4_modelo

January 4, 2021

[49]:
```
import numpy as np
import pandas as pd


from sklearn.model_selection import train_test_split, cross_val_score,␣
 ↪cross_validate
from sklearn.ensemble import RandomForestClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC
# import xgboost as xgb

from sklearn.feature_selection import RFE
from sklearn.feature_extraction.text import TfidfVectorizer

from nltk.corpus import stopwords

import matplotlib.pyplot as plt
```

[2]:
```
database = pd.read_csv('database.csv')
database.head()
```

[2]:
```
        idx                                          title_raw  \
0   0387659  \n\n                  SYSTEMS AND METHODS FO…
1  10729058  \n\n                  Systems and methods fo…
2   6745128  \n\n                  Methods and systems fo…
3   6549852  \n\n                  Methods and systems fo…
4   0018431  \n\n                  METHODS AND SYSTEMS FO…

                                           text_raw  \
0  \n           The present disclosure provides …
1  \n           The present disclosure provides …
2  \n           Methods and systems for characte…
3  \n           Methods and systems for characte…
4  \n           Methods and systems for characte…

                                               title  \
0  systems and methods for adjusting the output o…
```

```
1  systems and methods for adjusting the output o…
2          methods and systems for managing farmland
3          methods and systems for managing farmland
4          methods and systems for managing farmland

                                                text  \
0  the present disclosure provides systems and me…
1  the present disclosure provides systems and me…
2  methods and systems for characterizing and man…
3  methods and systems for characterizing and man…
4  methods and systems for characterizing and man…

                                             content  \
0  systems and methods for adjusting the output o…
1  systems and methods for adjusting the output o…
2  methods and systems for managing farmland meth…
3  methods and systems for managing farmland meth…
4  methods and systems for managing farmland meth…

                                             _topic_
0  method_crop_use; plurality_equipment_datum; le…
1  method_crop_use; plurality_equipment_datum; le…
2  method_crop_use; plurality_equipment_datum; le…
3  method_crop_use; plurality_equipment_datum; le…
4  method_crop_use; plurality_equipment_datum; le…
```

[3]:
```
database_train = database[database['_topic_'].notna()]
# database_ = database[not database['_topic_'].isna()]
```

[4]:
```
vectorizer = TfidfVectorizer()
doc_vec = vectorizer.fit_transform(database_train['content'])
```

[5]:
```
df_doc_vec = pd.DataFrame(doc_vec.toarray(), columns = vectorizer.
 ↪get_feature_names())
df_doc_vec.shape
```

[5]:
```
(300, 2775)
```

[6]:
```
df_doc_vec.head()
```

[6]:
```
   001  01dhd10  08pb  09dsq1   10  102  104  106  108  10845353  …  yields  \
0  0.0      0.0   0.0     0.0  0.0  0.0  0.0  0.0  0.0       0.0  …     0.0
1  0.0      0.0   0.0     0.0  0.0  0.0  0.0  0.0  0.0       0.0  …     0.0
2  0.0      0.0   0.0     0.0  0.0  0.0  0.0  0.0  0.0       0.0  …     0.0
3  0.0      0.0   0.0     0.0  0.0  0.0  0.0  0.0  0.0       0.0  …     0.0
4  0.0      0.0   0.0     0.0  0.0  0.0  0.0  0.0  0.0       0.0  …     0.0
```

	you	zea	zein	zeolite	zinc	zn	zone	zones	mol
0	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	0.000000	0.0
1	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	0.000000	0.0
2	0.0	0.0	0.0	0.0	0.0	0.0	0.296266	0.101953	0.0
3	0.0	0.0	0.0	0.0	0.0	0.0	0.296266	0.101953	0.0
4	0.0	0.0	0.0	0.0	0.0	0.0	0.296266	0.101953	0.0

[5 rows x 2775 columns]

[7]:
```
X = df_doc_vec
y = database_train['_topic_'].to_numpy()
```

[8]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=0)
```

[9]:
```
print(X_train.shape, X_test.shape, y_train.shape, y_test.shape)
```

```
(225, 2775) (75, 2775) (225,) (75,)
```

[10]:
```
# teste com random forest
```

[11]:
```
rf = RandomForestClassifier(random_state=42)
rf.fit(X_train, y_train)
rf.score(X_test, y_test)
```

[11]:
```
0.5466666666666666
```

[12]:
```
rf_scores = cross_val_score(rf, X, y, cv=10)
print(rf_scores)
print(np.mean(rf_scores))
```

```
/home/henrique/anaconda3/lib/python3.8/site-
packages/sklearn/model_selection/_split.py:670: UserWarning: The least populated
class in y has only 1 members, which is less than n_splits=10.
  warnings.warn(("The least populated class in y has only %d"

