PRIMEIRA AVALIAÇÃO DE APRENDIZADO ESTATÍSTICO DE MÁQUINA

Miguel Zanchettin de Oliveira*

Maio de 2024

EXERCÍCIO 01 (ML)

Diferencie os seguintes termos e utilize um exemplo.

01.A FUNÇÃO DE PERDA E ERRO

A resposta do exercício está em: Resposta - Exercício 01.A

01.B VALIDAÇÃO E DATA SPLITING

A resposta do exercício está em: Resposta - Exercício 01.B

01.C OVERFITTING E COMPLEXIDADE

A resposta do exercício está em: Resposta - Exercício 01.C

01.D RISCO E RISCO EMPÍRICO

A resposta do exercício está em: Resposta - Exercício 01.D

01.E DIMENSÃO VC E COEF. DE SHATTERING

A resposta do exercício está em: Resposta - Exercício 01.E

EXERCÍCIO 02

Escolha dois métodos de aprendizado de máquina diferentes de SVM ou KNN. Pesquise sobre a Dimensão VC de cada um. Crie um texto explicativo sobre sua pesquisa e vincule com o que foi visto na disciplina.

Qualquer pesquisa relacionada ao tema revelará, de forma rápida, que são pouco frequentes aquelas capazes de concluir algo. Principalmente, que têm forma fechada e simplificada de resultado. Isto porquê, denotar a possibilidade

*Mestrando do Programa de Pós-Graduação em Métodos Numéricos Aplicados à Engenharia - PPGMNE, da Universidade Federal do Paraná - UFPR.

de dimensões VC para quaisquer modelos demonstra-se de relativa dificuldade quando por métodos dedutivos. Embora sejam difíceis, tais teorias não são impossíveis. Para esta resposta, serão expostas as conclusões para redes neuras feedforward e as redes neuras com funções de ativação Piecewise polinomial, dois métodos de classificação com modelos similares, porém distintos para a classificação.

As redes neurais feedfoward, com saídas binárias, possuem dimensão VC $O(w \cdot \log w)$ [2]. Sendo w a quantidade de pesos e O a utilização da Big-O notation, medida de complexidade de um algorítmo. A definição formal realizada pelo autor está em seu Teorema 2.1, citado abaixo.

> Seja $\mathcal{N}$ uma rede neural feedforward arbitrária com w pesos que consiste em portões de limiar linear. Então, $VCdim(\mathcal{N}) = O(w \cdot \log w)$ [3](Tradução pelo autor).

Quanto às redes neuras com funções de ativação Piecewise polinomial, sua dimensão VC é descrita como $wl \log w + wl^2$, sendo w a quantidade de pesos e l, de camadas.

> Suponha que $\mathcal{N}$ é uma rede feed-forward com w pesos, l camadas, e todos os portões não de saída tendo uma função de ativação polinomial em partes fixas com um número fixo de partes. Então, $VCdim(\mathcal{N}) = O(wl \log w + wl^2)$.[1].

Assim, percebe-se que, pelas dimensões VC demonstradas pelos autores, seria possível identificar, dada a dimensão VC, um mínimo global para o risco empírico nas funções, porém, este problema recai sobre uma estrutura ótima de formulação das redes neurais em si e das matrizes de pesos utilizadas. Em função disso, ambos os modelos não têm convergência garantida para o mínimo global, uma vez que, dadas quantidades de w ou l, diferentes mínimos locais são passíveis de aparecimento.

EXERCÍCIO 03 (ML)

Leia artigo RAPER, Simon. Leo Breiman's "two cultures". Significance, v. 17, n. 1, p. 34-37, 2020. Acesso aqui. Argumente relacionando o artigo com os conteúdos de aprendizado de máquina vistos na disciplina.

A resposta do exercício, elaborada à mão, está nos anexos (Resposta - Exercício 03).

EXERCÍCIO 04

Pesquise duas métricas de capacidade preditiva não vistas em sala de aula para o contexto de classificação. Implemente a versão micro das métricas escolhidas. Utilize os dados Wall-Following Robot Navigation e o classificador KNN.

O código utilizado para responder o exercício está em: Código: Exercício 4. Os métodos utilizados foram: Recall e F-Beta Score. Embora o Recall seja bem comum, não foi apresentado em aula, já o F-Beta Score é, de fato, menos convencional.

EXERCÍCIO 05

Escolha um conjunto de dados para classificação binária e de seu interesse. Aplique os métodos de validação holdout, holdout repetido, k-fold, k-fold repetido e leave-one-out. Compare os resultados.

Os resultados estão disponíveis na tabela 1. Para validar um modelo de regressão em uma base de dados binária gerada aleatoriamente na biblioteca Sickit-Learn do Python, foi utilizado o Matthews Correlation Coefficient (MCC), pois costuma ser um dos métodos mais utiilizados em pesquisas de Machine Learning, conforme informado pelo professor em classe.

Metodologia de validação	Reg. Log.
Houldout Simples	0.986681
Houldout Repetido (20 vezes)	0.986681
KFold	0.984071
KFold Repetido (20 vezes)	0.985376
Leave One Out	0.993333

Table 1: Resultados calculados

Os códigos utilizados, por sua vez, estão disponíveis em Código: Exercício 5.

EXERCÍCIO 06 (ML)

Considere $n = 3$ observações, $Y = -1, +1$ e $\mathcal{X} = \mathbf{R}$ e $G = I_{[a,b]} | a < b \in \mathbf{R}$. Calcule o coeficiente de shattering $\mathcal{M}(G, n)$.

A resposta do exercício, elaborada à mão, está nos anexos (Resposta - Exercício 06 e Resposta - Exercício 06 (Continuação)).

EXERCÍCIO 07 (ML)

Mostre matematicamente que o princípio de minimização do risco empírico não é consistente para o coef. de shattering com crescimento exponencial. Interprete o resultado.

A resposta do exercício, elaborada à mão, está nos anexos (Resposta - Exercício 07 e Resposta - Exercício 07 (Continuação)).

EXERCÍCIO 08 (ML)

Justifique o Método de Máquina de Vetores de Suporte com base na Teoria do Aprendizado Estatístico.
A resposta do exercício, elaborada à mão, está nos anexos (Resposta - Exercício 08).

