Spam or Ham mail Detection and Classification

Spam mail, or junk mail, is a type of email that is sent to a massive number of users at one time, frequently containing cryptic messages, scams, or most dangerously, phishing content.
While spam emails are sometimes sent manually by a human, most often, they are sent using a bot. Most popular email platforms, like Gmail and Microsoft Outlook, automatically filter spam emails by screening for recognizable phrases and patterns. A few common spam emails include fake advertisements, chain emails, and impersonation attempts. While these built-in spam detectors are usually pretty effective, sometimes, a particularly well-disguised spam email may fall through the cracks, landing in your inbox instead of your spam folder.
Clicking on a spam email can be dangerous, exposing your computer and personal information to different types of malware. Therefore, it's important to implement additional safety measures to protect your device, especially when it handles sensitive information like user data.

We use Python to build an email spam detector. Then, we'll use machine learning to train our spam detector to recognize and classify emails into spam and non-spam or ham.
We have explained step-by-step methods regarding the implementation of the Email spam detection and classification using machine learning algorithms in the Python programming language.

1. **Creating virtual environment in cmd**
   >>Conda create -n spam_env python=3.9

Here, spam_env= virtual environment

Then Activate new created venv(spam_env)
>>Activate spam_env

After Install python libraries with pip

- pip install pandas
- Pip install notebook
- pip install numpy
- pip install scikit-learn

Then launch jupyter notebook in cmd and jupyter open in browser for python.

2. **Import Python Libraries**

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
```

```
from sklearn.linear_model import LogisticRegression
from sklearn.feature_extraction.text import CountVectorizer

from sklearn.metrics import accuracy_score
```

3. **Loading dataset**
   >> mail= pd.read_csv("mail_data.csv")

4. **Understanding basic information of dataset**
   There are 5572 rows and 2 column

5. **Categorized dataset in Spam and ham**
   Categorized dataset into spam and ham from category columns as 0 and1.

6. **Define Variable**
   Define X for massage & Y for category variable

7. **Split dataset into test and train data(train_test_split)**
   We'll use a train-test split method to train our email spam detector to recognize and categorize spam emails. The train-test split is a technique for evaluating the performance of a machine learning algorithm. We can use it for either classification or regression of any supervised learning algorithm.

   - Train dataset: used to fit the machine learning model
   - Test dataset: used to evaluate the fit of the machine learning model

8. **Feature Extraction**

In cv= CountVectorizer(), CountVectorizer() randomly assigns a number to each word in a process called tokenizing. Then, it counts the number of occurrences of words and saves it to cv. At this point, we've only assigned a method to cv.

- X_train_features = cv.fit_transform(X_train)
- X_test_features = cv.transform(X_test)
- Y_train = Y_train.astype('int')
- Y_test = Y_test.astype('int')

9. **Train the Model & Fit The model**

Logistic regression is a supervised machine learning algorithm mainly used for classification tasks where the goal is to predict the probability that an instance of belonging to a given class or not. It is a kind of statistical algorithm, which analyze the relationship between a set of independent variables and the dependent binary variables. It is a powerful tool for decision-making.

10. **Evaluation and Prediction¶**
    Now, to ensure accuracy, let's test our application.

Our model having

- Accuracy for train_model: 0.997083239847431
- Accuracy for test_model: 0.979372197309417

Conclusion:

In this way we learned how to build and run our model, comparing our predictions against the actual output. Finally, we tested our model using count vectorization.