**KNN (K-vizinhos mais próximos)[[1]](#footnote-1)**

Este é um algoritmo simples que prevê pontos de dados desconhecidos com os seus vizinhos mais próximos. O valor de k é um fator crítico aqui quanto à precisão da predição. Ele determina o mais próximo ao calcular a distância usando funções básicas de distância como Euclideana.

No entanto, esse algoritmo precisa de alto poder de computação e precisamos normalizar os dados inicialmente para trazer todos os pontos de dados para o mesmo intervalo.

Código Python

from sklearn import datasets

from sklearn.neighbors import KNeighborsClassifier

#digit dataset from sklearn

digits = datasets.load\_digits()

#create the KNeighborsClassifier

clf = KNeighborsClassifier(n\_neighbors=6)

#set training set

x, y = digits.data[:-1], digits.target[:-1]

#train model

clf.fit(x, y)

#predict

y\_pred = clf.predict([digits.data[-1]])

y\_true = digits.target[-1]

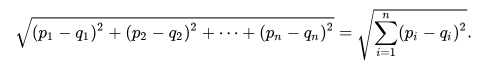
print(y\_pred)

print(y\_true)

Cálculo da Distância Euclidiana[[2]](#footnote-2)

Definição

A distância Euclidiana entre os pontos *P = (p1, p2,..., pn)* e *Q = (q1, q2,..., qn)*, num *espaço euclidiano* [[3]](#footnote-3) n-dimencional, é definida como:



Exemplo:[[4]](#footnote-4)

pi = 5, 7, 9

qj = 5, 5, 5

1º passo - Subtração de cada posição do vetor (pi – qj)

5 – 5 = 0

7 – 5 = 2

9 – 5 = 4

2º passo – Raiz quadrada da Soma dos quadros dos resultados do 1º passo.

Raiz(0² + 2² + 4²)

Raiz(0 + 4 + 16)

Raiz(20)

Resultado = 4,47

Base de dados utilizada para referencial e treinamento dos dados para a classificação KNN.

iris.csv[[5]](#footnote-5)

<https://www.youtube.com/watch?v=cCMi1fjfAUY>

12 passos para a classificação do K vizinhos mais próximos em Python

<https://www.youtube.com/watch?v=pTf4Vmwp6Vw>

3 super técnicas em data Science

<https://www.youtube.com/watch?v=5Nmzd9slQk0>

correlação e regressão linear simples

Anexo 1 – Código em Python – KNN Iris

knn\_iris.py

# Projeto Big Data - Uniasselvi - outubro / 2018  
# Dionisio Gause Junior  
# Missão: Implementar o metodo kNN para a flor IRIS  
# Desenvolimento: Baseado na aula do professor Sanderson Macedo  
# Disponível em: https://www.youtube.com/watch?v=cCMi1fjfAUY

#Importando bibliotecas necessárias  
import numpy as np  
import pandas as pd  
import matplotlib.pyplot as plt  
import seaborn as sb  
  
# carrega dados em dataframe df  
df = pd.read\_csv('iris.csv')  
  
# usado para identificar o nome das colunas  
print(df.columns)  
  
# usado para visualizar os dados  
print(df)  
  
# print usado para descrição dos dados  
print(df.describe())  
  
# Usado para visualizar a dispersão dos dados  
sb.pairplot(df, hue='Species')  
plt.show()  
  
# selecionando as caracteristicas para a classificação em um array  
X = np.array(df.drop('Species', 1))  
  
# visualizar os dados do array criado (X)  
print(X)  
  
# selecionando a classes para classificação em array  
y = np.array(df.Species)  
  
# visualizar os dados do array criado(y)  
print(y)  
  
# importando a funcao KNeighborsClassifier da biblioteca sklearn.neighbors  
from sklearn.neighbors import KNeighborsClassifier  
  
# informando o n\_neighbors a verificar - quanto maior, maior a probabilidade de acerto  
# sempre em valores impares (evitando empate na classificação)  
knn = KNeighborsClassifier(n\_neighbors=5)  
  
# Treinando um classificador com base nas caracteristicas (X) e classe (y)  
knn.fit(X, y)  
  
comp\_sepala = float(input('Digite o comprimento da Sépala: '))  
larg\_sepala = float(input('Digite a largura da Sépala....: '))  
comp\_petala = float(input('Digite o comprimento da Pétala: '))  
larg\_petala = float(input('Digite a largura da Pétala....: '))  
  
p = knn.predict([[comp\_sepala, larg\_sepala, comp\_petala, larg\_petala]])  
  
print(f'Com base nos dados informados e utilizando o método K-NN, é provável que seja uma Iris {p}\n'  
 f'Obrigado por utilizar nosso APP, volte sempre !')