ANEXOS

RESOLUÇÃO À MÃO LIVRE - EXERCICIO 01.A

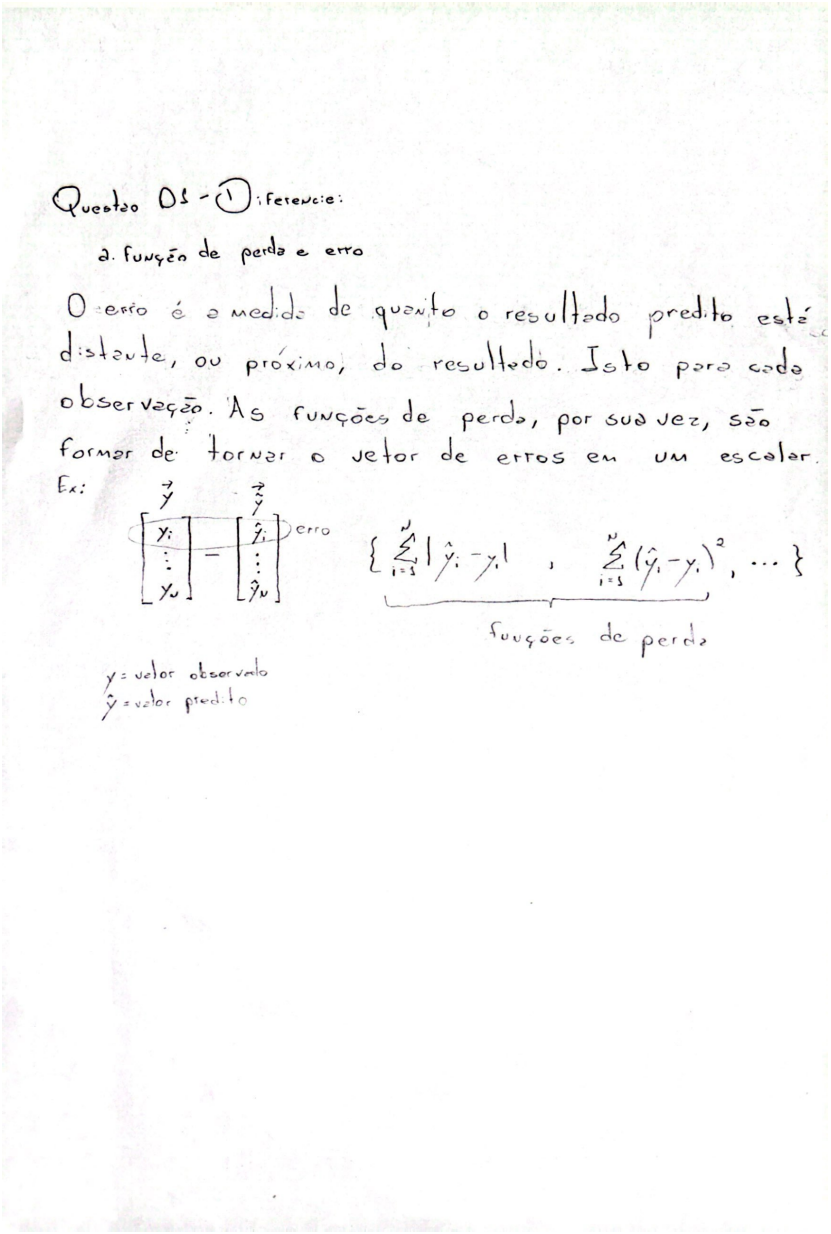


Figure 1: Resposta - Exercício 01.A

RESOLUÇÃO À MÃO LIVRE - EXERCICIO 01.B

