Knn Solução

K-Nearest Neighbor (KNN)

Empregue para este Laboratório o **material de Teoria da Trilha**.

IMPORTANTE: Antes de começar este Lab

Execute a célula final desse Lab. Ela irá **inicializar** o seu ambiente com os datasets requeridos para esse lab que poderão ser então lidos como arquivos locais, isto é `df = pd.read_csv('df.csv')` .

Após a execução você pode verificar os datasets criados na aba lateral do Google Colab ou no seu diretório de trabalho.

Exercício. CASE: Prevendo o Sucesso de Projetos e Faixa de Seguros (RESOLVIDO)

Aqui você vai empregar os datasets criados no set up do Lab.

Acesse os dados de

```
projects.csv
new_projects.csv
```

E veja com o `knn` é empregado para prever o status de novos projetos.

In [ ]:
```
# Acessando os dados
projects = pd.read_csv('projects.csv')
print(projects.head())

new_projects = pd.read_csv('new_projects.csv')
print(new_projects.head())
```

```
   Ana  Gabriela  Pedro  Luiz     Status
0    1         1      0     1  successed
1    1         1      1     1  successed
2    0         0      1     0     failed
3    0         0      1     1     failed
```

```
   Ana  Gabriela  Pedro  Luiz Status
0    0         1      1     0      ?
1    1         1      1     0      ?
```

Criando o modelo e prevendo os novos projetos:

In [ ]:
```
from sklearn import neighbors

# Prepara os dados para o Treinamento
X_train = projects.drop(columns=['Status'])           # Entradas
y_train = projects['Status']                          # Saída

# Declara o Modelo
n_neighbors = 3                                       # Parametros do modelo
clf = neighbors.KNeighborsClassifier(n_neighbors)     #

# Treinamento (Treina o Modelo)
clf.fit(X_train, y_train)                             # Treinamento

# Predição (Emprega o Modelo)
X_test = new_projects.drop(columns=['Status'])        # Dados para predição
y_pred = clf.predict(X_test)                          # Resposta do modelo

print(y_pred)
```

```
['failed' 'successed']
```

In [ ]:
```
new_projects['Status'] = y_pred
print(new_projects)
```

```
   Ana  Gabriela  Pedro  Luiz     Status
0    0         1      1     0     failed
1    1         1      1     0  successed
```

Exercício. **CASE: Prevendo a Faixa de Seguros**

Agora, acesse os dados de:

```
insurance.csv
new_insurance.csv
```

e empregue o modelo anterior para prever a Faixa de Seguros dos novos Seguros.

In [ ]:
```
# Acessando os Dados

# Seu código aqui
insurance = pd.read_csv('insurance.csv')
print(insurance.head())

new_insurance = pd.read_csv('new_insurance.csv')
print(new_insurance.head())
```

```
   Age  Income  Car_Value  Years_Hab Insurance
0   21     5.0         40          3      high
1   20     3.5         40          2      high
2   35     8.0         34         10       low
3   40    12.0         60         12       low
   Age  Income  Car_Value  Years_Hab Insurance
0   19       5         50          1         ?
```

```
1   50      10          45          15          ?
2   23       4          80           1          ?
```

In [ ]:

```
# Seu código aqui
from sklearn import neighbors

# Prepara os dados para o Treinamento
X_train = insurance.drop(columns=['Insurance'])           # Entradas
y_train = insurance['Insurance']                          # Saída

# Declara o Modelo
n_neighbors = 3                                         # Parametros do modelo
clf = neighbors.KNeighborsClassifier(n_neighbors)       #

# Treinamento (Treina o Modelo)
clf.fit(X_train, y_train)                               # Treinamento

# Predição (Emprega o Modelo)
X_test = new_insurance.drop(columns=['Insurance'])      # Dados para predição
y_pred = clf.predict(X_test)                            # Resposta do modelo

print(y_pred)
```

```
['high' 'low' 'high']
```

CASE: **Predição de Diagnósticos a partir de Dados de Imagens**

Agora vamos ver um caso de dados reais.

https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic)

Features are computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. They describe characteristics of the cell nuclei present in the image.

Os dados estão na URL: https://meusite.mackenzie.br/rogerio/TIC/breast-cancer-wisconsin.csv

Mas aqui empregue os dados já tratados no set up do Lab:

```
breast.csv
new_breast.csv
```

Exercício. Acesse e Explore os dados

1. Qual o atributo **classe**?
2. Todos os dados são numéricos? (por quê?)
3. Qual o tamanho dos dados? (linhas e atributos)
4. Quantos casos são **'B'enignos** e quantos **'M'alignos**?

In [ ]:

```
# Acessando os Dados

# Seu código aqui
breast = pd.read_csv('breast.csv')
print(breast.head())
```

```
new_breast = pd.read_csv('new_breast.csv')
print(new_breast.head())
```

```
        id  radius_mean  ...  fractal_dimension_worst  diagnosis
0   858986        14.25  ...                  0.11320          M
1   859575        18.94  ...                  0.06589          M
2   914101        12.46  ...                  0.07028          B
3  8911800        13.59  ...                  0.06263          B
4  88411702        13.75  ...                  0.06321          B

[5 rows x 32 columns]
        id  radius_mean  ...  fractal_dimension_worst  diagnosis
0  8712766        17.47  ...                  0.09300          ?
1   907409        10.48  ...                  0.09646          ?
2   901836        11.04  ...                  0.07287          ?
3  9111805        19.59  ...                  0.06091          ?
4   897880        10.05  ...                  0.07664          ?

[5 rows x 32 columns]
```

In [ ]:

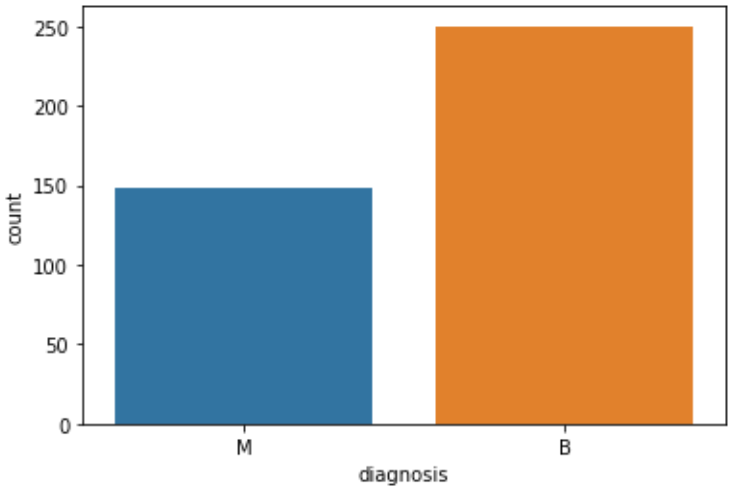
```
# Explore os dados
# Seu código aqui
breast.shape
breast.head()

sns.countplot(breast.diagnosis)
breast.isnull().sum()
breast.dtypes

breast.diagnosis.value_counts()
```

Out[ ]:

```
B    250
M    148
Name: diagnosis, dtype: int64
```



Exercício. Preparando os dados

A preparação dos dados é simples pois não existem dados nulos a serem tratados e todos os dados de entrada são numéricos.(*)

Mas **um atributo parece não estar adequado para o Treinamento. Qual?**

Elimine esse atributo do conjunto de dados de treinamento e dos novos casos.

(*) Neste momento, também não faremos aqui a normalização dos dados.

In [ ]:
```
# Seu código aqui
# delete the unwanted id column
breast.drop(columns=['id'], inplace=True)
new_breast.drop(columns=['id'], inplace=True)
```

Exercício. Classificando `new_breast` com K=3 Vizinhos mais Próximos

Treine agora o modelo com **k=3 vizinhos mais próximos** e faça a previsão dos novos casos `new_breast` .

In [ ]:
```
# Seu código aqui
from sklearn import neighbors

# Prepara os dados para o Treinamento
X_train = breast.drop(columns=['diagnosis'])
y_train = breast['diagnosis']

# Declara o Modelo
n_neighbors = 3                                     # Parametros do modelo
clf = neighbors.KNeighborsClassifier(n_neighbors)   #

# Treinamento (Treina o Modelo)
clf.fit(X_train, y_train)                           # Treinamento

# Predição (Emprega o Modelo)
X_test = new_breast.drop(columns=['diagnosis'])     # Dados para predição
y_pred = clf.predict(X_test)                        # Resposta do modelo

print(y_pred)
```

```
['M' 'B' 'B' 'M' 'B' 'B' 'B' 'B' 'B' 'B' 'M' 'B' 'B' 'M' 'B' 'M' 'M' 'B'
 'B' 'M' 'B' 'B' 'B' 'B' 'B' 'M' 'B' 'B' 'M' 'B' 'B' 'B' 'B' 'B' 'B' 'B'
 'M' 'M' 'B' 'B' 'B' 'B' 'B' 'B' 'M' 'M' 'B' 'M' 'B' 'M' 'B' 'B' 'B' 'B'
 'B' 'B' 'B' 'B' 'B' 'M' 'B' 'M' 'M' 'B' 'M' 'B' 'B' 'B' 'B' 'M' 'M' 'B'
 'M' 'B' 'B' 'B' 'B' 'B' 'M' 'B' 'M' 'B' 'M' 'B' 'M' 'M' 'M' 'B' 'M' 'B'
 'B' 'M' 'M' 'B' 'B' 'B' 'M' 'M' 'B' 'B' 'M' 'B' 'B' 'B' 'M' 'B' 'B' 'M'
 'B' 'M' 'B' 'M' 'M' 'M' 'M' 'B' 'B' 'B' 'B' 'M' 'M' 'B' 'B' 'B' 'M' 'B'
 'B' 'B' 'B' 'M' 'B' 'B' 'M' 'B' 'B' 'M' 'M' 'B' 'B' 'B' 'B' 'M' 'B' 'B'
 'B' 'B' 'B' 'B' 'B' 'M' 'M' 'B' 'M' 'M' 'B' 'B' 'M' 'B' 'B' 'B' 'B' 'B'
 'B' 'B' 'B' 'B' 'M' 'M' 'B' 'B' 'B']
```

Exercício.

