fake-real-news-classification

April 2, 2024

[1]:

```python
# This Python 3 environment comes with many helpful analytics libraries installed
# It is defined by the kaggle/python Docker image: https://github.com/kaggle/docker-python
# For example, here's several helpful packages to load

import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

# Input data files are available in the read-only "../input/" directory
# For example, running this (by clicking run or pressing Shift+Enter) will list all files under the input directory

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))

# You can write up to 20GB to the current directory (/kaggle/working/) that gets preserved as output when you create a version using "Save & Run All"
# You can also write temporary files to /kaggle/temp/, but they won't be saved outside of the current session
```

```
/kaggle/input/fake-news/fake.csv
```

[2]:

```python
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
from sklearn.feature_extraction.text import TfidfVectorizer,CountVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier
from sklearn.neighbors import KNeighborsClassifier
from xgboost import XGBClassifier
from sklearn.metrics import roc_curve,auc
from sklearn.metrics import classification_report, accuracy_score
from sklearn.pipeline import Pipeline
from sklearn.model_selection import cross_val_score, train_test_split
from urllib.parse import urlparse
import warnings
warnings.filterwarnings("ignore")
```

[3]:

```
data=pd.read_csv("/kaggle/input/fake-news/fake.csv")
```

[4]:

```
data.isnull().sum()
```

[4]:

```
uuid                     0
ord_in_thread            0
author                2424
published                0
title                  680
text                    46
language                 0
crawled                  0
site_url                 0
country                176
domain_rank           4223
thread_title            12
spam_score               0
main_img_url          3643
replies_count            0
participants_count       0
likes                    0
comments                 0
shares                   0
type                     0
dtype: int64
```

[5]:

```
data.country.value_counts()
```

[5]:

```
US    10367
GB      831
RU      400
DE      224
FR      207
TV      201
EU      112
CA      103
IS      100
ES      100
NL       55
ME       34
IN       23
BG       19
```

```
CO        17
LI        10
IR         7
EE         4
ZA         3
SG         2
IO         1
SE         1
AU         1
CH         1
Name: country, dtype: int64
```

[6]:

```
data.country.fillna("US",inplace=True)
```

[7]:

```
data.type.value_counts()
```

[7]:

```
bs            11492
bias            443
conspiracy      430
hate            246
satire          146
state           121
junksci         102
fake             19
Name: type, dtype: int64
```

[8]:

```
value_counts=data.language.value_counts()
to_remove=value_counts[value_counts<500].index
data.replace(to_remove,np.nan,inplace=True)
```

[9]:

```
value_counts=data.country.value_counts()
to_remove=value_counts[value_counts<20].index
data.replace(to_remove,np.nan,inplace=True)
```

[10]:

```
columns=['uuid', 'ord_in_thread', 'published','language', 'crawled',␣
 ↪'domain_rank', 'replies_count','participants_count', 'likes', 'comments',␣
 ↪'shares',"main_img_url"]
data.drop(columns,axis=1,inplace=True)
data.dropna(axis=0,inplace=True)
```

[11]:

```
stop_words=stopwords.words("english")
def stemmer(txt):
    #txt=txt.lower()
    words=word_tokenize(txt)
    words=[w for w in words if w.isalpha()]
    words=[w for w in words if not w in stop_words]
    return( " ".join(words ))
```

```
data["title"]=data["title"].apply(stemmer)
data["text"]=data["text"].apply(stemmer)
data["thread_title"]=data["thread_title"].apply(stemmer)
```

```
[12]: tfidf=TfidfVectorizer()
```

```
[13]: data["title"]=tfidf.fit_transform(data["title"]).toarray()
```

```
[14]: data["text"]=tfidf.fit_transform(data["text"]).toarray()
```

```
[15]: data["thread_title"]=tfidf.fit_transform(data["thread_title"]).toarray()
```

```
[16]: data["author"]=tfidf.fit_transform(data["author"]).toarray()
```

```
[17]: data["site_url"]=tfidf.fit_transform(data["site_url"]).toarray()
```

```
[18]: data.head(5)
```

```
[18]:    author  title  text  site_url country  thread_title  spam_score  type
      0     0.0    0.0   0.0  0.984487      US           0.0       0.000  bias
      1     0.0    0.0   0.0  0.984487      US           0.0       0.000  bias
      2     0.0    0.0   0.0  0.984487      US           0.0       0.000  bias
      3     0.0    0.0   0.0  0.984487      US           0.0       0.068  bias
      4     0.0    0.0   0.0  0.984487      US           0.0       0.865  bias
```

```
[19]: data["spam_score"]=data["spam_score"]-0.5
```

```
[20]: news_type=[]
      for i in data["spam_score"]:
          if(i<0):
              news_type.append("0")
          else:
              news_type.append("1")
      data["news_type"]=news_type
```

```
[21]: data.drop("spam_score",axis=1,inplace=True)
      data.head()
```

```
[21]:    author  title  text  site_url country  thread_title  type news_type
      0     0.0    0.0   0.0  0.984487      US           0.0  bias         0
      1     0.0    0.0   0.0  0.984487      US           0.0  bias         0
      2     0.0    0.0   0.0  0.984487      US           0.0  bias         0
      3     0.0    0.0   0.0  0.984487      US           0.0  bias         0
      4     0.0    0.0   0.0  0.984487      US           0.0  bias         1
```

```
[22]: data.type.value_counts()
      value_counts=data.type.value_counts()
```

```
to_remove=value_counts[value_counts<20].index
data.replace(to_remove,np.nan,inplace=True)
```