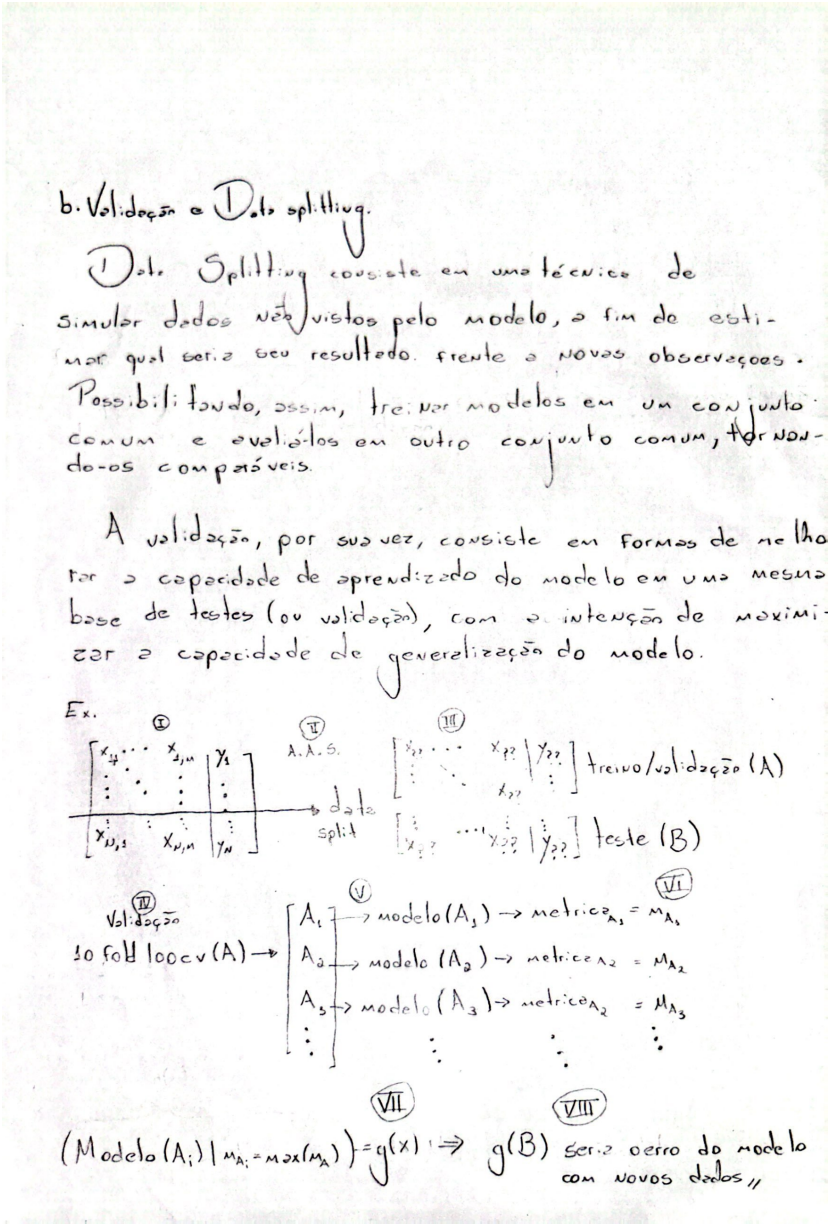


Figure 2: Resposta - Exercício 01.B

RESOLUÇÃO À MÃO LIVRE - EXERCICIO 01.C

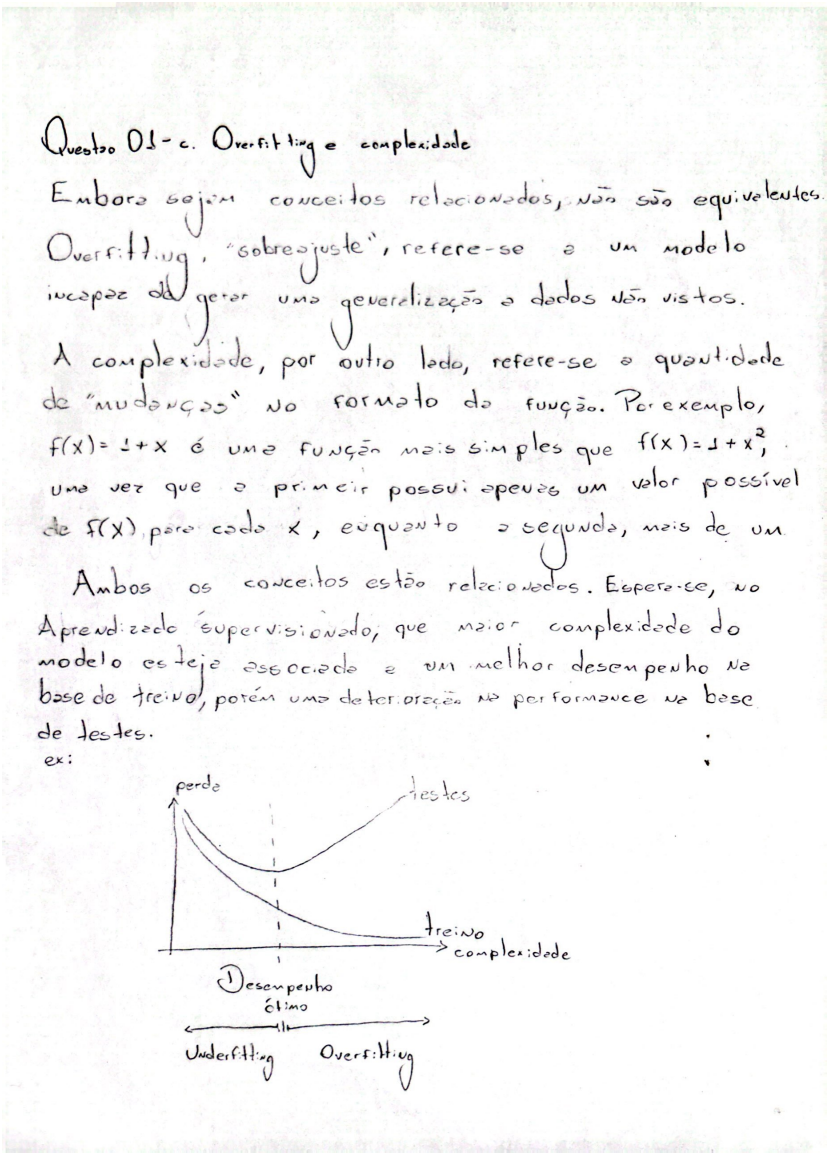
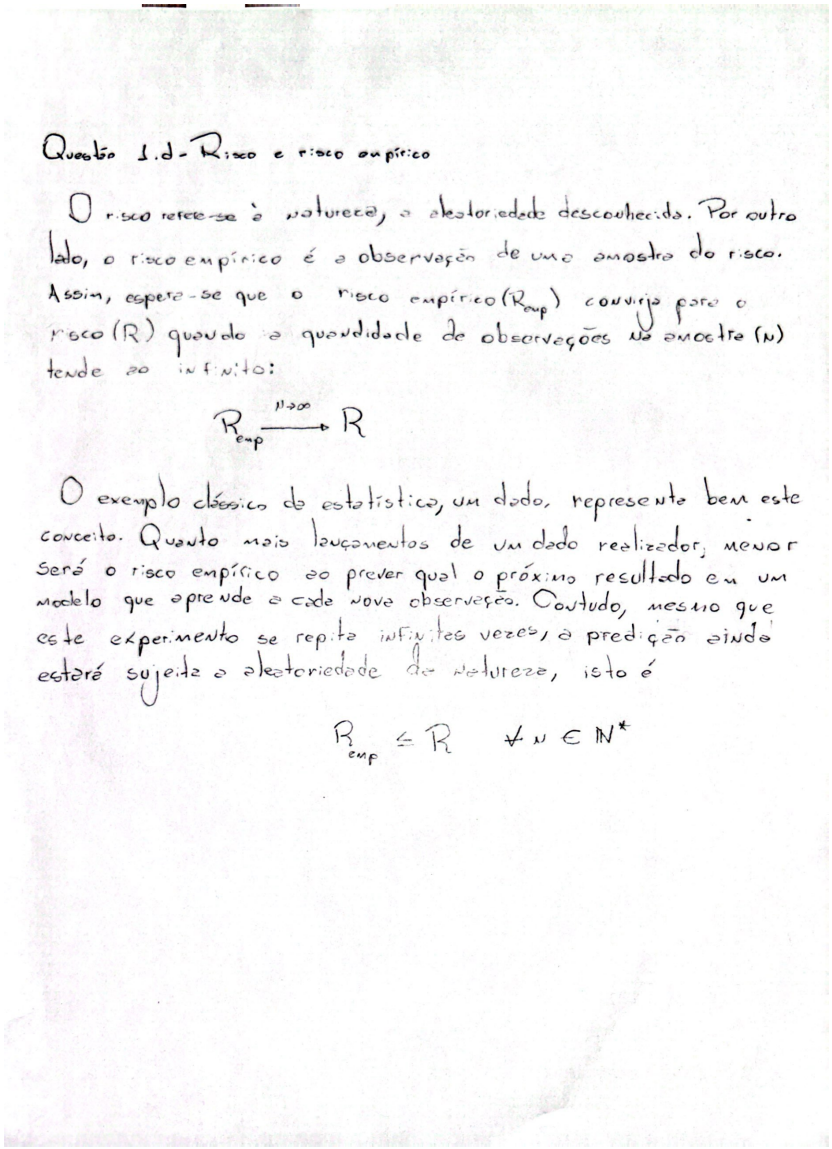


