**SPAM CLASSIFIER**

**Introduction**

The upsurge in the volume of unwanted emails called spam has created an intense need for the development of more dependable and robust antispam filters. Any promotional messages or advertisements that end up in our inbox can be categorised as spam as they don’t provide any value and often irritates us.

Overview of the Dataset used

We will make use of the SMS spam classification data.

The SMS Spam Collection is a set of SMS tagged messages that have been collected for SMS Spam research. It contains one set of SMS messages in English of 5,574 messages, tagged according to being ham (legitimate) or spam.

The data was obtained from UCI’s Machine Learning Repository, alternatively, I have also uploaded the dataset and completed Jupiter notebook onto my GitHub repo.

**In this article, we’ll discuss:**

**Data processing:**

* Import the required packages
* Loading the Dataset
* Remove the unwanted data columns
* Preprocessing and Exploring the Dataset
* Build word cloud to see which message is spam and which is not.
* Remove the stop words and punctuations
* Convert the text data into vectors

**Building a sms spam classification model:**

* Split the data into train and test sets
* Use Sklearn built-in classifiers to build the models
* Train the data on the model
* Make predictions on new data
* **System Architecture: –**

**PROGRAM:**

Import the required packages

%matplotlib inline

Import matplotlib.pyplot as plt

Import csv

Import sklearn

Import pickle

From wordcloud import WordCloud

Import pandas as pd

Import numpy as np

Import nltk

From nltk.corpus import stopwords

From sklearn.feature\_extraction.text import CountVectorizer, TfidfTransformer

From sklearn.tree import DecisionTreeClassifier

From sklearn.model\_selection import GridSearchCV,train\_test\_split,StratifiedKFold,cross\_val\_score,learning\_curve

Data = pd.read\_csv(‘dataset/spam.csv’, encoding=’latin-1’)

Data.head()

Data = data.drop([“Unnamed: 2”, “Unnamed: 3”, “Unnamed: 4”], axis=1)

Data = data.rename(columns={“v2” : “text”, “v1”:”label”})

Data[1990:2000]

now that the data is looking pretty, let’s move on.

Data[‘label’].value\_counts()

# **OUTPUT**

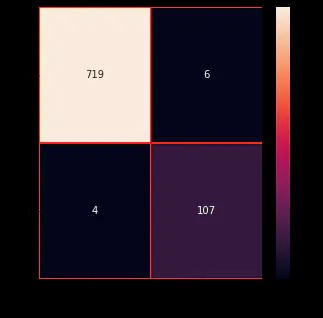
Ham 4825

Spam 747

Name: label, dtype: int64

Preprocessing and Exploring the Dataset

If you are completely new to NLTK and Natural Language Processing(NLP) I would recommend checking out this short article before continuing. Introduction to Word Frequencies in NLP



**Conclusion:**

**In conclusion, a robust and effective spam-based classifier is a crucial tool in today's digital landscape to filter out unwanted and potentially harmful content. By implementing advanced machine learning techniques and continually updating the classifier with new data, we can significantly enhance its accuracy and performance. However, it's essential to balance accuracy with false positives to ensure that legitimate messages aren't mistakenly classified as spam. Regular monitoring and fine-tuning of the classifier are key to maintaining its effectiveness in an ever-evolving online environment.**