Stacking Classifier using Natural Language Processing to build a Fake News Predictor model

NAME : SHRIYA CHOWDHURY

PYTHON CODE :

```
from google.colab import drive
drive.mount('/content/drive')
import os
import pandas as pd
news_dataset=pd.read_csv('/content/drive/MyDrive/train (1).csv')
import numpy as np
import pandas as pd
import re
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import SVC, LinearSVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score
from mlxtend.plotting import plot_confusion_matrix
from mlxtend.classifier import StackingClassifier

import nltk
nltk.download('stopwords')
print(stopwords.words('english')) #to be removed from dataset later
```

DATA PREPROCESSING

```
news_dataset.shape
#print first 5 rows of the dataframe
news_dataset.head()
#counting the number of missing values in the dataset
news_dataset.isnull().sum()
#replacing the missing values with empty string
news_dataset=news_dataset.fillna('')

#merging the title column and author name column
news_dataset['content']=news_dataset['author']+'
'+news_dataset['title']
print(news_dataset['content'])
```

STEMMING : process of reducing a word to its root by trimming the prefix and suffix

```
port_stem=PorterStemmer()
def stemming(content):
    stemmed_content=re.sub('[^a-zA-Z]',' ',content)# replace any
character other than alphabet with space
    stemmed_content=stemmed_content.lower() #convert all words the
small letters
    stemmed_content=stemmed_content.split() #words spliited at spaces
and stored as strings in a list
    stemmed_content=[port_stem.stem(word) for word in stemmed_content
if not word in stopwords.words('english')] #choose only the words that
are not stopwords
    stemmed_content=' '.join(stemmed_content)
    return stemmed_content

news_dataset['content']=news_dataset['content'].apply(stemming)

print(news_dataset['content'])
#separating the data and label
x=news_dataset['content'].values
y=news_dataset['label'].values
print(x)
print(y)
y.shape
```

Converting the textual into numerical data using nltk vectorizer

```
#converting the textual data to numerical data
vectorizer=TfidfVectorizer()
vectorizer.fit(x)
x=vectorizer.transform(x)

print(x)
```

splitting the dataset to training and text data

```
xtrain,xtest,ytrain,ytest=train_test_split(x, y, test_size=0.2,
stratify=y, random_state=2)
```

training the model : 5 classifier models

```
clf1 = KNeighborsClassifier(n_neighbors=2)
clf2 = RandomForestClassifier(n_estimators = 2,random_state=0)
clf3 = SVC(kernel = 'linear', random_state = 0, probability=True)
```

```
clf4 = GaussianNB()
clf5 = LogisticRegression()

clf1.fit(xtrain, ytrain)
clf2.fit(xtrain, ytrain)
clf3.fit(xtrain, ytrain)
clf4.fit(xtrain.toarray(), ytrain)
clf5.fit(xtrain, ytrain)
```

evaluation and accuracy score of each classifier model

```
xtest_pred1=clf1.predict(xtest)
testing_data_acc1=accuracy_score(xtest_pred1, ytest)

xtest_pred2=clf2.predict(xtest)
testing_data_acc2=accuracy_score(xtest_pred2, ytest)

xtest_pred3=clf3.predict(xtest.toarray())
testing_data_acc3=accuracy_score(xtest_pred3, ytest)

xtest_pred4=clf4.predict(xtest.toarray())
testing_data_acc4=accuracy_score(xtest_pred4, ytest)

xtest_pred5=clf5.predict(xtest)
testing_data_acc5=accuracy_score(xtest_pred5, ytest)

print(testing_data_acc1, testing_data_acc2, testing_data_acc3,
testing_data_acc4, testing_data_acc5)
```

Meta Stacking Classifier where the super classification model is logistic regression

```
lr = LogisticRegression()  # defining meta-classifier
clf_stack = StackingClassifier(classifiers =[clf1, clf2, clf3, clf4,
clf5], meta_classifier = lr, use_probas = True,
use_features_in_secondary = True)
```

Training the stacking classifier

```
model_stack = clf_stack.fit(xtrain.toarray(), ytrain)   # training of
stacked model
pred_stack = model_stack.predict(xtest)
```

evaluation and accuracy score of stacking classifier

```
acc_stack = accuracy_score(ytest, pred_stack)  # evaluating accuracy
```

```
print('accuracy score of Stacked model:', acc_stack * 100)
```

making a predictive system

```
xnew=xtest[0]
print(xnew)

prediction=model_stack.predict(xnew)
print(prediction)
if (prediction[0]==0):
    print('the news is real')
else:
    print('the news is fake')
```