Figure 3: Resposta - Exercício 01.C

RESOLUÇÃO À MÃO LIVRE - EXERCICIO 01.D



Questão 1.d - Risco e risco empírico

O risco refere-se à natureza, a aleatoriedade desconhecida. Por outro lado, o risco empírico é a observação de uma amostra do risco. Assim, espera-se que o risco empírico (R_{emp}) convirja para o risco (R) quando a quantidade de observações na amostra (N) tende ao infinito:

$$R_{emp} \xrightarrow{N \to \infty} R$$

O exemplo clássico de estatística, um dado, representa bem este conceito. Quanto mais lançamentos de um dado realizados, menor será o risco empírico ao prever qual o próximo resultado em um modelo que aprende a cada nova observação. Contudo, mesmo que este experimento se repita infinitas vezes, a predição ainda estará sujeita a aleatoriedade da natureza, isto é

$$R_{emp} \leq R \quad \forall N \in \mathbb{N}^*$$

Figure 4: Resposta - Exercício 01.D

RESOLUÇÃO À MÃO LIVRE - EXERCICIO 01.E

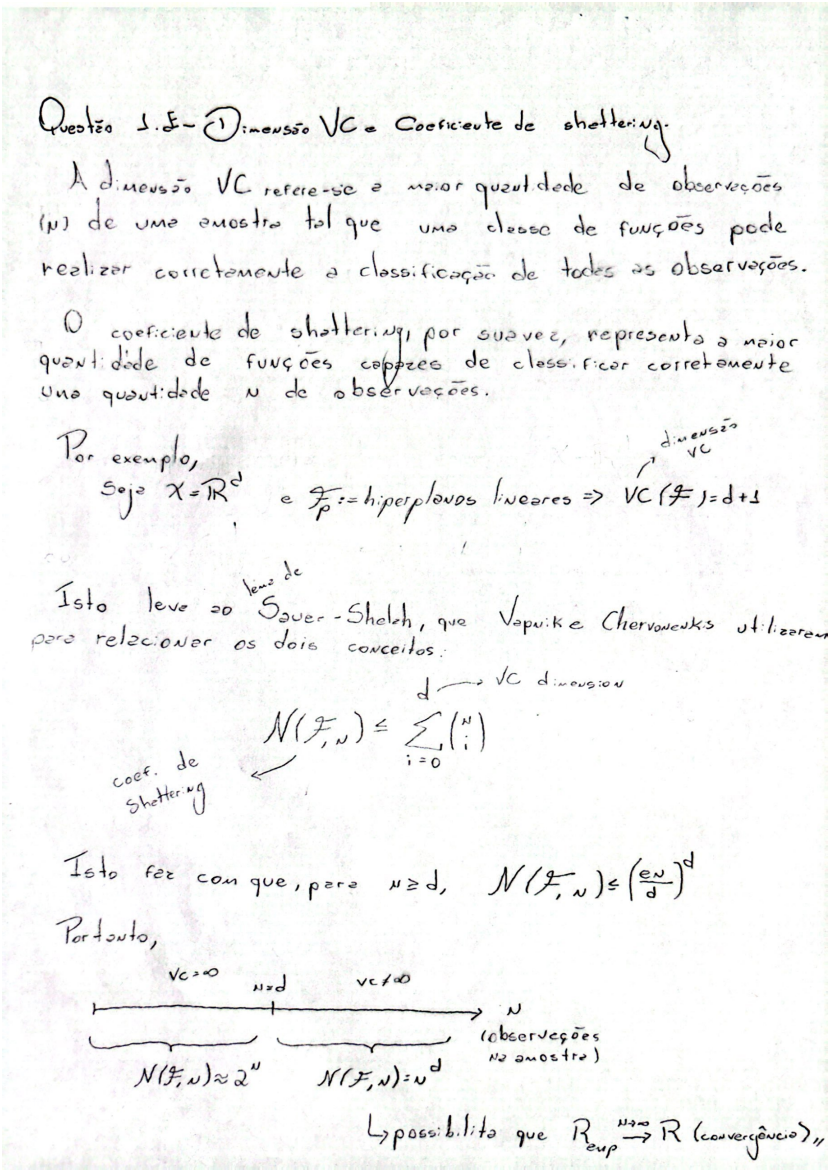
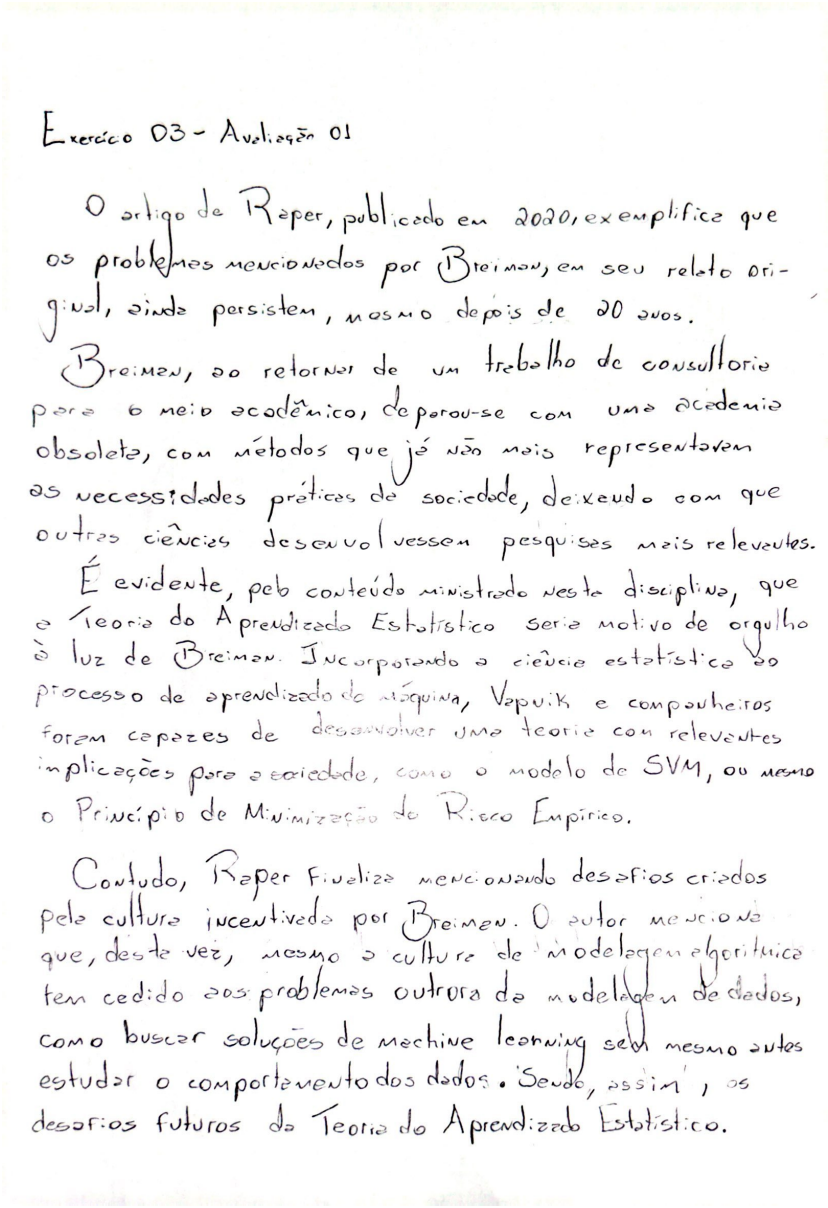


Figure 5: Resposta - Exercício 01.E

RESOLUÇÃO À MÃO LIVRE - EXERCICIO 03



Exercício 03 - Avaliação 01

O artigo de Raper, publicado em 2020, exemplifica que os problemas mencionados por Breiman, em seu relato original, ainda persistem, mesmo depois de 20 anos.