[0.8        0.73333333 0.63333333 0.63333333 0.6        0.53333333
 0.46666667 0.56666667 0.6        0.73333333]
0.63
```

[13]:
```
# teste com Naive Bayes
```

[14]:
```
nb = GaussianNB()
nb.fit(X_train, y_train)
nb.score(X_test, y_test)
```

[14]:
```
0.5466666666666666
```

[15]:
```
nb_scores = cross_val_score(nb, X, y, cv=10)
print(nb_scores)
print(np.mean(nb_scores))
```

```
/home/henrique/anaconda3/lib/python3.8/site-
packages/sklearn/model_selection/_split.py:670: UserWarning: The least populated
class in y has only 1 members, which is less than n_splits=10.
  warnings.warn(("The least populated class in y has only %d"
```

```
[0.8        0.73333333 0.63333333 0.63333333 0.6        0.53333333
 0.46666667 0.56666667 0.63333333 0.73333333]
0.6333333333333333
```

[16]:
```
# teste com SVM
```

[17]:
```
svm = SVC(C=15, random_state=150, probability=True)
svm.fit(X_train, y_train)
svm.score(X_test, y_test)
```

[17]:
```
0.5466666666666666
```

[18]:
```
svm_scores = cross_val_score(svm, X, y, cv=10)
print(svm_scores)
print(np.mean(svm_scores))
```

```
/home/henrique/anaconda3/lib/python3.8/site-
packages/sklearn/model_selection/_split.py:670: UserWarning: The least populated
class in y has only 1 members, which is less than n_splits=10.
  warnings.warn(("The least populated class in y has only %d"
```

```
[0.76666667 0.73333333 0.63333333 0.63333333 0.56666667 0.5
 0.46666667 0.56666667 0.63333333 0.73333333]
0.6233333333333333
```

[19]:
```
# Removendo colunas que sejam stopwords
```

[20]:
```
column_names = df_doc_vec.columns.tolist()
```

[21]:
```
keep = []
for column_name in column_names:
    keep.append(column_name not in stopwords.words('english'))
print(len(keep), sum(keep))
```

```
2775 2687
```

[22]:
```
df_doc_vec_filtered = df_doc_vec[df_doc_vec.columns[keep]]
```

[23]:
```
X1 = df_doc_vec_filtered
X1.shape
```

[23]:
```
(300, 2687)
```

[24]:
```
X_train, X_test, y_train, y_test = train_test_split(X1, y, random_state=0)
```

[25]:
```
# teste com random forest
```

[26]:
```
rf = RandomForestClassifier(random_state=185)
rf.fit(X_train, y_train)
rf.score(X_test, y_test)
```

[26]:
```
0.5466666666666666
```

[27]:
```
rf_scores = cross_val_score(rf, X1, y, cv=10)
print(rf_scores)
print(rf_scores.mean())
```

```
/home/henrique/anaconda3/lib/python3.8/site-
packages/sklearn/model_selection/_split.py:670: UserWarning: The least populated
class in y has only 1 members, which is less than n_splits=10.
  warnings.warn(("The least populated class in y has only %d"

[0.8        0.73333333 0.63333333 0.63333333 0.6        0.5
 0.46666667 0.56666667 0.63333333 0.73333333]
0.63
```

[ ]:
```
```

[28]:
```
# Removendo caracteristicas
```

[29]:
```
model = RandomForestClassifier(n_estimators=100)
```

[30]:
```
rfe = RFE(model, n_features_to_select=20)
rfe.fit(X1, y)
```

[30]:
```
RFE(estimator=RandomForestClassifier(), n_features_to_select=20)
```

[31]:
```
X1.columns[rfe.support_]
```

[31]:
```
Index(['agronomy', 'also', 'another', 'complements', 'corn', 'cultures',
       'described', 'disclosed', 'includes', 'invention', 'method', 'methods',
       'one', 'plant', 'plants', 'provided', 'relates', 'seed', 'seeds',
       'using'],
      dtype='object')
```

[32]:
```
X1_rfe = X1[X1.columns[rfe.support_]]
X1_rfe.shape
```

[32]:
```
(300, 20)
```

[33]:
```
X1_rfe.head()
```

[33]:

	agronomy	also	another	complements	corn	cultures	described	disclosed	\
0	0.305484	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
1	0.305484	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
2	0.073199	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
3	0.073199	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
4	0.073199	0.0	0.0	0.0	0.0	0.0	0.0	0.0	

	includes	invention	method	methods	one	plant	plants	provided	\
0	0.0	0.0	0.0	0.068008	0.0	0.0	0.0	0.000000	
1	0.0	0.0	0.0	0.068008	0.0	0.0	0.0	0.000000	
2	0.0	0.0	0.0	0.065183	0.0	0.0	0.0	0.034642	
3	0.0	0.0	0.0	0.065183	0.0	0.0	0.0	0.034642	
4	0.0	0.0	0.0	0.065183	0.0	0.0	0.0	0.034642	

	relates	seed	seeds	using
0	0.0	0.000000	0.0	0.0
1	0.0	0.000000	0.0	0.0
2	0.0	0.036034	0.0	0.0
3	0.0	0.036034	0.0	0.0
4	0.0	0.036034	0.0	0.0

[34]:
```
X_train, X_test, y_train, y_test = train_test_split(X1_rfe, y, random_state=42)
print(X_train.shape, X_test.shape, y_train.shape, y_test.shape)
```

