fake-news

June 27, 2024

[20]:
```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split as ttp
from sklearn.metrics import classification_report
import re
import string
import matplotlib.pyplot as plt
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression, LinearRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
```

[2]:
```
data_true = pd.read_csv('/content/drive/MyDrive/True.csv')
data_fake = pd.read_csv('/content/drive/MyDrive/Fake.csv')
```

[3]:
```
data_true['label'] = 1
data_fake['label'] = 0
```

[4]:
```
data_true_manual_testing = data_true.tail(10)
data_true = data_true.iloc[:-10]
```

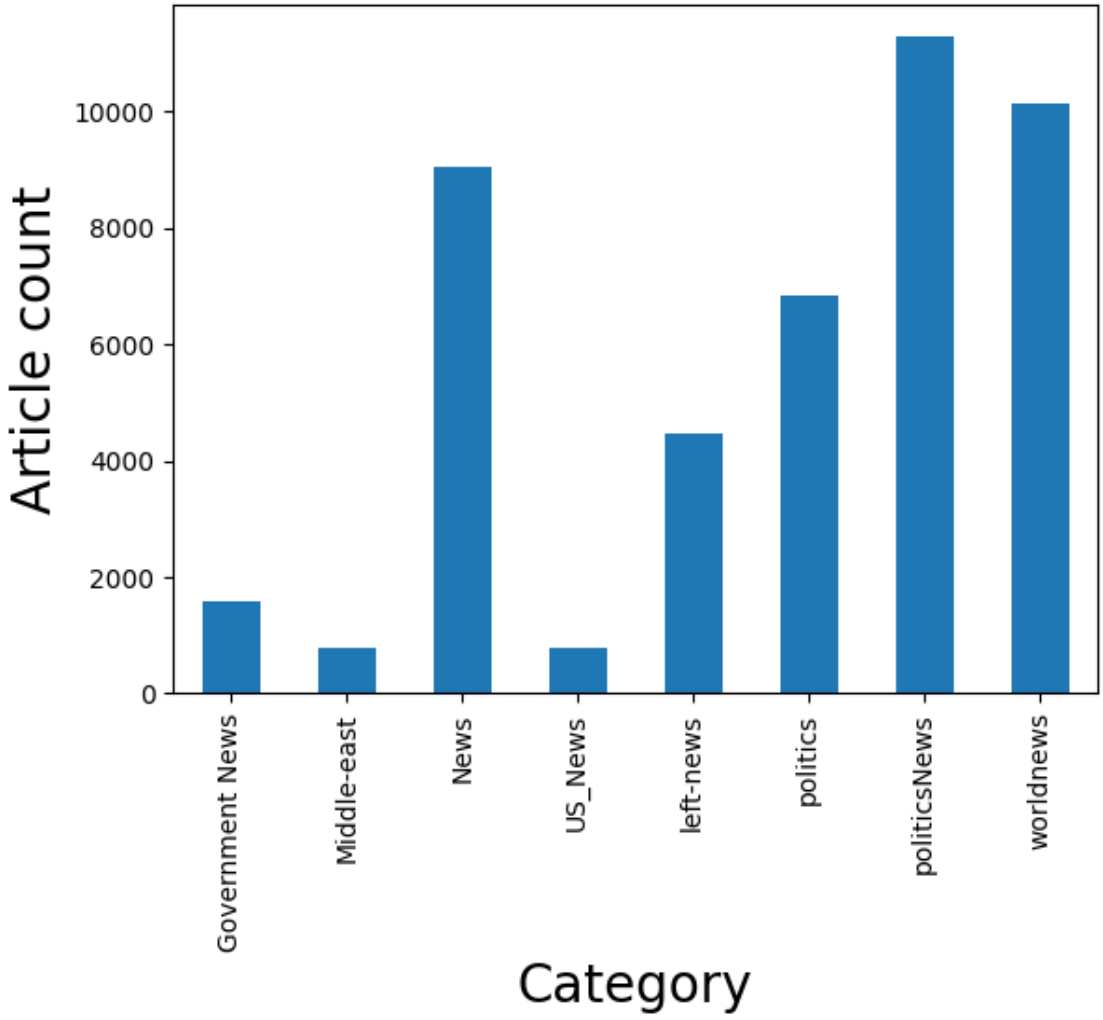
[5]:
```
data_fake_manual_testing = data_fake.tail(10)
data_fake = data_fake.iloc[:-10]
```

