Fake News

November 15, 2021

1 Creating an Accurate Fake News Detection Model

[1]:
```
# Packages
import numpy as np
import pandas as pd
import itertools
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import PassiveAggressiveClassifier
from sklearn.metrics import accuracy_score, confusion_matrix
```

Getting our data

[2]:
```
data = pd.read_csv('news.csv')
```

Looking at our shape and our data

[3]:
```
data.shape
```

[3]:
```
(6335, 4)
```

[4]:
```
data.head()
```

[4]:
```
   Unnamed: 0                                              title  \
0        8476                       You Can Smell Hillary's Fear
1       10294  Watch The Exact Moment Paul Ryan Committed Pol…
2        3608        Kerry to go to Paris in gesture of sympathy
3       10142  Bernie supporters on Twitter erupt in anger ag…
4         875   The Battle of New York: Why This Primary Matters

                                                text label
0  Daniel Greenfield, a Shillman Journalism Fello…  FAKE
1  Google Pinterest Digg Linkedin Reddit Stumbleu…  FAKE
2  U.S. Secretary of State John F. Kerry said Mon…  REAL
3  - Kaydee King (@KaydeeKing) November 9, 2016 T…  FAKE
4  It's primary day in New York and front-runners…  REAL
```

[5]:
```
labels = data.label
labels.head()
```

[5]:
```
0    FAKE
1    FAKE
2    REAL
3    FAKE
4    REAL
Name: label, dtype: object
```

Split our data into training and testing sets

[6]:
```
x_train,x_test,y_train,y_test = train_test_split(data['text'],
                                                 labels,
                                                 test_size = 0.2,
                                                 random_state = 7)
```

Initialize a TfidfVectorizer with stop words

[7]:
```
vectorizer = TfidfVectorizer(stop_words = 'english', max_df = 0.7)
```

Fit and Transform training set and Transform the testing set

[8]:
```
tfidf_train = vectorizer.fit_transform(x_train)
tfidf_test = vectorizer.transform(x_test)
```

Initializing our PassiveAgressiveClassifier

[9]:
```
PAC = PassiveAggressiveClassifier(max_iter = 50)
PAC.fit(tfidf_train, y_train)
```

[9]:
```
PassiveAggressiveClassifier(max_iter=50)
```

Predict on our testing set

[10]:
```
y_pred = PAC.predict(tfidf_test)
```

Calculate the accuracy

[11]:
```
score = accuracy_score(y_test, y_pred)

print(f'Accuracy: {round(score*100,2)}%')
```

```
Accuracy: 92.98%
```

Building confusion matrix

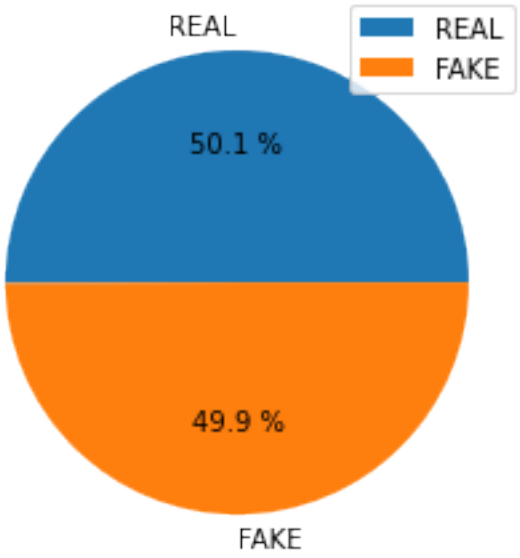
[12]:
```
confusion_matrix(y_test, y_pred, labels = ['FAKE', 'REAL'])
```

[12]:
```
array([[589,  49],
       [ 40, 589]])
```

We have 587 True positives, 588 True Negative, 41 False Positives and 51 False Negatives

[13]:
```
data['label'].value_counts(normalize=True).plot.pie(autopct='%.1f %%',␣
↪ylabel='', legend=True)
```

[13]: <AxesSubplot:>



[16]:
```
import pandas as pd
import matplotlib.pyplot as plt
from wordcloud import WordCloud
```

[18]:
```
text = " ".join(news for news in data.text)
```

[19]:
```
word_cloud = WordCloud(collocations = False, background_color = 'white').
 ↪generate(text)
```

[22]:
```
plt.imshow(word_cloud, interpolation='bilinear')
plt.axis("off")
plt.show()
```

[25]:
```
fake = data[data['label'] == "FAKE"]
fakenews = " ".join(news for news in fake.text)
```

[26]:
```
fake_word_cloud = WordCloud(collocations = False, background_color = 'white').
↪generate(fakenews)
```

[27]:
```
plt.imshow(fake_word_cloud, interpolation='bilinear')
plt.axis("off")
plt.show()
```

[ ]: