Spam Email Detecion With Machine Learning

Conclusion:

> About This Project

In this project, we developed a spam email detection system using machine learning techniques. The objective was to accurately categorize incoming emails as either spam or legitimate (ham) messages. The following steps were undertaken:

1. Data Preparation: We imported necessary libraries and loaded the dataset containing email messages and their corresponding categories. After addressing any missing values, we transformed the categorical labels into numerical values for modeling.
2. Feature Extraction: To facilitate model training, we utilized TF-IDF Vectorization. This process converted the textual content of emails into numerical features that the machine learning algorithm could understand.
3. Model Training and Evaluation: Logistic Regression was chosen as the classification algorithm. The dataset was divided into training and testing sets for model training and evaluation. The accuracy achieved on both training and testing sets was around 96.70% and 96.59% respectively, showcasing the model's ability to effectively distinguish between spam and legitimate emails.
4. Predictive System: We demonstrated the model's predictive capability by inputting an example email message. The model correctly classified it as legitimate (ham), highlighting its practical applicability.

In conclusion, this project highlights the efficacy of machine learning in addressing the challenge of spam email detection. While the achieved accuracy is commendable, continuous improvements could involve exploring more advanced algorithms, optimizing model parameters, and staying updated with evolving spamming tactics. The

project underscores the importance of email security in the digital age and showcases the potential of machine learning in enhancing it.

Importing Necessary Libraries

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
```

Loading Dataset

```
raw_mail_data = pd.read_csv('/content/mail_data.csv')
```

Data Exploration

```
print(raw_mail_data)
```

```
     Category                                            Message
0         ham  Go until jurong point, crazy.. Available only ...
1         ham                      Ok lar... Joking wif u oni...
2        spam  Free entry in 2 a wkly comp to win FA Cup fina...
3         ham  U dun say so early hor... U c already then say...
4         ham  Nah I don't think he goes to usf, he lives aro...
...       ...                                                ...
5567     spam  This is the 2nd time we have tried 2 contact u...
```

```
5568      ham               Will ü b going to esplanade fr home?
5569      ham  Pity, * was in mood for that. So...any other s...
5570      ham  The guy did some bitching but I acted like i'd...
5571      ham                         Rofl. Its true to its name

[5572 rows x 2 columns]
```

```
# replace the null Values
mail_data = raw_mail_data.where((pd.notnull(raw_mail_data)),'')
```

```
# first 5 rows
mail_data.head()
```

	Category	Message
0	ham	Go until jurong point, crazy.. Available only ...
1	ham	Ok lar... Joking wif u oni...
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...
3	ham	U dun say so early hor... U c already then say...
4	ham	Nah I don't think he goes to usf, he lives aro...

```
# Checking the size of Dataframe
mail_data.shape
```

```
(5572, 2)
```

Label Encoding Spam As 0 & Ham As 1

```
# spam mail as 0; ham mail as 1;
mail_data.loc[mail_data['Category'] == 'spam','Category',] = 0
```

```
mail_data.loc[mail_data['Category'] == 'ham','Category',] = 1


# separted the text and label

x = mail_data['Message']

y = mail_data['Category']


print(x)
```

```
     0       Go until jurong point, crazy.. Available only ...
     1                           Ok lar... Joking wif u oni...
     2       Free entry in 2 a wkly comp to win FA Cup fina...
     3       U dun say so early hor... U c already then say...
     4       Nah I don't think he goes to usf, he lives aro...
                                 ...
     5567    This is the 2nd time we have tried 2 contact u...
     5568                 Will ü b going to esplanade fr home?
     5569    Pity, * was in mood for that. So...any other s...
     5570    The guy did some bitching but I acted like i'd...
     5571                           Rofl. Its true to its name
     Name: Message, Length: 5572, dtype: object
```

```
print(y)
```

```
     0       1
     1       1
     2       0
     3       1
     4       1
            ..
     5567    0
     5568    1
     5569    1
     5570    1
     5571    1
     Name: Category, Length: 5572, dtype: object
```

Train and Test Data

```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=3)


print(x.shape)
print(x_train.shape)
print(x_test.shape)
```

```
(5572,)
(4457,)
(1115,)
```

Feature Extraction

```
# transform the text data to feature vectors to Logistic regration

feature_extraction = TfidfVectorizer(min_df = 1, stop_words='english', lowercase=True)

x_train_features = feature_extraction.fit_transform(x_train)
x_test_features = feature_extraction.transform(x_test)

# Convert y_train and y_test as int

y_train = y_train.astype('int')

y_test = y_test.astype('int')


print(x_train_features)
```

```
(0, 5413)     0.6198254967574347
(0, 4456)     0.4168658090846482
(0, 2224)     0.413103377943378
(0, 3811)     0.34780165336891333
(0, 2329)     0.38783870336935383
(1, 4080)     0.18880584110891163
(1, 3185)     0.29694482957694585
(1, 3325)     0.31610586766078863
(1, 2957)     0.3398297002864083
(1, 2746)     0.3398297002864083
(1, 918)      0.22871581159877646
(1, 1839)     0.2784903590561455
(1, 2758)     0.3226407885943799
(1, 2956)     0.33036995955537024
(1, 1991)     0.33036995955537024
(1, 3046)     0.2503712792613518
(1, 3811)     0.17419952275504033
(2, 407)      0.509272536051008
(2, 3156)     0.4107239318312698
(2, 2404)     0.45287711070606745
(2, 6601)     0.6056811524587518
(3, 2870)     0.5864269879324768
(3, 7414)     0.8100020912469564
(4, 50)       0.23633754072626942
(4, 5497)     0.15743785051118356
:     :
(4454, 4602)  0.2669765732445391
(4454, 3142)  0.32014451677763156
(4455, 2247)  0.37052851863170466
(4455, 2469)  0.35441545511837946
(4455, 5646)  0.33545678464631296
(4455, 6810)  0.29731757715898277
(4455, 6091)  0.23103841516927642
(4455, 7113)  0.30536590342067704
(4455, 3872)  0.3108911491788658
(4455, 4715)  0.30714144758811196
(4455, 6916)  0.19636985317119715
(4455, 3922)  0.31287563163368587
(4455, 4456)  0.24920025316220423
(4456, 141)   0.292943737785358
```

```
(4456, 647)   0.30133182431707617
(4456, 6311)  0.30133182431707617
(4456, 5569)  0.4619395404299172
(4456, 6028)  0.21034888000987115
(4456, 7154)  0.24083218452280053
(4456, 7150)  0.3677554681447669
(4456, 6249)  0.17573831794959716
(4456, 6307)  0.2752760476857975
(4456, 334)   0.2220077711654938
(4456, 5778)  0.16243064490100795
(4456, 2870)  0.31523196273113385
```

Tarin the Model

Logistic Regression

```
model = LogisticRegression()


# Training the LogisticRegreession with data
model.fit(x_train_features,y_train)
```

```
LogisticRegression
LogisticRegression()
```

Evaluating the trained model

```
# prediction on data
```

```
prediction_on_training_data = model.predict(x_train_features)
accuracy_on_training_data = accuracy_score(y_train,prediction_on_training_data)


print('Accuracy on training data: ', accuracy_on_training_data)
```

```
     Accuracy on training data:  0.9670181736594121
```

```
# prediction on test data

prediction_on_test_data = model.predict(x_test_features)
accuracy_on_test_data = accuracy_score(y_test,prediction_on_test_data)


print('Accuracy on test data: ', accuracy_on_test_data)
```

```
     Accuracy on test data:  0.9659192825112107
```

Building a predictive System

```
input_mail = ["Nah I don't think he goes to usf, he lives around here though"]

#convert text to feature

input_data_features = feature_extraction.transform(input_mail)

# making Prediction

prediction = model.predict(input_data_features)
print(prediction)


if (prediction[0]==1):
  print('Ham Mail')
```

```
else:
  print('Spam Mail')
```

```
[1]
Ham Mail
```