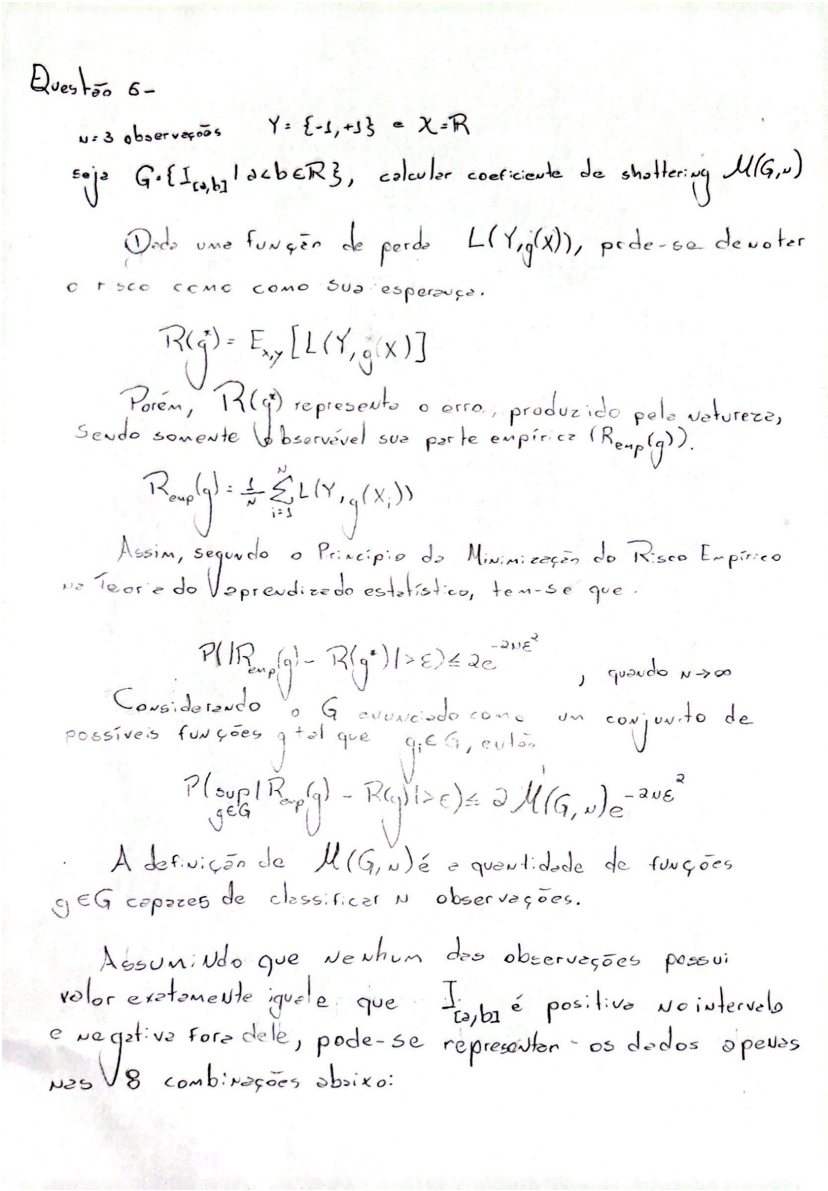
Breiman, ao retornar de um trabalho de consultoria para o meio acadêmico, deparou-se com uma academia obsoleta, com métodos que já não mais representavam as necessidades práticas de sociedade, deixando com que outras ciências desenvolvessem pesquisas mais relevantes.

É evidente, pelo conteúdo ministrado nesta disciplina, que a Teoria do Aprendizado Estatístico seria motivo de orgulho à luz de Breiman. Incorporando a ciência estatística ao processo de aprendizado de máquina, Vapnik e companheiros foram capazes de desenvolver uma teoria com relevantes implicações para a sociedade, como o modelo de SVM, ou mesmo o Princípio de Minimização do Risco Empírico.

Contudo, Raper finaliza mencionando desafios criados pela cultura incentivada por Breiman. O autor menciona que, desta vez, mesmo a cultura de modelagem algorítmica tem cedido aos problemas outrora da modelagem de dados, como buscar soluções de machine learning sem mesmo antes estudar o comportamento dos dados. Sendo, assim, os desafios futuros da Teoria do Aprendizado Estatístico.

Figure 6: Resposta - Exercício 03

RESOLUÇÃO À MÃO LIVRE - EXERCICIO 06



Questão 6 -

$n = 3$ observações $\quad Y = \{-1, +1\}$ e $X = \mathbb{R}$

seja $G = \{I_{[a,b]} \mid a < b \in \mathbb{R}\}$, calcular coeficiente de shattering $\mathcal{M}(G, n)$

① Dado uma função de perda $L(Y, g(X))$, pode-se denotar o risco como como sua esperança.

$$R(g^*) = E_{x,y}[L(Y, g(X))]$$

Porém, $R(g^*)$ representa o erro, produzido pela natureza, sendo somente observável sua parte empírica ($R_{emp}(g)$).

$$R_{emp}(g) = \frac{1}{N} \sum_{i=1}^{N} L(Y, g(X_i))$$

Assim, segundo o Princípio da Minimização do Risco Empírico na Teoria do Aprendizado estatístico, tem-se que:

$$P(|R_{emp}(g) - R(g^*)| > \varepsilon) \leq 2e^{-2N\varepsilon^2}, \text{ quando } N \to \infty$$

Considerando o G anunciado como um conjunto de possíveis funções g tal que $g_i \in G$, então

$$P(\sup_{g \in G} |R_{emp}(g) - R(g)| > \varepsilon) \leq 2\mathcal{M}(G, n)e^{-2N\varepsilon^2}$$

A definição de $\mathcal{M}(G, n)$ é a quantidade de funções $g \in G$ capazes de classificar n observações.