```
(225, 20) (75, 20) (225,) (75,)
```

[35]:
```
# teste com random forest
```

[36]:
```
rf = RandomForestClassifier(random_state=42)
```

[37]:
```
rf.fit(X_train, y_train)
rf.score(X_test, y_test)
```

[37]:
```
0.6266666666666667
```

[38]:
```
rf_scores = cross_val_score(rf, X1_rfe, y, cv=10)
print(rf_scores)
print(np.mean(rf_scores))
```

```
/home/henrique/anaconda3/lib/python3.8/site-
packages/sklearn/model_selection/_split.py:670: UserWarning: The least populated
```

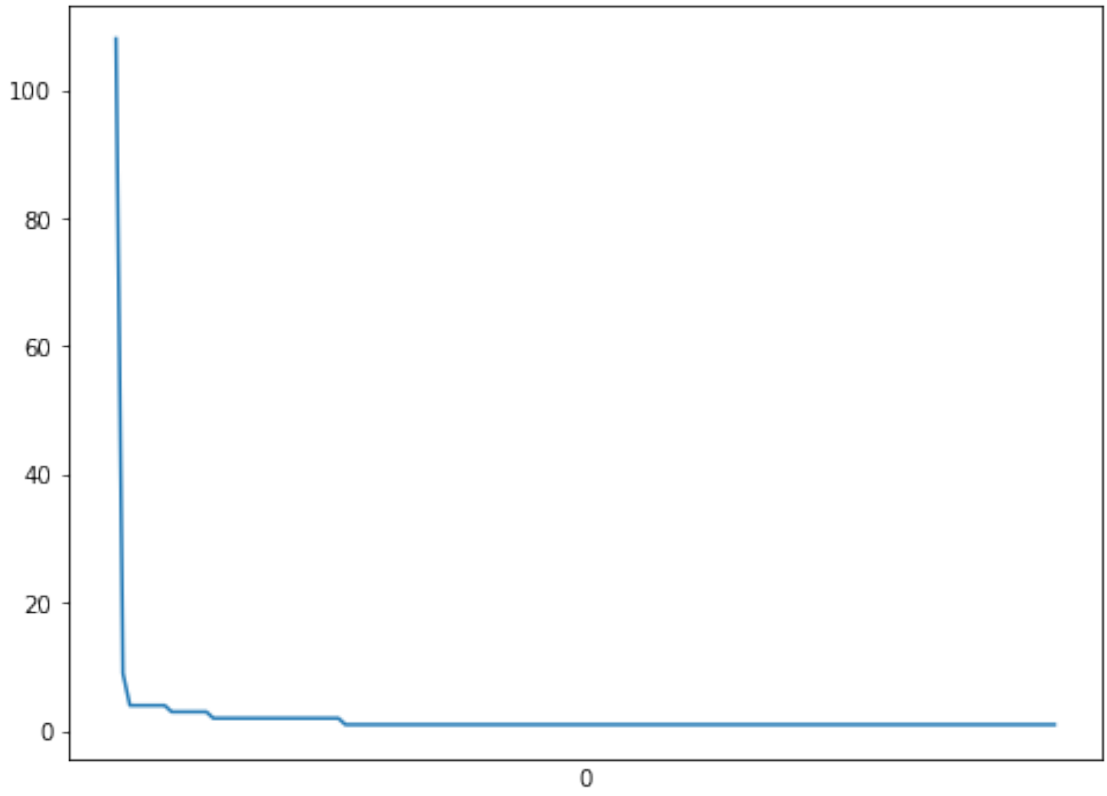
```
class in y has only 1 members, which is less than n_splits=10.
  warnings.warn(("The least populated class in y has only %d"

[0.8        0.73333333 0.63333333 0.63333333 0.56666667 0.5
 0.46666667 0.53333333 0.6        0.73333333]
0.6199999999999999
```

[ ]:
```
####### Avaliando a distribuicao do _topic_
```

[61]:
```
pd.DataFrame(y).value_counts().plot(xticks=[], figsize = [8,6])
```

[61]:
```
<AxesSubplot:xlabel='0'>
```



[ ]:
```

```