan::dbscan(cars_mut_features, eps = 500, minPts = 3)
print(db_01)
```

```
## DBSCAN clustering for 203 objects.
## Parameters: eps = 500, minPts = 3
## The clustering contains 7 cluster(s) and 51 noise points.
##
##  0  1  2  3  4  5  6  7
## 51 20 11 11 86  9  5 10
##
## Available fields: cluster, eps, minPts
```

```
# Experimento com valores de eps e minPts
db_02 <- dbscan::dbscan(cars_mut_features, eps = 700, minPts = 3)
print(db_02)
```

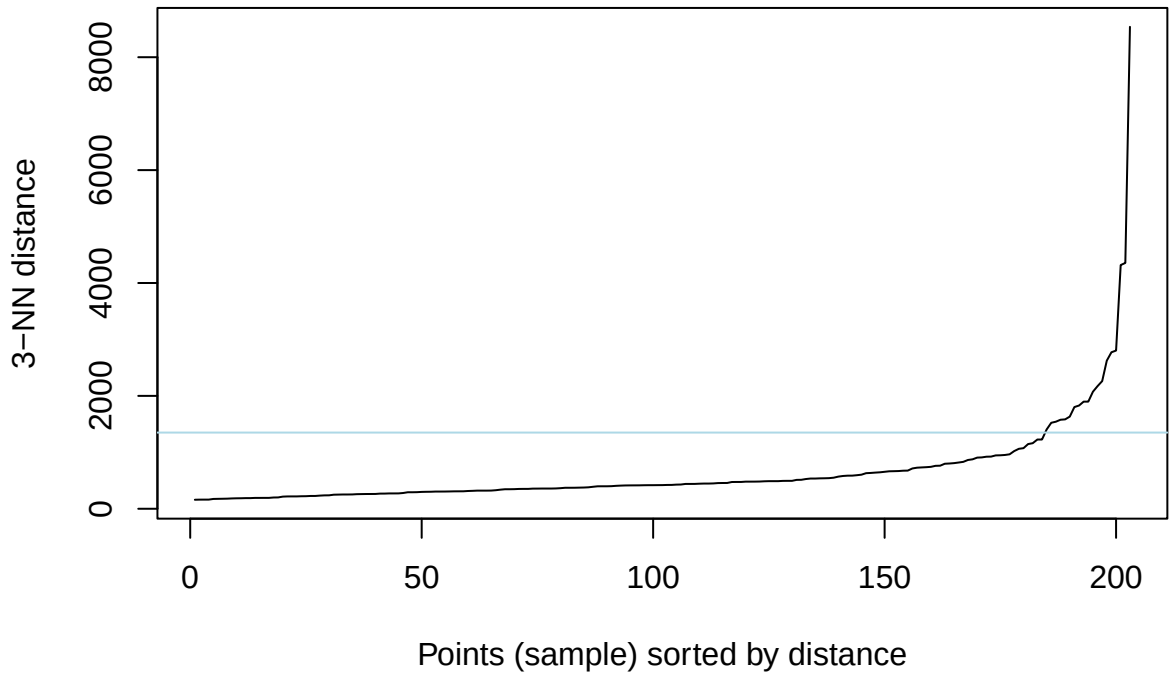
```
## DBSCAN clustering for 203 objects.
## Parameters: eps = 700, minPts = 3
## The clustering contains 4 cluster(s) and 28 noise points.
##
##   0   1   2   3   4
##  28 132  36   3   4
##
## Available fields: cluster, eps, minPts
```

```
# Experimento com valores de eps e minPts
db_03 <- dbscan::dbscan(cars_mut_features, eps = 500, minPts = 4)
print(db_03)
```