Assumindo que nenhum das observações possui valor exatamente igual que $I_{[a,b]}$ é positivo no intervalo e negativa fora dele, pode-se representar os dados apenas nas 8 combinações abaixo:

Figure 7: Resposta - Exercício 06

RESOLUÇÃO À MÃO LIVRE - EXERCICIO 06 (CONTINUAÇÃO)

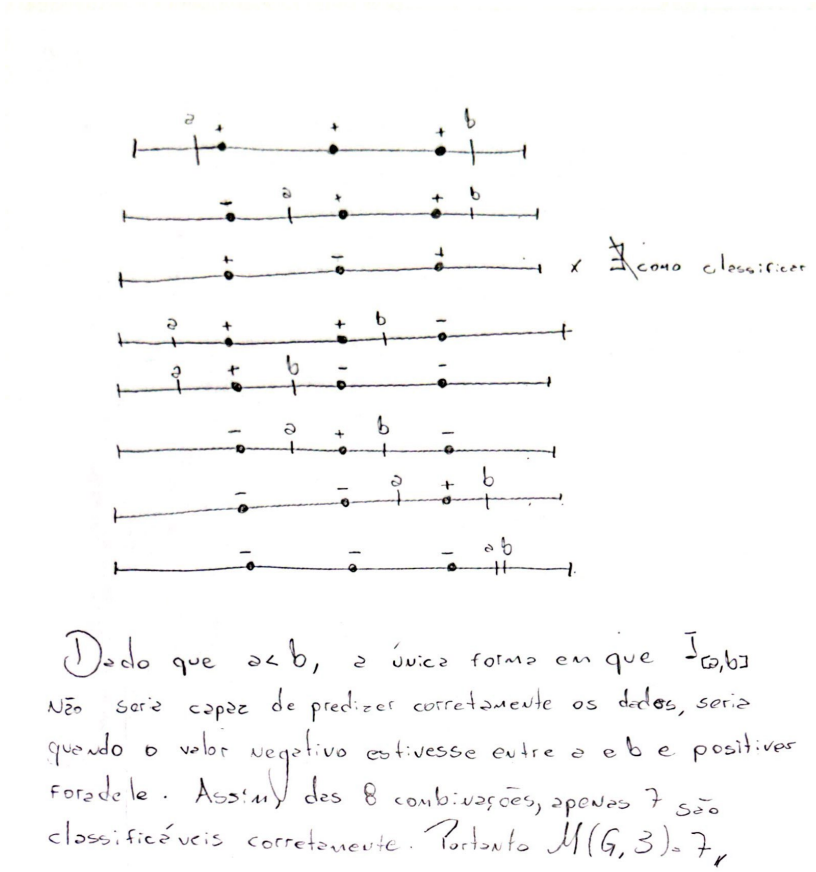
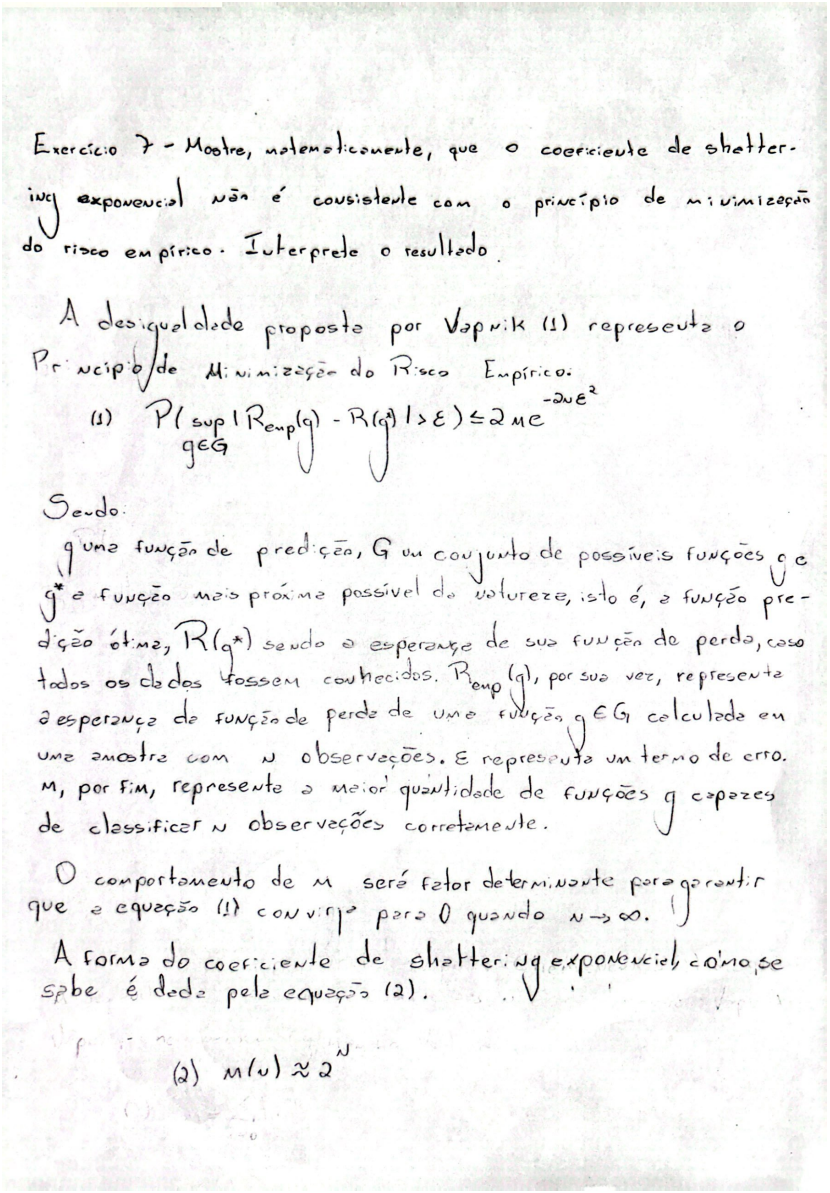


Figure 8: Resposta - Exercício 06 (Continuação)

RESOLUÇÃO À MÃO LIVRE - EXERCICIO 07



Exercício 7 - Mostre, matematicamente, que o coeficiente de shattering exponencial não é consistente com o princípio de minimização do risco empírico. Interprete o resultado.

A desigualdade proposta por Vapnik (1) representa o Princípio de Minimização do Risco Empírico.

(1) $P(\sup_{g \in G} |R_{emp}(g) - R(g)| > \varepsilon) \leq 2me^{-2n\varepsilon^2}$

Sendo:

g uma função de predição, G um conjunto de possíveis funções g e g^* a função mais próxima possível da natureza, isto é, a função predição ótima, $R(g^*)$ sendo a esperança de sua função de perda, caso todos os dados fossem conhecidos. $R_{emp}(g)$, por sua vez, representa a esperança da função de perda de uma função $g \in G$ calculada em uma amostra com n observações. E representa um termo de erro. m, por fim, representa a maior quantidade de funções g capazes de classificar n observações corretamente.