[6]:
```
data_manual_testing = pd.concat([data_true_manual_testing,␣
 ↪data_fake_manual_testing], axis=0)
data_manual_testing.to_csv('manual_testing.csv', index=False)
```

[7]:
```
data_merge = pd.concat([data_true, data_fake], axis=0)
```

[8]:
```
print(data_merge.groupby(['subject'])['text'].count())
data_merge.groupby(['subject'])['text'].count().plot(kind='bar')
plt.xlabel("Category", size=20)
plt.ylabel("Article count", size=20)
plt.show()
```

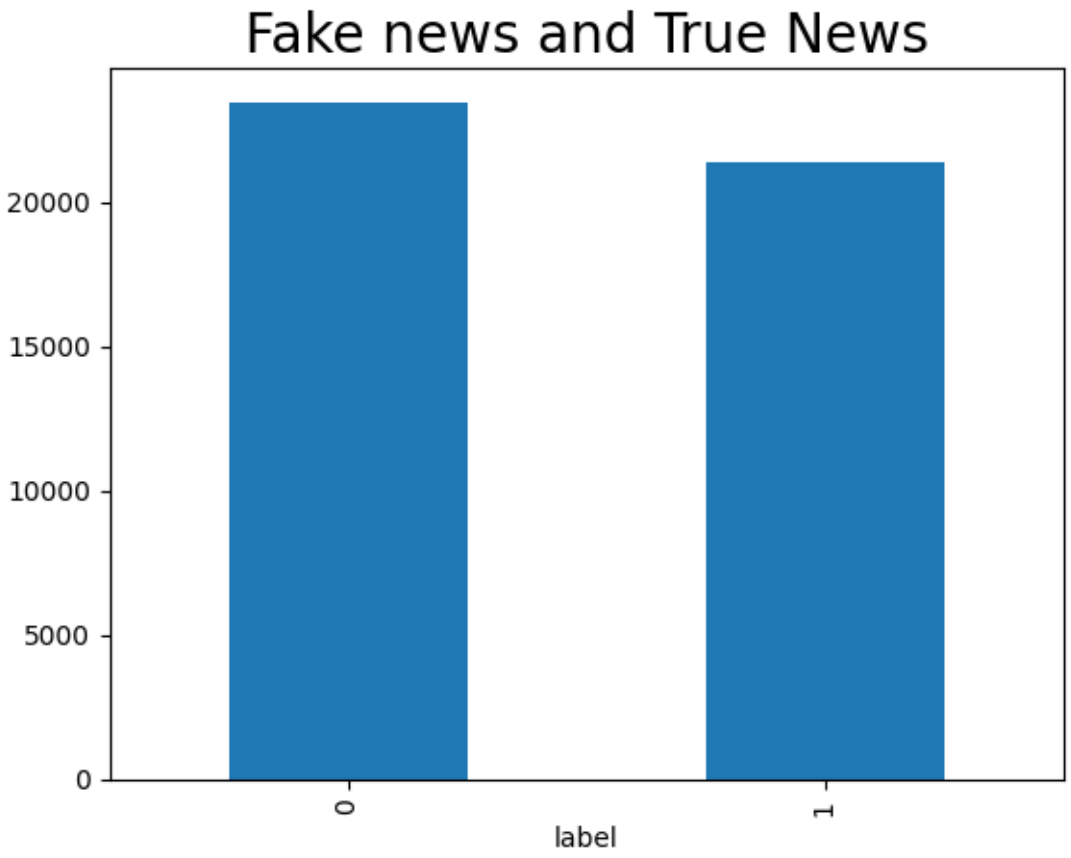
```
subject
Government News     1570
Middle-east          768
News                9050
US_News              783
left-news           4459
politics            6841
politicsNews       11272
worldnews          10135
Name: text, dtype: int64
```