```
## DBSCAN clustering for 203 objects.
## Parameters: eps = 500, minPts = 4
## The clustering contains 8 cluster(s) and 57 noise points.
##
##  0  1  2  3  4  5  6  7  8
## 57  9 10 11 83  9  5 10  9
##
## 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(cars_mut_features, k = 3) + abline(h = 1350, col="lightblue")
```



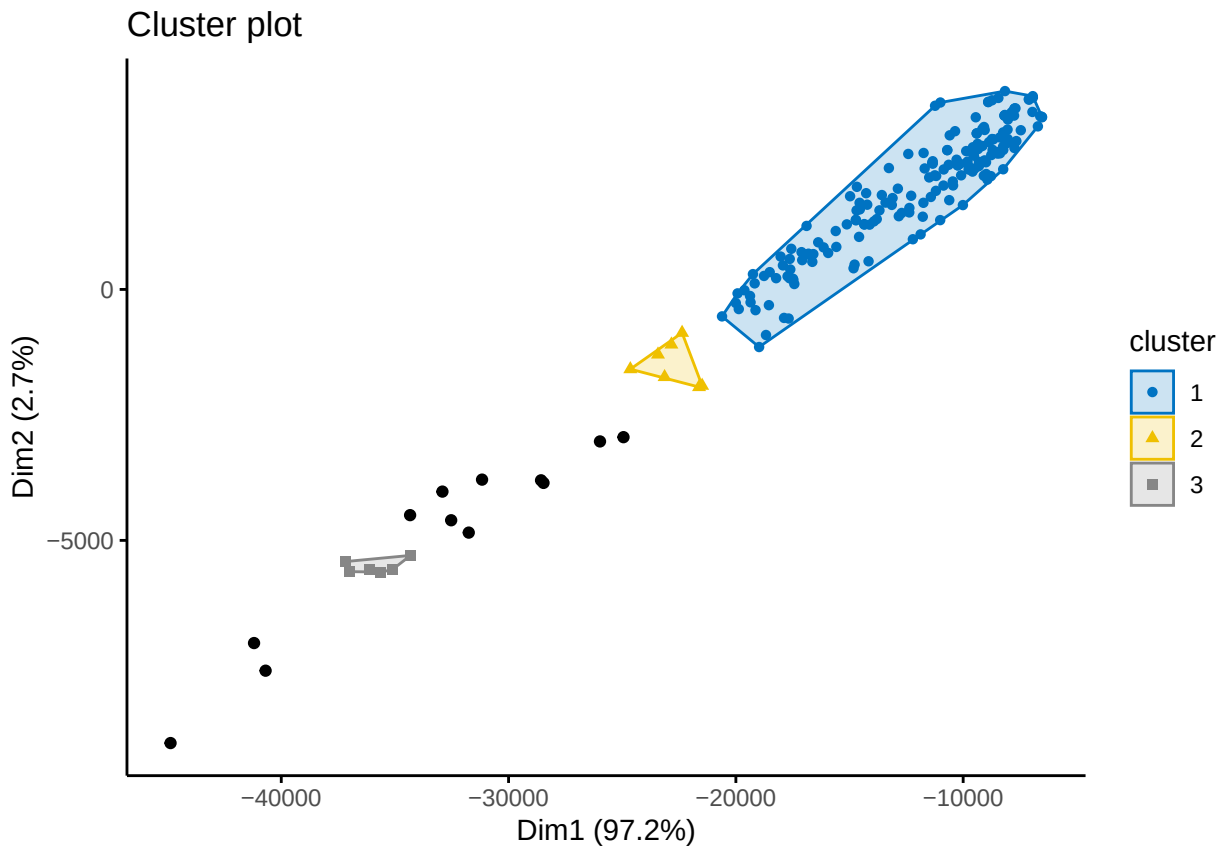