Anexo 2 – Código em Python - Regressão linear

Reg\_linear.py

# Projeto Big Data - Uniasselvi - outubro / 2018  
# Dionisio Gause Junior  
# Desafio: Implementar o metodo de regressão linear simples  
# Referências:  
# https://www.youtube.com/watch?v=gzyq\_6wdtSk  
# https://medium.com/@felipebormann/aprendendo-scikit-learn-e-um-pouco-mais-de-python-6b27025f9d5b  
# http://neylsoncrepalde.github.io/2018-02-25-regressao-linear-python/  
# https://imasters.com.br/back-end/data-science-regressoes-com-python

# Importando bibliotecas necessárias

import numpy as np  
import matplotlib.pyplot as plt  
  
# Criando os arrays com as variáveis x e y a serem utilizados  
km\_percorridos = [8, 10, 12,16,20]  
x = np.array(km\_percorridos)  
  
calorias\_queimadas = [90, 112, 148, 189, 212]  
y = np.array(calorias\_queimadas)  
  
# chama a função polyfit informando os vetores e o grau do polinomio  
# onde a regressão linear o grau é 1 - equação da reta  
p1 = np.polyfit(x, y, 1)  
  
# a partir dos coeficientes linear e angular calcula-se o y predito  
yfit = p1[0] \* x + p1[1]  
  
# Calcula-se o resíduo (valor de y menos o predito)  
yresid = y - yfit  
  
# Calcula-se o somatório dos quadrados dos residuos  
SQresid = sum(pow(yresid,2))  
  
# Calcula-se o somatório da diferença entre o valor real e a média dos valores de y  
# quantidade de elementos de y vezes a variança  
SQtotal = len(y) \* np.var(y)  
  
# com base nos valores SQresid e SQtotal calcula-se o coeficiente de determinação  
R2 = 1 - SQresid/SQtotal  
  
# mostra o intercepto e inclinação  
print(f'\nCoeficiente angular = {p1[0]}\n')  
print(f'Coeficiente linear = {p1[1]}\n')  
  
# mostra o coeficiente de determinação  
print(f'Coeficiente de Determinação = {R2}')  
  
# determina-se as variáveis para apresentção do gráfico (plt)  
plt.title('Quantidade de km percorridos por calorias queimadas') #título do gráfico  
plt.xlabel('“Km percorridos') #Legenda na horizontal  
plt.ylabel('Calorias Queimadas') #Legenda do eixo Y  
plt.plot(x, y, 'o') # os dados nos dois eixos  
  
# valores preditos de y (linha)  
plt.plot(x, np.polyval(p1,x),'g--')  
plt.grid(True) # Grid = grades, zonas do gráfico  
plt.show() # mostra o gráfico

1. *<https://medium.com/@cristianofurquim/10-algoritmos-de-aprendizagem-de-m%C3%A1quinas-machine-learning-que-voc%C3%AA-precisa-saber-c49f9eefe319 - acessado em 15/09/2018 - 11:35>* [↑](#footnote-ref-1)
2. *<https://pt.wikipedia.org/wiki/Dist%C3%A2ncia\_euclidiana - acessado em 15/09/2018 - 11:44>* [↑](#footnote-ref-2)
3. *Espaço Euclidiano é um espaço vetorial real de dimensão finita munido de um produto interno.* [↑](#footnote-ref-3)
4. <https://www.youtube.com/watch?v=KBh4vnsCDc4 - acessado em 15/09/2018 - 09:30> [↑](#footnote-ref-4)
5. <https://raw.githubusercontent.com/uiuc-cse/data-fa14/gh-pages/data/iris.csv - acessado em 16/09/2018 - 18:30> [↑](#footnote-ref-5)