[9]:
```
print(data_merge.groupby(['label'])['text'].count())
print("0 = Fake news\n1 = True news")
data_merge.groupby(['label'])['text'].count().plot(kind='bar')
plt.title("Fake news and True News", size=20)
```

```
plt.show()
```

```
label
0    23471
1    21407
Name: text, dtype: int64
0 = Fake news
1 = True news
```



[10]:
```
data = data_merge.drop(['title', 'subject', 'date'], axis=1)
data = data.sample(frac=1).reset_index(drop=True)
```

[11]:
```
print(data.isnull().sum())
```

```
text     0
label    0
dtype: int64
```

[12]:
```
def filtering(text):
    text = text.lower()
```

```
    text = re.sub('\[.*?\]', ' ', text)
    text = re.sub("\\W", " ", text)
    text = re.sub('https?://\S+|www\.\S+', ' ', text)
    text = re.sub('<.*?>+', ' ', text)
    text = re.sub('[%s]' % re.escape(string.punctuation), ' ', text)
    text = re.sub('\n', ' ', text)
    text = re.sub('\w*\d\w*', ' ', text)
    return text
```

[13]:
```
data['text'] = data['text'].apply(filtering)
```

[14]:
```
vectorizer = TfidfVectorizer()
x = vectorizer.fit_transform(data['text'])
y = data['label']
```

[15]:
```
x_train, x_test, y_train, y_test = ttp(x, y, test_size=0.25, random_state=0)
```

[16]:
```
LR = LogisticRegression(max_iter=1000)
LR.fit(x_train, y_train)
```

[16]:
```
LogisticRegression(max_iter=1000)
```

[17]:
```
DT = DecisionTreeClassifier()
DT.fit(x_train, y_train)
```

[17]:
```
DecisionTreeClassifier()
```

[18]:
```
RF = RandomForestClassifier()
RF.fit(x_train, y_train)
```

[18]:
```
RandomForestClassifier()
```

[19]:
```
LinR = LinearRegression()
LinR.fit(x_train, y_train)
```

[19]:
```
LinearRegression()
```

[21]:
```
KNN = KNeighborsClassifier()
KNN.fit(x_train, y_train)
```

[21]:
```
KNeighborsClassifier()
```

[22]:
```
user_input = input("Enter the news article text: ")
user_input_transformed = vectorizer.transform([user_input])
```

```
Enter the news article text: Papadopoulos pleaded guilty to lying to the F.B.I
```

[30]:
```
prediction_LR = LR.predict(user_input_transformed)
if prediction_LR == 1:
    print("Logistic Regression: This news article is real.")
else:
    print("Logistic Regression: This news article is fake.")

prediction_DT = DT.predict(user_input_transformed)
if prediction_DT == 1:
    print("Decision Tree: This news article is real.")
else:
    print("Decision Tree: This news article is fake.")

prediction_RF = RF.predict(user_input_transformed)
if prediction_RF == 1:
    print("Random Forest: This news article is real.")
else:
    print("Random Forest: This news article is fake.")

prediction_LinR = LinR.predict(user_input_transformed)

prediction_LinR = np.where(prediction_LinR >= 0.5, 1, 0)
if prediction_LinR == 1:
    print("Linear Regression: This news article is real.")
else:
    print("Linear Regression: This news article is fake.")

prediction_KNN = KNN.predict(user_input_transformed)
if prediction_KNN == 1:
    print("K-Nearest Neighbors: This news article is real.")
else:
    print("K-Nearest Neighbors: This news article is fake.")
```

```
Logistic Regression: This news article is fake.
Decision Tree: This news article is fake.
Random Forest: This news article is fake.
Linear Regression: This news article is fake.
K-Nearest Neighbors: This news article is fake.
```