```
## integer(0)
```

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).

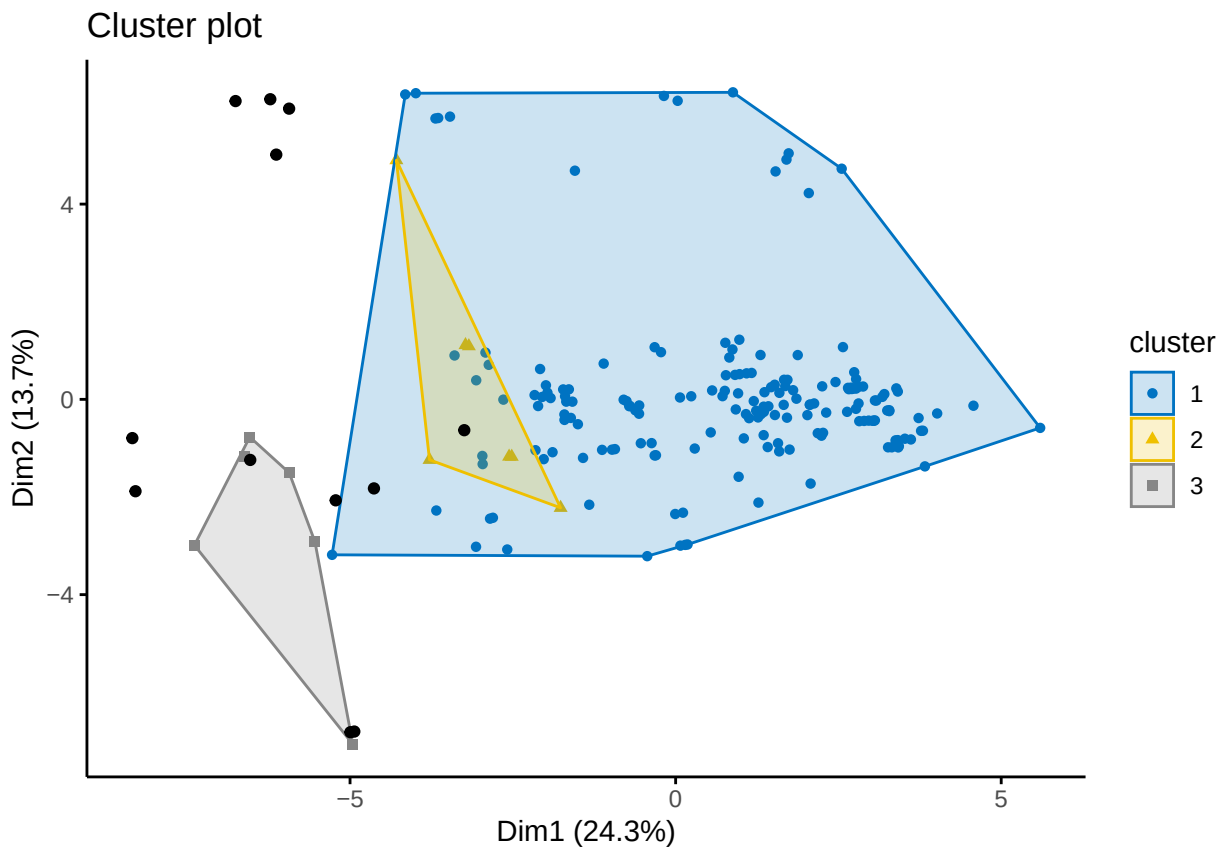
```
# Aplicando o DBscan com os parâmetros escolhidos
db_final <- dbscan::dbscan(cars_mut_features, eps = 1350, minPts = 3)
print(db_final)
```

```
## DBSCAN clustering for 203 objects.
## Parameters: eps = 1350, minPts = 3
## The clustering contains 3 cluster(s) and 12 noise points.
##
##   0   1   2   3
##  12 178   7   6
##
## Available fields: cluster, eps, minPts
```

```
# Construindo um gráfico de dispersão com stardadize = TRUE and FALSE
fviz_cluster(db_final , data = cars_mut_features, stand = FALSE,
             ellipse = TRUE, show.clust.cent = FALSE,
             geom = "point", palette = "jco",
             ggtheme = theme_classic())
```