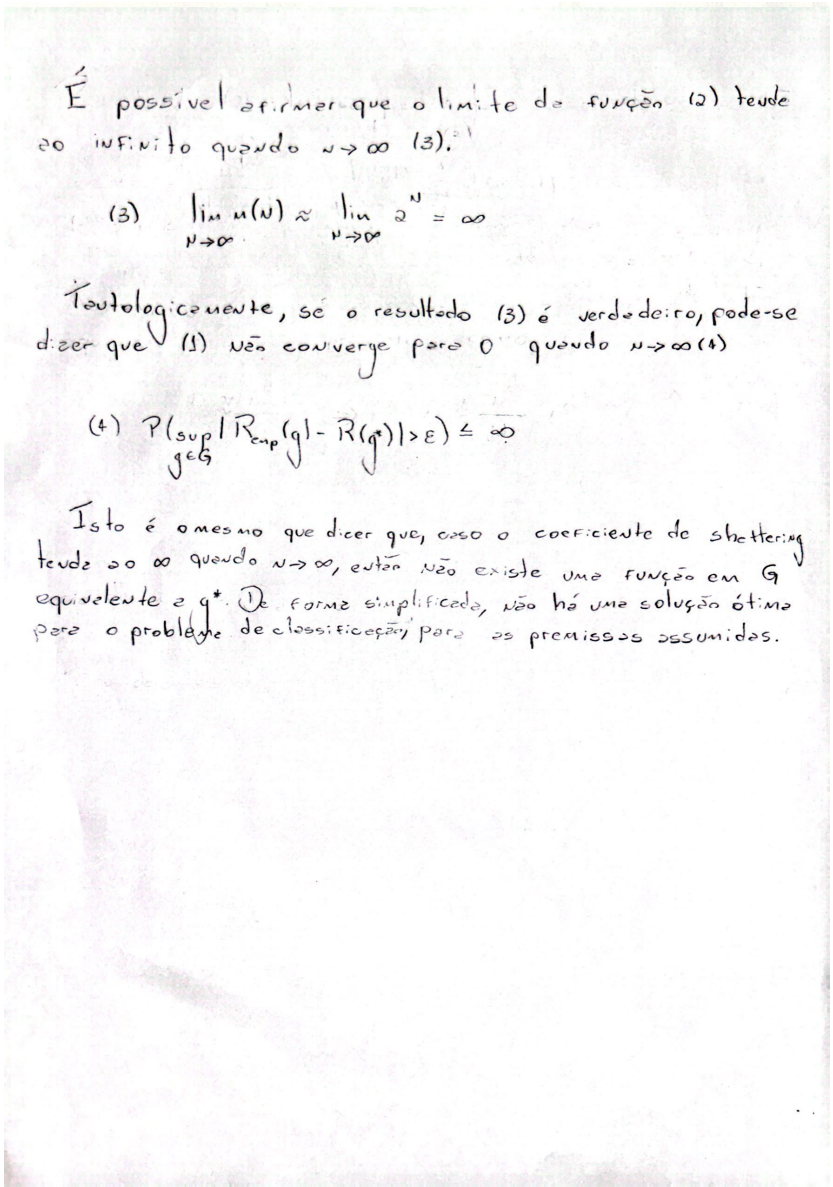
O comportamento de m será fator determinante para garantir que a equação (1) convirja para 0 quando $n \to \infty$.

A forma do coeficiente de shattering exponencial, como se sabe é dada pela equação (2).

(2) $m(v) \approx 2^n$

Figure 9: Resposta - Exercício 07

RESOLUÇÃO À MÃO LIVRE - EXERCICIO 07 (CONTINUAÇÃO)



É possível afirmar que o limite da função (2) tende ao infinito quando $n \to \infty$ (3).

(3) $\lim_{N \to \infty} n(N) \approx \lim_{N \to \infty} 2^N = \infty$

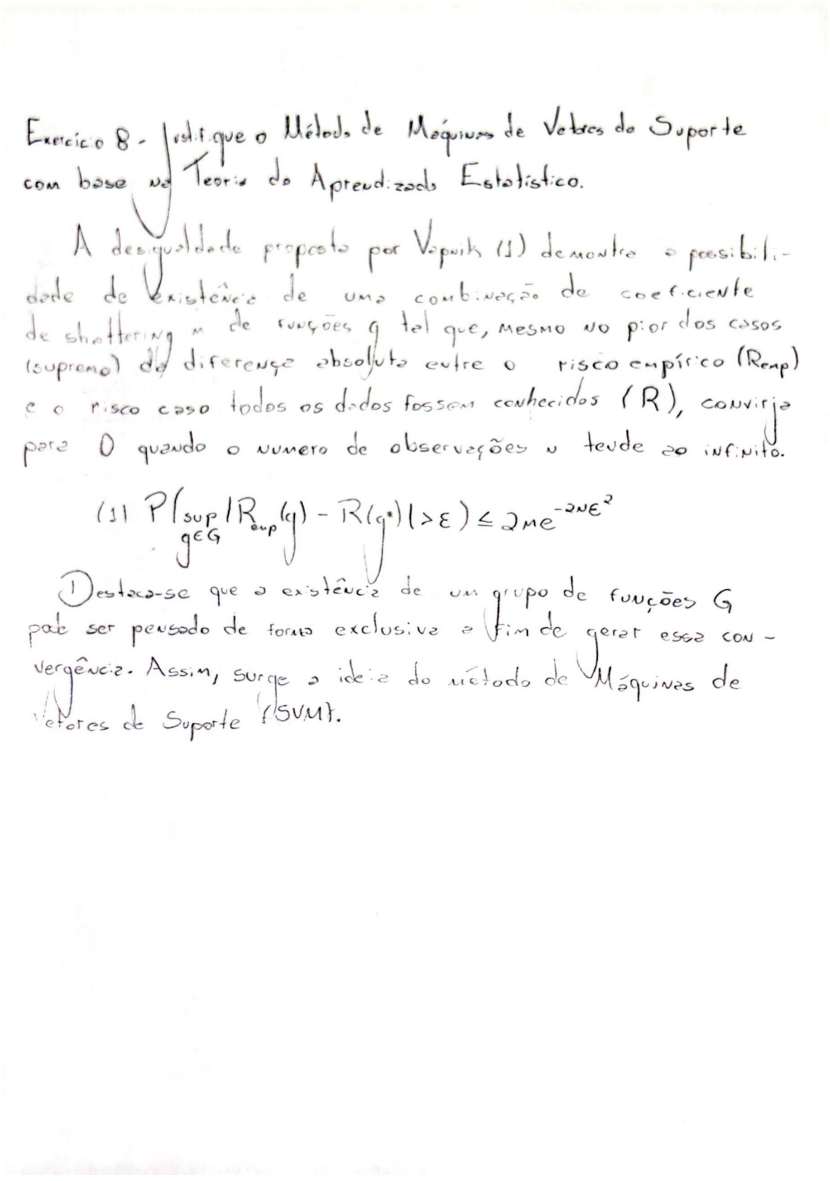
Tautologicamente, se o resultado (3) é verdadeiro, pode-se dizer que (1) não converge para 0 quando $n \to \infty$ (4)

(4) $P(\sup_{g \in G} |R_{emp}(g) - R(g^*)| > \varepsilon) \leq \infty$

Isto é o mesmo que dizer que, caso o coeficiente de shattering tenda ao ∞ quando $n \to \infty$, então não existe uma função em G equivalente a g^*. De forma simplificada, não há uma solução ótima para o problema de classificação para as premissas assumidas.