RESULTS :

1. **Testing data accuracy after training the k-nearest neighbours classifier model (classifier1) = 58.7019 %**
2. **Testing data accuracy after training the k-nearest Random Forest Classifier model (classifier 2)= 94.0385 %**
3. **Testing data accuracy after training the Support Vector Classifier model (classifier 3) = 99.1346 %**
4. **Testing data accuracy after training the Gaussian Naïve Baye's Probability model (classifier 4) = 80.3846 %**
5. **Testing data accuracy after training the Logistic Regression Classifier model (classifier 5) = 97.9086 %**
6. **Accuracy of the Stacked Classifier model = 98.3173 %**

SCREENSHOT OF CODE AND OUTPUT :

fake news prediction.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share S

+ Code + Text

RAM Disk

```
from google.colab import drive
drive.mount('/content/drive')
import os
import pandas as pd
news_dataset=pd.read_csv('/content/drive/MyDrive/train (1).csv')
```

```
Mounted at /content/drive
```

```
[3] import numpy as np
import pandas as pd
import re
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import SVC, LinearSVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score
from mlxtend.plotting import plot_confusion_matrix
from mlxtend.classifier import StackingClassifier
```

```
[4] import nltk
nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
```

fake news prediction.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share S

+ Code + Text

RAM Disk

[4] import nltk
nltk.download('stopwords')

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
True
```

print(stopwords.words('english')) #to be removed from dataset later

```
print(stopwords.words('english')) #to be removed from dataset later
```

```
['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're", "you've", "you'll", "you'd", 'your', 'yours', 'yourself', 'yourselves', 'he', 'him', 'his', 'himself', 's
```

```
[6] news_dataset.shape
```

```
(20800, 5)
```

```
[7] #print first 5 rows of the dataframe
news_dataset.head()
```

	id	title	author	text	label
0	0	House Dem Aide: We Didn't Even See Comey's Let...	Darrell Lucus	House Dem Aide: We Didn't Even See Comey's Let...	1
1	1	FLYNN: Hillary Clinton, Big Woman on Campus - ...	Daniel J. Flynn	Ever get the feeling your life circles the rou...	0
2	2	Why the Truth Might Get You Fired	Consortiumnews.com	Why the Truth Might Get You Fired October 29, ...	1

fake news prediction.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share S

+ Code + Text

RAM Disk

```
#counting the number of missing values in the dataset
news_dataset.isnull().sum()
```

```
id          0
title     558
author   1957
text       39
label       0
dtype: int64
```

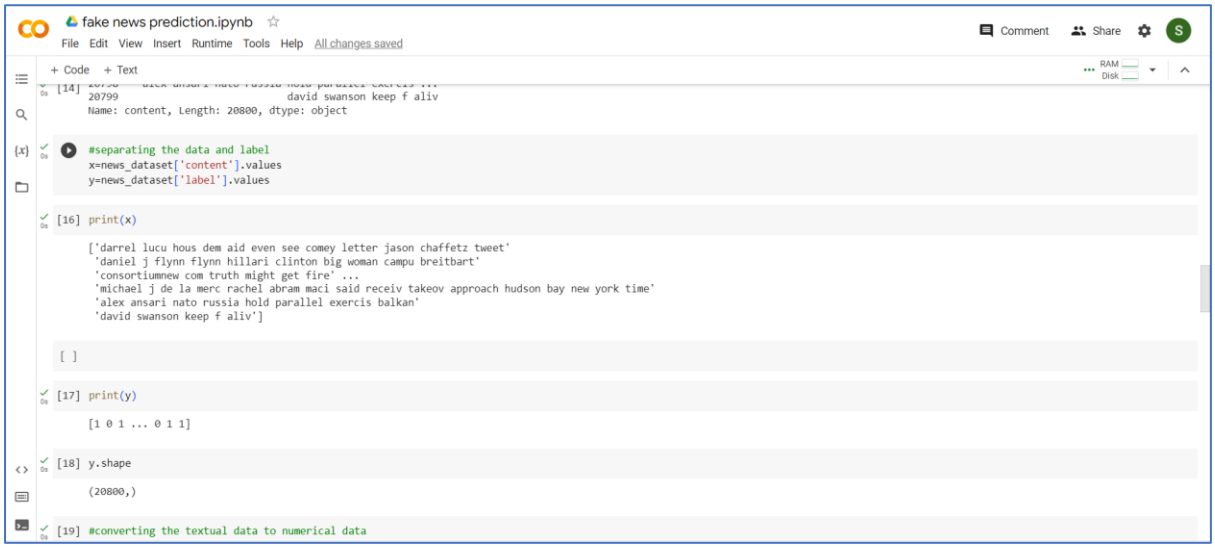
```
[9] #replacing the missing values with empty string
news_dataset=news_dataset.fillna('')
```

```
[10] #merging the title column and author name column
news_dataset['content']=news_dataset['author']+' '+news_dataset['title']
```

```
[11] print(news_dataset['content'])
```

```
0        Darrell Lucus House Dem Aide: We Didn't Even S...
1        Daniel J. Flynn FLYNN: Hillary Clinton, Big Wo...
2        Consortiumnews.com Why the Truth Might Get You...
3        Jessica Purkiss 15 Civilians Killed In Single ...
4        Howard Portnoy Iranian woman jailed for fictio...
                               ...
20795    Jerome Hudson Rapper T.I.: Trump a 'Poster Chi...
20796    Benjamin Hoffman N.F.L. Playoffs: Schedule, Ma...
20797    Michael J. de la Merced and Rachel Abrams Macy...
20798    Alex Ansary NATO, Russia To Hold Parallel Exer...
20799            David Swanson What Keeps the F-35 Alive
```


fake news prediction.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Name: content, Length: 20800, dtype: object
port_stem=PorterStemmer()
def stemming(content):
stemmed_content=re.sub('[^a-zA-Z]',' ',content)# replace any character other than alphabet with space
stemmed_content=stemmed_content.lower() #convert all words the small letters
stemmed_content=stemmed_content.split() #words spliited at spaces and stored as strings in a list
stemmed_content=[port_stem.stem(word) for word in stemmed_content if not word in stopwords.words('english')] #choose only the words that are not stopwords
stemmed_content=' '.join(stemmed_content)
return stemmed_content
[13] news_dataset['content']=news_dataset['content'].apply(stemming)
[14] print(news_dataset['content'])
0 darrel lucu hous dem aid even see comey letter...
1 daniel j flynn flynn hillari clinton big woman...
2 consortiumnew com truth might get fire
3 jessica purkiss civilian kill singl us airstri...
4 howard portnoy iranian woman jail fiction unpu...
...
20795 jerom hudson rapper trump poster child white s...
20796 benjamin hoffman n f l playoff schedul matchup...
20797 michael j de la merc rachel abram maci said re...
20798 alex ansari nato russia hold parallel exercis ...
20799 david swanson keep f aliv
Name: content, Length: 20800, dtype: object


fake news prediction.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
20799 david swanson keep f aliv
Name: content, Length: 20800, dtype: object
#separating the data and label
x=news_dataset['content'].values
y=news_dataset['label'].values
[16] print(x)
['darrel lucu hous dem aid even see comey letter jason chaffetz tweet'
'daniel j flynn flynn hillari clinton big woman campu breitbart'
'consortiumnew com truth might get fire' ...
'michael j de la merc rachel abram maci said receiv takeov approach hudson bay new york time'
'alex ansari nato russia hold parallel exercis balkan'
'david swanson keep f aliv']
[ ]
[17] print(y)
[1 0 1 ... 0 1 1]
[18] y.shape
(20800,)
[19] #converting the textual data to numerical data


fake news prediction.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
[19] #converting the textual data to numerical data
vectorizer=TfidfVectorizer()
vectorizer.fit(x)
x=vectorizer.transform(x)
print(x)
(0, 15686) 0.28485063562728646
(0, 13473) 0.2565896679337957
(0, 8909) 0.3635963806326075
(0, 8630) 0.29212514087043684
(0, 7692) 0.24785219520671603
(0, 7005) 0.21874169089359144
(0, 4973) 0.233316966909351
(0, 3792) 0.2705332480845492
(0, 3600) 0.3598939188262559
(0, 2959) 0.2468450128533713
(0, 2483) 0.3676519686797209
(0, 267) 0.27010124977708766
(1, 16799) 0.30071745655510157
(1, 6816) 0.1904660198296849
(1, 5503) 0.7143299355715573
(1, 3568) 0.26373768806048464
(1, 2813) 0.19094574062359204
(1, 2223) 0.3827320386859759
(1, 1894) 0.15521974226349364
(1, 1497) 0.2939891562094648
(2, 15611) 0.41544962664721613
(2, 9620) 0.49351492943649944


fake news prediction.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
(20798, 350) 0.28446937819072576
(20799, 14852) 0.5677577267055112
(20799, 8036) 0.45983893273780013
(20799, 3623) 0.37927626273066584
(20799, 377) 0.5677577267055112
[21] xtrain,xtest,ytrain,ytest=train_test_split(x, y, test_size=0.2, stratify=y, random_state=2)
clf1 = KNeighborsClassifier(n_neighbors=2)
clf2 = RandomForestClassifier(n_estimators = 2,random_state=0)
clf3 = SVC(kernel = 'linear', random_state = 0, probability=True)
clf4 = GaussianNB()
clf5 = LogisticRegression()
[23] clf1.fit(xtrain, ytrain)
clf2.fit(xtrain, ytrain)
clf3.fit(xtrain, ytrain)
clf4.fit(xtrain.toarray(), ytrain)
clf5.fit(xtrain, ytrain)
LogisticRegression
LogisticRegression()
[28] xtest_pred1=clf1.predict(xtest)
testing_data_acc1=accuracy_score(xtest_pred1, ytest)
xtest_pred2=clf2.predict(xtest)


fake news prediction.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
[23] LogisticRegression
LogisticRegression()
xtest_pred1=clf1.predict(xtest)
testing_data_acc1=accuracy_score(xtest_pred1, ytest)
xtest_pred2=clf2.predict(xtest)
testing_data_acc2=accuracy_score(xtest_pred2, ytest)
xtest_pred3=clf3.predict(xtest.toarray())
testing_data_acc3=accuracy_score(xtest_pred3, ytest)
xtest_pred4=clf4.predict(xtest.toarray())
testing_data_acc4=accuracy_score(xtest_pred4, ytest)
xtest_pred5=clf5.predict(xtest)
testing_data_acc5=accuracy_score(xtest_pred5, ytest)
[29] print(testing_data_acc1, testing_data_acc2, testing_data_acc3, testing_data_acc4, testing_data_acc5)
0.5870192307692308 0.9403846153846154 0.9913461538461539 0.8038461538461539 0.9790865384615385
[30] lr = LogisticRegression() # defining meta-classifier
clf_stack = StackingClassifier(classifiers =[clf1, clf2, clf3, clf4, clf5], meta_classifier = lr, use_probas = True, use_features_in_secondary = True)


fake news prediction.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
xtest_pred3=clf3.predict(xtest.toarray())
testing_data_acc3=accuracy_score(xtest_pred3, ytest)
xtest_pred4=clf4.predict(xtest.toarray())
testing_data_acc4=accuracy_score(xtest_pred4, ytest)
xtest_pred5=clf5.predict(xtest)
testing_data_acc5=accuracy_score(xtest_pred5, ytest)
[29] print(testing_data_acc1, testing_data_acc2, testing_data_acc3, testing_data_acc4, testing_data_acc5)
0.5870192307692308 0.9403846153846154 0.9913461538461539 0.8038461538461539 0.9790865384615385
[30] lr = LogisticRegression() # defining meta-classifier
clf_stack = StackingClassifier(classifiers =[clf1, clf2, clf3, clf4, clf5], meta_classifier = lr, use_probas = True, use_features_in_secondary = True)
[32] model_stack = clf_stack.fit(xtrain.toarray(), ytrain) # training of stacked model
pred_stack = model_stack.predict(xtest.toarray())
acc_stack = accuracy_score(ytest, pred_stack) # evaluating accuracy
print('accuracy score of Stacked model:', acc_stack * 100)
accuracy score of Stacked model: 98.3173076923077
0s completed at 1:03 AM

make a predictive system for test dataset

```
xnew=xtest[0]
print(xnew)

prediction=model_stack.predict(xnew)
print(prediction)
if (prediction[0]==0):
    print('the news is real')
else:
    print('the news is fake')
```

```
  (0, 12801)    0.2910746804557067
  (0, 9818)     0.30786004182651133
  (0, 7668)     0.22945314906455008
  (0, 6816)     0.16094563145945953
  (0, 6289)     0.288254092437116
  (0, 5941)     0.288254092437116
  (0, 5233)     0.21316265672448448
  (0, 4346)     0.3250084367199054
  (0, 3395)     0.3301936745912874
  (0, 2959)     0.24534646237198773
  (0, 1667)     0.30373060380734146
  (0, 908)      0.213510750423647
  (0, 239)      0.34297808354766485
[1]
the news is fake
```