[23]:

```
data.isnull().sum()
```

[23]:

```
author           0
title            0
text             0
site_url         0
country          0
thread_title     0
type            19
news_type        0
dtype: int64
```

[24]:

```
data.country.value_counts()
```

[24]:

```
US    8612
GB     546
RU     124
EU     111
TV     101
ES     100
IS      99
DE      62
FR      36
NL      34
ME      34
IN      23
CA       3
Name: country, dtype: int64
```

[25]:

```
data.head(5)
```

[25]:

```
   author  title  text  site_url country  thread_title  type news_type
0     0.0    0.0   0.0  0.984487      US           0.0  bias         0
1     0.0    0.0   0.0  0.984487      US           0.0  bias         0
2     0.0    0.0   0.0  0.984487      US           0.0  bias         0
3     0.0    0.0   0.0  0.984487      US           0.0  bias         0
4     0.0    0.0   0.0  0.984487      US           0.0  bias         1
```

[26]:

```
data=pd.get_dummies(data=data,columns=["country","type"])
```

[27]:

```
y=data["news_type"].values
x=data.drop("news_type",axis=1)
x=x.values
```

[28]:

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=0)
```

[29]:

```
LR=LogisticRegression()
model_LR=LR.fit(x_train,y_train)
predict=model_LR.predict(x_test)
```

[ ]:

```

```

[30]:

```
LR=LogisticRegression()
model_LR=LR.fit(x_train,y_train)
KNN=KNeighborsClassifier()
model_KNN=KNN.fit(x_train,y_train)
DTC=DecisionTreeClassifier(random_state=0)
model_DTC=DTC.fit(x_train,y_train)
RFC=RandomForestClassifier(random_state=0)
model_RFC=RFC.fit(x_train,y_train)
GBC=GradientBoostingClassifier(random_state=0)
model_GBC=GBC.fit(x_train,y_train)
XGB=XGBClassifier()
model_XGB=XGB.fit(x_train,y_train)


models=[model_LR,model_KNN,model_DTC,model_RFC,model_GBC,model_XGB]
for model in models:
    name=model.__class__.__name__
    R2=cross_val_score(model,x_test,y_test,cv=10,verbose=False).mean()
    ␣
  ↪error=-cross_val_score(model,x_test,y_test,cv=10,scoring="neg_mean_squared_error",verbose=F
  ↪mean()
    predict=model_LR.predict(x_test)
    print(name + ":")
    print("*"*20)
    print("R-squared")
    print(R2)
    print("Error")
    print(np.sqrt(error))
    print("classification Report")
    print(classification_report(y_test,predict))
    print("accuracy")
    print(accuracy_score(y_test,predict))
    print("*"*20)
```

```
[08:35:44] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the
default evaluation metric used with the objective 'binary:logistic' was changed
from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore
the old behavior.
LogisticRegression:
```

```
********************
R-squared
0.9843203609701071
Error
0.12521836538580447
classification Report
              precision    recall  f1-score   support

           0       0.98      1.00      0.99      1946
           1       0.00      0.00      0.00        31

    accuracy                           0.98      1977
   macro avg       0.49      0.50      0.50      1977
weighted avg       0.97      0.98      0.98      1977

accuracy
0.9843196762771876
********************
KNeighborsClassifier:
********************
R-squared
0.9843203609701071
Error
0.12521836538580447
classification Report
              precision    recall  f1-score   support