Figure 10: Resposta - Exercício 07 (Continuação)

RESOLUÇÃO À MÃO LIVRE - EXERCICIO 08



Exercício 8 - Justifique o Método de Máquinas de Vetores de Suporte com base na Teoria do Aprendizado Estatístico.

A desigualdade proposta por Vapnik (1) demonstra a possibilidade de existência de uma combinação de coeficiente de shattering n de funções g tal que, mesmo no pior dos casos (supremo) da diferença absoluta entre o risco empírico (R_{emp}) e o risco caso todos os dados fossem conhecidos (R), convirja para 0 quando o número de observações v tende ao infinito.

$$(1)\ P\left(\sup_{g \in G} |R_{emp}(g) - R(g^*)| > \varepsilon\right) \leq 2ne^{-2v\varepsilon^2}$$

Destaca-se que a existência de um grupo de funções G pode ser pensado de forma exclusiva a fim de gerar essa convergência. Assim, surge a ideia do método de Máquinas de Vetores de Suporte (SVM).

Figure 11: Resposta - Exercício 08

CÓDIGOS

CÓDIGO: EXERCÍCIO 4

```
#-------------------------------------------------------------
# Libraries
#

import numpy as np
import pandas as pd

from sklearn.metrics import recall_score
from sklearn.metrics import fbeta_score

from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier

from rich import print as pprint

#-------------------------------------------------------------
# Data Handling
#

# Extract
url = ('https://raw.githubusercontent.com/'
       'andersonara/datasets/master/'
       'wall-robot-navigation.csv')

df = pd.read_csv(url, delimiter=';')

# Transform data into numpy matrices
X = df[['X1', 'X2']].to_numpy()
y = np.ravel(df[['Y']])

# Data split
X_train, X_test, \
y_train, y_test = train_test_split(X, y,
                                   train_size=0.7,
                                   shuffle=True,
                                   random_state=2002)
```

```
#------------------------------------------------------------
# Model evaluation
#

def get_best_knn_by_metric(metric,
                           maxiter: int = 10,
                           **kwargs):

    def train_knn(k):
        model = KNeighborsClassifier(n_neighbors=k)
        model.fit(X_train, y_train)

        y_pred = model.predict(X_test)

        error = metric(y_test, y_pred, **kwargs)

        return (model, round(error, 5))

    def k_fit(maxiter):
        metrics = {}

        for k in range(1, maxiter + 1):
            _, metrics[k] = train_knn(k)

        best_k = max(metrics, key=metrics.get)

        return best_k

    def get_result(metric, maxiter):
        r = {}
        r['k'] = k_fit(maxiter)
        r['model'], r['metric_value'] = train_knn(r['k'])
        r['metric'] = metric

        return r

    return get_result(metric, maxiter)
```

```
#-------------------------------------------------------------
# Show results
#

metricas = {
    'Recall Micro': dict(metric=recall_score, average='micro'),
    'Recall Macro': dict(metric=recall_score, average='macro'),
    'F-Beta Micro': dict(metric=fbeta_score, average='micro',
        beta=2),
    'F-Beta Macro': dict(metric=fbeta_score, average='macro',
        beta=2)
}

resultados = {}
for nome, p in metricas.items():
    resultados[nome] = get_best_knn_by_metric(**p)['
                                        metric_value']

pprint(resultados)

# {
#   'Recall Micro': 0.99023,
#   'Recall Macro': 0.98358,
#   'F-Beta Micro': 0.99023,
#   'F-Beta Macro': 0.98508
# }
```

CÓDIGO: EXERCÍCIO 5

```
import numpy as np
import pandas as pd

from sklearn.linear_model import LogisticRegression
from sklearn.datasets import make_classification

from sklearn.model_selection import \
    cross_val_score, \
    train_test_split, \
    KFold, \
    LeaveOneOut

from sklearn.metrics import \
    matthews_corrcoef, \
    f1_score, \
    make_scorer

X, y = make_classification(
    n_samples=10000,
    n_features=2,
    n_informative=2,
    n_redundant=0,
    n_clusters_per_class=1,
    random_state=2002,
)

X_train, X_test, \
y_train, y_test = train_test_split(X, y,
                                   train_size=0.7,
                                   shuffle=True,
                                   random_state=2002)


model = LogisticRegression(penalty='l2')
model.fit(X_train, y_train)
```

```
mcc = make_scorer(matthews_corrcoef)

r = {}

# Houldout simples 50%
_, HS_X, _, HS_y = train_test_split(X_test, y_test,
                                    test_size=0.5,
                                    random_state=2002)

r['Houldout Simples'] = matthews_corrcoef(HS_y,
                                          model.predict(HS_X))

# Houldout repetido
HR = []
for _ in range(20):
    _, HS_X, _, HS_y = train_test_split(X_test, y_test,
                                        test_size=0.5,
                                        random_state=2002)

    HR.append(matthews_corrcoef(HS_y, model.predict(HS_X)))

r['Houldout Repetido (20 vezes)'] = np.mean(HR)


# KFold
cv = KFold(n_splits=10, shuffle=True, random_state=2002)
r['KFold'] = np.mean(cross_val_score(model,
                                     X_test, y_test,
                                     cv=cv, scoring=mcc))

# KFold Repetido
KR = []
for _ in range(20):
    cv = KFold(n_splits=10, shuffle=True, random_state=2002)
    HR.append(np.mean(cross_val_score(model,
                                      X_test, y_test,
                                      cv=cv, scoring=mcc)))

r['KFold Repetido (20 vezes)'] = np.mean(HR)

# LOOCV
loo = LeaveOneOut()
total_error = 0
for train_index, test_index in loo.split(X_test):
    X_tst = X_test[test_index, ]
    y_tst = y_test[test_index]
    # Evaluate the model on the test data
    error = model.score(X_tst, y_tst)
    total_error += error

r['Leave One Out'] = np.mean(total_error / len(X_test))

df = pd.DataFrame.from_dict(r, orient='index')
df.columns = ['Reg. Log.']
df.head()
```

REFERENCES

[1] Peter Bartlett, Vitaly Maiorov, and Ron Meir. Almost linear vc dimension bounds for piecewise polynomial networks. *Advances in neural information processing systems*, 11, 1998. Apud [2].

[2] Peter L Bartlett and Wolfgang Maass. Vapnik-chervonenkis dimension of neural nets. *The handbook of brain theory and neural networks*, pages 1188–1192, 2003.

[3] Eric Baum and David Haussler. What size net gives valid generalization? *Advances in neural information processing systems*, 1, 1988. Apud [2].