trabalho_4

May 7, 2022

[1]:
```
!pip install pandas
!pip install scikit-learn
```

```
Requirement already satisfied: pandas in /opt/conda/lib/python3.9/site-packages
(1.4.2)
Requirement already satisfied: python-dateutil>=2.8.1 in
/opt/conda/lib/python3.9/site-packages (from pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /opt/conda/lib/python3.9/site-
packages (from pandas) (2022.1)
Requirement already satisfied: numpy>=1.18.5 in /opt/conda/lib/python3.9/site-
packages (from pandas) (1.21.6)
Requirement already satisfied: six>=1.5 in /opt/conda/lib/python3.9/site-
packages (from python-dateutil>=2.8.1->pandas) (1.16.0)
Requirement already satisfied: scikit-learn in /opt/conda/lib/python3.9/site-
packages (1.0.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in
/opt/conda/lib/python3.9/site-packages (from scikit-learn) (3.1.0)
Requirement already satisfied: joblib>=0.11 in /opt/conda/lib/python3.9/site-
packages (from scikit-learn) (1.1.0)
Requirement already satisfied: numpy>=1.14.6 in /opt/conda/lib/python3.9/site-
packages (from scikit-learn) (1.21.6)
Requirement already satisfied: scipy>=1.1.0 in /opt/conda/lib/python3.9/site-
packages (from scikit-learn) (1.8.0)
```

[2]:
```
from collections import Counter

import pandas as pd
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import confusion_matrix, f1_score, accuracy_score,␣
 ↪recall_score, precision_score
from sklearn.cluster import KMeans
```

[3]:
```
df = pd.read_csv("https://sbcb.inf.ufrgs.br/data/cumida/Genes/Breast/GSE70947/
 ↪Breast_GSE70947.csv")
df.head()
```

[3]:

	samples	type	NM_144987	NM_013290	ENST00000322831	NM_001625	lincRNA:chr7:226042-232442_R	NM_032391	ENST00000238571	XR_108906	…
0	GSM1823702_252800417016_S01_GE1_107_Sep09_1_2	normal	8.693318	7.718016	6.044438	10.747077	9.133777	4.735581	5.634732	4.670231	…
1	GSM1823703_252800417016_S01_GE1_107_Sep09_2_1	normal	9.375980	7.072232	6.976741	10.429671	9.526500	5.221089	5.425187	4.860931	…
2	GSM1823704_252800416877_S01_GE1_107_Sep09_2_3	normal	8.943442	7.964573	6.269055	10.825025	9.396855	5.258506	5.824921	4.964604	…
3	GSM1823705_252800416894_S01_GE1_107_Sep09_1_1	normal	9.020798	7.824639	6.165165	11.646788	8.776462	4.648655	6.676692	4.770186	…
4	GSM1823706_252800416894_S01_GE1_107_Sep09_1_3	normal	8.806154	7.555348	6.230969	11.635247	8.911383	4.518054	6.520691	4.540453	…

	lincRNA:chr4:77860976-77869926_F	NM_152343	NM_001005327	NM_001039355	lincRNA:chr21:44456656-44468556_R	lincRNA:chr9:4869500-4896050_F	NM_016053	NM_001080425	ENST00000555638	ENST00000508993
0	7.570363	6.368684	4.784042	10.747723	5.090500	5.994149	10.649336	8.969439	4.985693	5.090500
1	7.903335	5.713115	4.421074	11.299200	4.447052	4.421074	10.746854	8.174489	4.464177	4.536891
2	7.705765	6.595364	4.410870	10.576807	5.003699	6.529257	10.430034	8.473468	4.668447	5.084127
3	6.633058	5.786781	4.572984	11.175090	4.990888	6.669871	11.110395	8.880818	4.537626	4.648655
4	6.211581	5.538635	4.613828	12.014365	4.979883	6.414621	10.909805	9.526500	4.670490	4.613828

[5 rows x 35983 columns]

0.0.1 a) Realize a normalização do dataset de estudo utilizando o z-score

[4]:
```
sc = StandardScaler()
X = sc.fit_transform(df.iloc[:,2:])
X
```

[4]:
```
array([[-0.29044738,  0.15716683, -0.8408758 , …,  0.29101254,
         0.57661691,  1.16698088],
       [ 1.44237276, -0.96712986,  1.08865853, …, -1.01273948,
        -0.75420077, -0.79655215],
       [ 0.34444946,  0.58641694, -0.37599821, …, -0.52240203,
        -0.23294027,  1.1443803 ],
       …,
       [ 0.61844411,  0.04204796, -1.2523575 , …, -0.05398532,
        -0.51800249, -0.47654098],
       [-1.75940706,  0.81641611, -0.19732212, …,  1.42053716,
        -0.62246924, -0.63175092],
       [ 0.05034099, -0.15467297, -0.17834283, …,  1.55371374,
         0.49169946,  0.03375616]])
```

0.0.2 b) Realize a divisão do dataset de estudo (resultante da letra 'a') em conjunto de teste e conjunto de treinamento de forma a contemplar o conceito de amostragem estratificada

[5]:
```
le = LabelEncoder()
y = le.fit_transform(df.type)
```

[6]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,␣
↪stratify=y, random_state=42)
```

0.0.3 c) Fazendo uso de bibliotecas (por exemplo, scikit-learn https://scikit-learn.org/stable) crie um classificador SVM para o dataset de estudo. O treinamento do classificador deve ser realizada com base no grupo de treinamento criado no item 'b'.

[7]:
```
clf = SVC(kernel='linear')
```

[8]:
```
clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)
```

0.0.4 d) A partir criado na letra 'c' e do cojunto de testes, avalie o classificador considerando as seguintes métricas: (i) a matriz de confusão; (ii) a acurácia; (iii) Sensitivity; (iv) Specificity; e (v) F1-score. O valor destas métricas deve ser reportado. Ao analisar as métricas você considera que o classificador teve um desempenho adequado?

[9]:
```
pd.DataFrame(confusion_matrix(y_test, y_pred), index=le.classes_, columns=le.
↪classes_)
```

[9]:

	breast_adenocarcinoma	normal
breast_adenocarcinoma	37	6
normal	3	41

[10]:
```
accuracy_score(y_test, y_pred)
```

[10]: 0.896551724137931

[11]:
```
recall_score(y_test, y_pred)
```

[11]: 0.9318181818181818

[12]:
```
f1_score(y_test, y_pred)
```

[12]: 0.9010989010989012

O classificador teve um bom desempenho pois ambas as classes - normal e breast_adenocarcinoma - tiveram um boa acurácia, sem desbalanceamento.

0.1 e) Fazendo uso de bibliotecas (por exemplo, scikit-learn https://scikit-learn.org/stable) utilize o método k-means para analisar o dataset de estudo (preparado no item 'a') considerando os seguintes cenários: existência de 2 grupos; 3 grupos e 4 grupos. Para cada um dos cenários reporte o número de amostras presentes de cada grupo.

[13]:
```
for n_groups in [2, 3, 4]:
    print(dict(Counter(KMeans(n_clusters=n_groups, random_state=42).
↪fit_predict(X_train))))
```

```
{0: 99, 1: 103}
{1: 65, 0: 85, 2: 52}
{0: 59, 1: 80, 2: 62, 3: 1}
```