```
fviz_cluster(db_final , data = cars_mut_features, stand = TRUE,
             ellipse = TRUE, show.clust.cent = FALSE,
             geom = "point", palette = "jco",
             ggtheme = theme_classic())
```



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: Para a escolha dos parâmetros, começando testando valores próximos dos sugeridos em aula para `MinPts` começando entre 4 e 5, e percebemos que valores entre 3 e 4 estavam gerando números de clusters razoavelmente próximos do valor correto que esperamos (7 labels). Além do mais, nosso dataset possui um valor de aproximadamente 200/300 amostras, e o parâmetro `MinPts` representa o mínimo de pontos necessários para gerar um cluster. Se este valor é muito elevado, não teremos o número de clusters o suficiente para o problema em questão.

Assim sendo, fixamos tal valor e plotamos o gráfico `kNNdistplot` para encontrarmos o valor ideal de `eps` a nível teórico. Tal gráfico nos mostra um ponto de inflexão (cotovelo) para o qual a distância do k-ésimo vizinho começa a aumentar, ponto para o qual o valor de eps é ótimo.

Para o gráfico de dispersão, geramos dois gráficos com `standardize = FALSE e TRUE` assim como fizemos para o K-means. Nesta técnica, percebemos que o encontro de clusters não se mostrou satisfatório para nenhum dos conjuntos, mesmo com a PCA não normalizada que, a priori, separou bem o conjunto de dados em duas dimensões.

Atividade 4 – Comparando os Algoritmos

```
# Observando a classificação e o target correto
unclass(table(km_final$cluster, cars_target))
```

```
##    cars_target
##     -2 -1  0  1  2  3
##   1  0  0 13 31 10  0
##   2  0  2  7  1  0  4
##   3  1  3 16  2  5 10
##   4  0  5  3  2  0  1
##   5  2  6 10  5  6  9
##   6  0  6 17 12 11  3
```

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) O método que apresentou melhor resultado dentre os métodos aplicados foi o método `K-means` clustering, dado que o mesmo chegou num valor ideal de clusters próximo ao necessário para explicar o valor de `symboling` apresentado pelo enunciado. No entanto, pelo que observamos do dataset fornecido, o enunciado está equivocado no que diz respeito aos valores de `symboling`: pelo que observamos, o valor mais baixo é de -2 ao invés de -3, ou seja, 6 labels ao invés de 7 como foi apresentado pelo exercício. Desta maneira, vemos que o K-means para K = 6 como K ideal se mostrou como o valor exato de labels para previsão do conjunto. Ademais, é possível concluir visualmente que existem clusters bem definidos (PCA não normalizada) para K-means enquanto a visualização para o DBSCAN se mostra bem ruim, demostrando claramente que a técnica não foi capaz de separar os dados. Conceitualmente falando, a técnica DBSCAN funciona bem quando temos ruído e quando os grupos que precisam ser gerados possuem diferentes formatos e tamanho. Similarmente, tal técnica não costuma funcionar bem quando temos densidades variáveis e alta dimensionalidade. Na nossa situação vigente:

- Temos pouco ruído (ponto que prejudica DBSCAN)
- O formato dos dados (ao menos na decomposição PCA) se mostra aproximadamente o mesmo (ponto que prejudica DBSCAN)
- Temos densidades variáveis (ponto que prejudica DBSCAN)
- Temos alta dimensionalidade (ponto que prejudica DBSCAN)

Desta maneira, todos os pontos evidenciados acima podem ter causado uma performance ruim para o método DBSCAN.

b) Foram obtidos 7 agrupamentos na escolha do parâmetro ótimo utilizando-se o `elbow method`.

c) Analisando-se o campo `symboling`, e comparando-se os resultados da clusterização com o label obtido, observa-se que:

- Para symboling = -2, de um total de 3 amostras, 2 amostras foram classificadas no `cluster 5` e 1 amostra
- Para symboling = -1, de um total de 22 amostras, os `clusters 5 e 6` se demonstraram como os maiores agrupadores
- Para symboling = 0, de um total de 66 amostras, os `clusters 3 e 6` se demonstraram como os maiores agrupadores
- Para symboling = 1, de um total de 53 amostras, o `cluster 1` se demonstrou como o maior agrupador
- Para symboling = 2, de um total de 32 amostras, os `clusters 1 e 6` se demonstraram como os maiores agrupadores
- Para symboling = 3, de um total de 27 amostras, os `clusters 3 e 5` se demonstraram como os maiores agrupadores

Desta forma, mesmo que a clusterização visual utilizando PCA e nossas features se mostre como adequada, os atributos utilizados não parecem se demonstrar suficientes para a previsão de um valor de risco de seguro, dado que, mesmo que bem separados, temos os dados separados em diferentes clusters para um valor de K fixado.

d)