           0       0.98      1.00      0.99      1946
           1       0.00      0.00      0.00        31

    accuracy                           0.98      1977
   macro avg       0.49      0.50      0.50      1977
weighted avg       0.97      0.98      0.98      1977

accuracy
0.9843196762771876
********************
DecisionTreeClassifier:
********************
R-squared
0.9843203609701071
Error
0.12521836538580447
classification Report
              precision    recall  f1-score   support

           0       0.98      1.00      0.99      1946
           1       0.00      0.00      0.00        31
    accuracy                           0.98      1977
   macro avg       0.49      0.50      0.50      1977
weighted avg       0.97      0.98      0.98      1977

accuracy
0.9843196762771876
********************
RandomForestClassifier:
********************
R-squared
0.9843203609701071
Error
0.12521836538580447
classification Report
              precision    recall  f1-score   support

           0       0.98      1.00      0.99      1946
           1       0.00      0.00      0.00        31

    accuracy                           0.98      1977
   macro avg       0.49      0.50      0.50      1977
weighted avg       0.97      0.98      0.98      1977

accuracy
0.9843196762771876
********************
GradientBoostingClassifier:
********************
R-squared
0.9843203609701071
Error
0.12521836538580447
classification Report
              precision    recall  f1-score   support

           0       0.98      1.00      0.99      1946
           1       0.00      0.00      0.00        31

    accuracy                           0.98      1977
   macro avg       0.49      0.50      0.50      1977
weighted avg       0.97      0.98      0.98      1977

accuracy
0.9843196762771876
********************
[08:35:53] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the
default evaluation metric used with the objective 'binary:logistic' was changed
```

```
from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore
the old behavior.
[08:35:54] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the
default evaluation metric used with the objective 'binary:logistic' was changed
from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore
the old behavior.
[08:35:55] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the
default evaluation metric used with the objective 'binary:logistic' was changed
from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore
the old behavior.
[08:35:55] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the
default evaluation metric used with the objective 'binary:logistic' was changed
from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore
the old behavior.
[08:35:56] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the
default evaluation metric used with the objective 'binary:logistic' was changed
from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore
the old behavior.
[08:35:56] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the
default evaluation metric used with the objective 'binary:logistic' was changed
from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore
the old behavior.
[08:35:57] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the
default evaluation metric used with the objective 'binary:logistic' was changed
from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore
the old behavior.
[08:35:57] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the
default evaluation metric used with the objective 'binary:logistic' was changed
from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore
the old behavior.
[08:35:58] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the
default evaluation metric used with the objective 'binary:logistic' was changed
from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore
the old behavior.
[08:35:58] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the
default evaluation metric used with the objective 'binary:logistic' was changed
from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore
the old behavior.
[08:35:58] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the
default evaluation metric used with the objective 'binary:logistic' was changed
from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore
the old behavior.
[08:35:59] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the
default evaluation metric used with the objective 'binary:logistic' was changed
from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore
the old behavior.
[08:35:59] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the
default evaluation metric used with the objective 'binary:logistic' was changed
```

```
from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore
the old behavior.
[08:36:00] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the
default evaluation metric used with the objective 'binary:logistic' was changed
from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore
the old behavior.
[08:36:00] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the
default evaluation metric used with the objective 'binary:logistic' was changed
from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore
the old behavior.
[08:36:01] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the
default evaluation metric used with the objective 'binary:logistic' was changed
from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore
the old behavior.
[08:36:01] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the
default evaluation metric used with the objective 'binary:logistic' was changed
from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore
the old behavior.
[08:36:02] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the
default evaluation metric used with the objective 'binary:logistic' was changed
from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore
the old behavior.
[08:36:02] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the
default evaluation metric used with the objective 'binary:logistic' was changed
from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore
the old behavior.
[08:36:02] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the
default evaluation metric used with the objective 'binary:logistic' was changed
from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore
the old behavior.
XGBClassifier:
********************
R-squared
0.9843203609701071
Error
0.12521836538580447
classification Report
              precision    recall  f1-score   support

           0       0.98      1.00      0.99      1946
           1       0.00      0.00      0.00        31

    accuracy                           0.98      1977
   macro avg       0.49      0.50      0.50      1977
weighted avg       0.97      0.98      0.98      1977

accuracy
0.9843196762771876
```

****