Solução Lista 01

Gustavo Gonzalez do Nascimento,
gonzalez.gustavo@aluno.ufabc.edu.br
Bruno Henrique Batista Pinto,
batista.henrique@aluno.ufabc.edu.br

Exercício 1:

A) Classificação de água potável e não potável
Podemos usar um modelo de classificação para identificar se a água é boa para consumo ou não. Basta coletar amostras de água de diferentes fontes (rios, torneiras, poços).
E medimos características, como:

- pH (nível de acidez).
- Presença de bactérias.
- Metais pesados (exemplo: chumbo, mercúrio).

Com essas informações, o modelo classifica novas amostras como "Potável" ou "Não Potável".

- **Entrada (X)**:
    - pH,
    - Bactérias
    - Metais pesados
- **Saída (Y)**: **"Potável"** ou **"Não Potável"**.

- -

B) Probabilidade de recuperação de dívida
O objetivo é prever a chance de um cliente pagar sua dívida após entrar em inadimplência, essa previsão é expressa em valores percentuais.

- **Entrada (X)**:
    - Tempo de atraso
    - Renda do cliente
    - Valor da dívida
- **Saída (Y)**: Probabilidade de pagamento (valor entre 0 e 1).

- -

C) Segmentação de alunos em uma universidade
Universidades podem usar aprendizado de máquina para agrupar alunos em diferentes categorias com base em seus perfis socioeconômicos e demográficos.

O modelo separa os alunos em grupos como:

- Ampla Concorrência
- PCD (Pessoa com Deficiência)
- Cota para escola pública
- Cota racial, entre outros
- **Entrada (X)**:
    - Tipo de escola cursada (pública, privada).
    - Renda familiar.
    - Idade.
    - Raça.
- **Saída (Clusters)**: Os diferentes segmentos de alunos, como “Ampla Concorrência", "PCD", “Cota de escola pública”, “Cota Racial”, etc.

Exercício 2:

A maldição da dimensionalidade acontece quando aumentamos o número de variáveis de um modelo, tornando o aprendizado de máquina menos eficiente. Com um maior número de dimensões, os dados ficam mais dispersos, dificultando a identificação de padrões e fazendo com que medidas de distância, essenciais para algoritmos como kNN, percam o sentido. Além disso, aumenta o risco de overfitting, pois o modelo pode acabar aprendendo padrões do conjunto de treino, sem generalizar bem para novos dados.

Exercício 3:

Código:

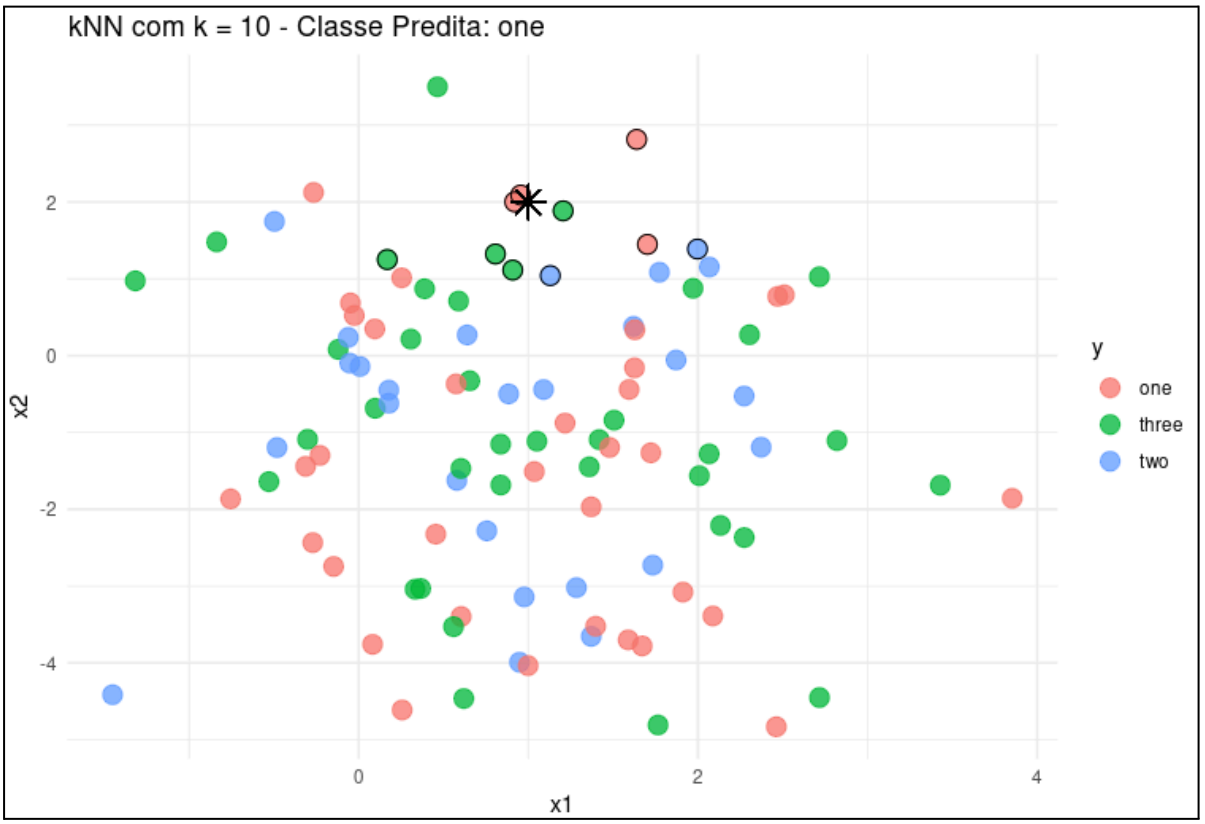
```
library(tidyverse)

knn_classificador <- function(k, x, D) {
  D2 <- D %>%
    mutate(dist = (x[1] - x_1)^2 + (x[2] - x_2)^2) %>%
    arrange(dist) %>%
    head(k)

  predicao <- D2 %>%
    count(y) %>%
    arrange(desc(n)) %>%
    slice(1) %>%
    pull(y)

  return(list(predicao = predicao, neighbors = D2))
}

set.seed(2025)
```

```
D <- tibble(
  x_1 = rnorm(100, 1, 1),
  x_2 = rnorm(100, -1, 2),
  y = factor(sample(c("one", "two", "three"), 100, replace = TRUE))
)

x_test <- c(1, 2)
k <- 10

resultado <- knn_classificador(k, x_test, D)
print(resultado$predicao)

ggplot(D, aes(x = x_1, y = x_2, color = y)) +
  geom_point(size = 4, alpha = 0.75) +
  geom_point(data = resultado$neighbors, aes(x = x_1, y = x_2), color = "black", shape = 1, size = 4) +
  geom_point(aes(x = x_test[1], y = x_test[2]), color = "black", shape = 8, size = 5) +
  labs(title = paste("kNN com k =", k, "- Classe Predita:", resultado$predicao),
       x = "x1", y = "x2") +
  theme_minimal()
```

Output:

```
one
Levels: one three two
```

Gráfico:



Exercício 4:

Código:

```
library(tidyverse)

knn_classificador <- function(k, x, D) {
  D2 <- D %>%
    mutate(dist = (x[1] - x_1)^2 + (x[2] - x_2)^2) %>%
    arrange(dist) %>%
    head(k)

  predicao <- D2 %>%
    count(y) %>%
    arrange(desc(n)) %>%
    slice(1) %>%
    pull(y)

  return(predicao)
}

data("iris")
iris <- as_tibble(iris) %>%
  select(Petal.Length, Sepal.Length, Species) %>%
  rename(x_1 = Petal.Length, x_2 = Sepal.Length, y = Species)

avaliar_knn <- function(k, dataset) {
  l_iris <- as.list(dataset)

  v_bool <- pmap_lgl(l_iris, function(x_1, x_2, y) {
    predicao <- knn_classificador(k, c(x_1, x_2), dataset)
    return(predicao == y)
  })

  accuracy <- sum(v_bool) / length(v_bool) * 100
  return(accuracy)
}

acuracia_k10 <- avaliar_knn(10, iris)
acuracia_k1 <- avaliar_knn(1, iris)

cat("Acurácia do kNN no dataset:\n")
cat("Para k = 10: ", acuracia_k10, "%\n")
cat("Para k = 1: ", acuracia_k1, "%\n")
```

Output:

```
Acurácia do kNN no dataset:
Para k = 10: 95.33 %
Para k = 1: 99.33 %
```