Quanto casos Benignos e Malignos foram previstos?

In [ ]:
```
# Seu código
print('Bs =', sum(y_pred == 'B'))
print('Ms =', sum(y_pred == 'M'))
```

```
Bs = 115
Ms = 56
```

Comparando resultados

Suponha que temos as respostas do casos de `new_breast` (Ohhh!). Neste caso podemos então comparar com nosso modelo.

De fato as respostas estão no data-set `new_breast_answers` e então podemos comparar o resultado obtido com a resposta do modelo.

In [ ]:
```
new_breast_answers = pd.read_csv('new_breast_answers.csv')
new_breast_answers.head()

pd.set_option('display.max_rows', 500)
pd.concat([pd.DataFrame(y_pred, columns=['predicted']), new_breast_answers],ax
```

Out[ ]:

	predicted	diagnosis
0	M	M
1	B	B
2	B	B
3	M	M
4	B	B
5	B	B
6	B	B
7	B	B
8	B	B
9	B	B
10	M	M
11	B	M
12	B	B
13	M	M
14	B	M
15	M	M
16	M	M
17	B	B
18	B	B
19	M	M
20	B	B
21	B	B
22	B	B
23	B	B
24	B	M
25	M	M
26	B	B
27	B	B

28	M	M
29	B	B
30	B	B
31	B	B
32	B	B
33	B	B
34	B	B
35	B	B
36	M	M
37	M	B
38	B	B
39	B	B
40	B	B
41	B	B
42	B	B
43	B	M
44	M	M
45	M	M
46	B	B
47	M	M
48	B	B
49	M	M
50	B	B
51	B	B
52	B	M
53	B	B
54	B	B
55	B	B
56	B	B
57	B	B
58	B	B
59	M	M
60	B	B
61	M	M
62	M	M
63	B	B

64	M	M
65	B	B
66	B	B
67	B	B
68	B	B
69	M	M
70	M	M
71	B	B
72	M	M
73	B	B
74	B	B
75	B	B
76	B	B
77	B	B
78	M	M
79	B	B
80	M	M
81	B	B
82	M	M
83	B	B
84	M	M
85	M	M
86	M	M
87	B	B
88	M	M
89	B	B
90	B	B
91	M	M
92	M	M
93	B	B
94	B	B
95	B	B
96	M	B
97	M	M
98	B	B
99	B	B
100	M	M

100	M	M
101	B	B
102	B	B
103	B	B
104	M	M
105	B	B
106	B	B
107	M	M
108	B	B
109	M	M
110	B	B
111	M	M
112	M	M
113	M	M
114	M	M
115	B	B
116	B	B
117	B	B
118	B	B
119	M	B
120	M	M
121	B	M
122	B	B
123	B	M
124	M	M
125	B	B
126	B	B
127	B	B
128	B	B
129	M	M
130	B	M
131	B	B
132	M	M
133	B	B
134	B	B
135	M	M
136	M	M

137	B	B
138	B	M
139	B	B
140	B	B
141	M	M
142	B	B
143	B	M
144	B	B
145	B	B
146	B	B
147	B	B
148	B	B
149	M	M
150	M	M
151	B	B
152	M	M
153	M	M
154	B	M
155	B	B
156	M	M
157	B	B
158	B	B
159	B	B
160	B	B
161	B	B
162	B	B
163	B	B
164	B	B
165	B	B
166	M	M
167	M	M
168	B	B
169	B	B
170	B	B

Exercício.

Calcule o percentual de acerto obtido.

In [ ]:
```
# Seu código
temp = pd.concat([pd.DataFrame(y_pred, columns=['predicted']), new_breast_answ
percentual = sum(temp.predicted == temp.diagnosis)/len(temp)

print('Percentual de acertos: ', format(percentual, '.2f') )
```

```
Percentual de acertos:  0.92
```

Exercício. Classificando `new_breast` com K=7 Vizinhos mais Próximos

Reexecute o modelo agora com **k=7 vizinhos mais próximos** e verifique o novo percentual de acerto.

In [ ]:
```
# Seu código aqui
from sklearn import neighbors

# Prepara os dados para o Treinamento
X_train = breast.drop(columns=['diagnosis'])
y_train = breast['diagnosis']

# Declara o Modelo
n_neighbors = 7                                         # Parametros do modelo
clf = neighbors.KNeighborsClassifier(n_neighbors)       #

# Treinamento (Treina o Modelo)
clf.fit(X_train, y_train)                               # Treinamento

# Predição (Emprega o Modelo)
X_test = new_breast.drop(columns=['diagnosis'])         # Dados para predição
y_pred = clf.predict(X_test)                            # Resposta do modelo

# print(y_pred)

temp = pd.concat([pd.DataFrame(y_pred, columns=['predicted']), new_breast_answ
percentual = sum(temp.predicted == temp.diagnosis)/len(temp)

print('Percentual de acertos: ', format(percentual, '.2f') )
```

```
Percentual de acertos:  0.94
```

LAB SET UP