spam-email-filter-1

January 15, 2024

1 Spam Email Filter using NLP and Machine Learning Algorithm

1.0.1 Task:- Build a spam filter using NLP and machine learning to identify and filter out spam emails

- Develop a robust Spam Email Filter using Natural Language Processing (NLP) techniques and machine learning algorithms.
- The goal is to create an intelligent system capable of accurately classifying emails as either spam or legitimate (ham) based on their content and linguistic features

1.1 Importing necessary libraries

[1]:
```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score, classification_report
from sklearn.pipeline import Pipeline
```

[2]:
```
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer
import joblib
```

1.2 Loading and Exploring the Dataset

[3]:
```
df = pd.read_csv('emails.csv')
df.head()
```

[3]:
```
                                                text  spam
0  Subject: naturally irresistible your corporate…     1
1  Subject: the stock trading gunslinger  fanny i…     1
2  Subject: unbelievable new homes made easy  im …     1
3  Subject: 4 color printing special  request add…     1
4  Subject: do not have money , get software cds …     1
```

[4]:
```
df.shape
```

[4]: (5728, 2)

[5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5728 entries, 0 to 5727
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   text    5728 non-null   object
 1   spam    5728 non-null   int64
dtypes: int64(1), object(1)
memory usage: 89.6+ KB
```

[6]:
```
df.groupby('spam').describe()
```

[6]:
```
      text
     count unique                                                top freq
spam
0     4360   4327  Subject: * special notification * aurora versi…    2
1     1368   1368  Subject: naturally irresistible your corporate…    1
```

1.3 Data Preprocessing

[7]:
```
import nltk
nltk.download('stopwords')
nltk.download('punkt')
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\suman\AppData\Roaming\nltk_data…
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\suman\AppData\Roaming\nltk_data…
[nltk_data]   Package punkt is already up-to-date!
```

[7]: True

[8]:
```
stop_words = set(stopwords.words('english'))
ps = PorterStemmer()
```

[9]:
```
def preprocess_text(text):
    words = word_tokenize(text)
    words = [ps.stem(word) for word in words if word.isalpha() and word.lower()␣
 ↪not in stop_words]

    return ' '.join(words)
df['processed_text'] = df['text'].apply(preprocess_text)
```

[10]:
```
df.sample(5)
```

[10]:
```
                                                   text  spam  \
1761  Subject: internal var / credit candidate : ami…     0
3361  Subject: making room for " summer interns "  h…     0
213   Subject: high growth investing for tomorrow  h…     1
2778  Subject: re : new risk management book  vince …     0
5181  Subject: re : good morning  john ,  it does no…     0

                                         processed_text
1761  subject intern var credit candid amit bartarya…
3361  subject make room summer intern hello good new…
213   subject high growth invest tomorrow high growt…
2778  subject new risk manag book vinc thank much pe…
5181  subject good morn john sound silli get mani op…
```

1.4 Training the Model

[11]:
```
X = df['processed_text']
y = df['spam']
```

[12]:
```
# Splitting the Dataset
x_train, x_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
↪random_state=42)
```

[13]:
```
len(x_train), len(y_train)
```

[13]:
```
(4582, 4582)
```

[14]:
```
len(x_test), len(y_test)
```

[14]:
```
(1146, 1146)
```

[15]:
```
# Building the Machine Learning Pipeline
model = Pipeline([
    ('vectorizer', CountVectorizer()),
    ('nb', MultinomialNB())
])
```

[16]:
```
model.fit(x_train, y_train)
```

[16]:
```
Pipeline(steps=[('vectorizer', CountVectorizer()), ('nb', MultinomialNB())])
```

[17]:
```
model.score(x_test, y_test)
```

[17]:
```
0.9860383944153578
```

1.5 Evaluating the Model

[18]:
```
y_test.head()
```

[18]:
```
4445    0
4118    0
3893    0
4210    0
5603    0
Name: spam, dtype: int64
```

[19]:
```
y_pred = model.predict(x_test)
y_pred
```

[19]:
```
array([0, 0, 0, …, 1, 0, 0], dtype=int64)
```

[20]:
```
print(f'Accuracy of the model: {accuracy_score(y_test, y_pred)}')
```

```
Accuracy of the model: 0.9860383944153578
```

[21]:
```
print('Classification Report:\n')
print(classification_report(y_test, y_pred))
```

```
Classification Report:

              precision    recall  f1-score   support

           0       0.99      0.99      0.99       856
           1       0.98      0.97      0.97       290

    accuracy                           0.99      1146
   macro avg       0.98      0.98      0.98      1146
weighted avg       0.99      0.99      0.99      1146
```

[22]:
```
# Using Cross-validation to assess generalizability
from sklearn.model_selection import cross_val_score
cv_score = cross_val_score(model, X, y, cv=5)
```

[23]:
```
print(f'Cross-validation Scores: {cv_score}')
```

```
Cross-validation Scores: [0.9877836  0.9904014  0.9921466  0.99039301
0.99388646]
```

[24]:
```
print('Mean CV Score:', cv_score.mean())
```

```
Mean CV Score: 0.9909222128230336
```

1.5.1 Example Mail

[25]:
```python
new_email = ["Congratulations! You've won a free vacation. Click here to claim␣
↪your prize."]
prediction = model.predict(new_email)
```

[26]:
```python
if prediction[0] == 1:
    print('The email is classified as "ham" (non-spam).')
elif prediction[0] == 0:
    print('The email is classified as "spam".')
else:
    print('Invalid prediction label.')
```

```
The email is classified as "ham" (non-spam).
```

[27]:
```python
probability_spam = model.predict_proba(new_email)[0][0]
probability_ham = model.predict_proba(new_email)[0][1]

print(f"Spam Probability: {probability_spam:.2f}")
print(f"Ham Probability: {probability_ham:.2f}")
```

```
Spam Probability: 0.00
Ham Probability: 1.00
```

[ ]:
```python

```