BUILDING A SMARTER AI POWERED SPAM CLASSIFIER

TEAM MEMBER

Abish S
963521104004

PHASE 4

DEVELOPMENT PART 2

TOPIC:BUILDING A SMARTER AI POWERED SPAM CLASSIFIER

INTRODUCTION:

Building a smarter AI-powered spam classifier is a crucial step in our ongoing battle against unwanted and potentially harmful communication. To create an effective solution, one of the foundational tasks is loading and preprocessing the dataset. This process lays the groundwork for training a robust and accurate spam filter capable of distinguishing between legitimate and unsolicited content.

The journey to building a smarter AI-powered spam classifier begins with the acquisition of a well-structured and diverse dataset. This dataset should contain a representative sample of both spam and legitimate messages, reflecting the complexities and nuances of real-world communication. Once we have collected this dataset, the subsequent steps of loading and preprocessing it are essential for the following reasons:

Data Understanding: By loading the dataset, we can gain insights into its structure and characteristics. This is vital in understanding the nature of the data, such as the format of messages, any accompanying metadata, and the distribution of spam versus non-spam content.

Data Cleaning: Spam datasets can be noisy, containing a wide range of text types, formats, and possibly irrelevant information. Preprocessing involves cleaning and sanitizing the data by removing duplicates, irrelevant content, and any inconsistencies that may hinder model training.

Text Normalization: Messages within the dataset may exhibit variations in capitalization, punctuation, and special characters. Text normalization techniques, such as lowercasing and removing punctuation, ensure that the data is consistent and easier for the AI model to process.

Tokenization: To make the text data machine-readable, we tokenize the messages into individual words or subword units. This step is critical for converting raw text into numerical representations that AI models can work with effectively.

Feature Engineering: During preprocessing, we can create additional features like word frequencies, n-grams, or embeddings to provide more context to the AI model, improving its ability to identify spam messages accurately.

GIVEN DATA SET

	v1	v2	Unnamed: 2	Unnamed: 3	Unnamed: 4
0	ham	Go until jurong point, crazy.. Available only ...	NaN	NaN	NaN
1	ham	Ok lar... Joking wif u oni...	NaN	NaN	NaN
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...	NaN	NaN	NaN
3	ham	U dun say so early hor... U c already then say...	NaN	NaN	NaN
4	ham	Nah I don't think he goes to usf, he lives aro...	NaN	NaN	NaN

NECESSARY STEPS TO FOLLOW

1. IMPORT LIBRARIES:

Start by improving the necessary libraries

PROGRAM:

```
%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,learn
ing_curve
```

2. Loading the Dataset

```
data = pd.read_csv('dataset/spam.csv', encoding='latin-1')data.head()
```

	v1	v2	Unnamed: 2	Unnamed: 3	Unnamed: 4
0	ham	Go until jurong point, crazy.. Available only ...	NaN	NaN	NaN
1	ham	Ok lar... Joking wif u oni...	NaN	NaN	NaN
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...	NaN	NaN	NaN
3	ham	U dun say so early hor... U c already then say...	NaN	NaN	NaN
4	ham	Nah I don't think he goes to usf, he lives aro...	NaN	NaN	NaN

3.Removing unwanted columns

From the above figure, we can see that there are some unnamed columns and the label and text column name is not intuitive so let's fix those in this step.

data = data.drop(["Unnamed: 2", "Unnamed: 3", "Unnamed: 4"], axis=1)data = data.rename(columns={"v2" : "text", "v1":"label"})data[1990:2000]

	label	text
1990	ham	HI DARLIN IVE JUST GOT BACK AND I HAD A REALLY...
1991	ham	No other Valentines huh? The proof is on your ...
1992	spam	Free tones Hope you enjoyed your new content. ...
1993	ham	Eh den sat u book e kb liao huh...
1994	ham	Have you been practising your curtsey?
1995	ham	Shall i come to get pickle
1996	ham	Lol boo I was hoping for a laugh
1997	ham	\YEH I AM DEF UP4 SOMETHING SAT
1998	ham	Well, I have to leave for my class babe ... Yo...
1999	ham	LMAO where's your fish memory when I need it?

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

data['label'].value_counts()# OUTPUTham 4825spam 747Name: label, dtype: int64

4.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.

import nltk packages and Punkt Tokenizer Modelsimport nltknltk.download("punkt")import warningswarnings.filterwarnings('ignore')

5.Removing punctuation and stopwords from the messages

Punctuation and stop words do not contribute anything to our model, so we have to remove them. Using NLTK library we can easily do it.

```
import nltk
nltk.download('stopwords')
#remove the punctuations and stopwords
import string
def text_process(text):
text = text.translate(str.maketrans('', '', string.punctuation))
text = [word for word in text.split() if word.lower() not in
stopwords.words('english')]
return " ".join(text)
data['text'] = data['text'].apply(text_process)
data.head()
```

	label	text
0	0	Go jurong point crazy Available bugis n great ...
1	0	Ok lar Joking wif u oni
2	1	Free entry 2 wkly comp win FA Cup final tkts 2...
3	0	U dun say early hor U c already say
4	0	Nah dont think goes usf lives around though

Now, create a data frame from the processed data before moving to the next step.

```
text = pd.DataFrame(data['text'])label = pd.DataFrame(data['label'])
```

IMPORTANCE OF LOADING AND DATASET

Loading and preprocessing a dataset is a critical and foundational step in data analysis, machine learning, and artificial intelligence for several important reasons:

Data Understanding: Loading the dataset allows you to familiarize yourself with its structure, contents, and format. This understanding is crucial for making informed decisions throughout the analysis process.

Data Quality Assurance: Preprocessing begins with data quality control. You can identify and rectify missing values, errors, duplicates, and inconsistencies in the data, ensuring that the analysis is based on accurate and reliable information.

Data Cleaning: Datasets are often noisy, and preprocessing helps clean the data by removing irrelevant or redundant information, making it more suitable for analysis.

Data Transformation: Different algorithms require data in specific formats. For example, text data may need to be tokenized, numerical features may require scaling, and categorical variables may need encoding. Preprocessing ensures that the data is in the right format for machine learning algorithms.

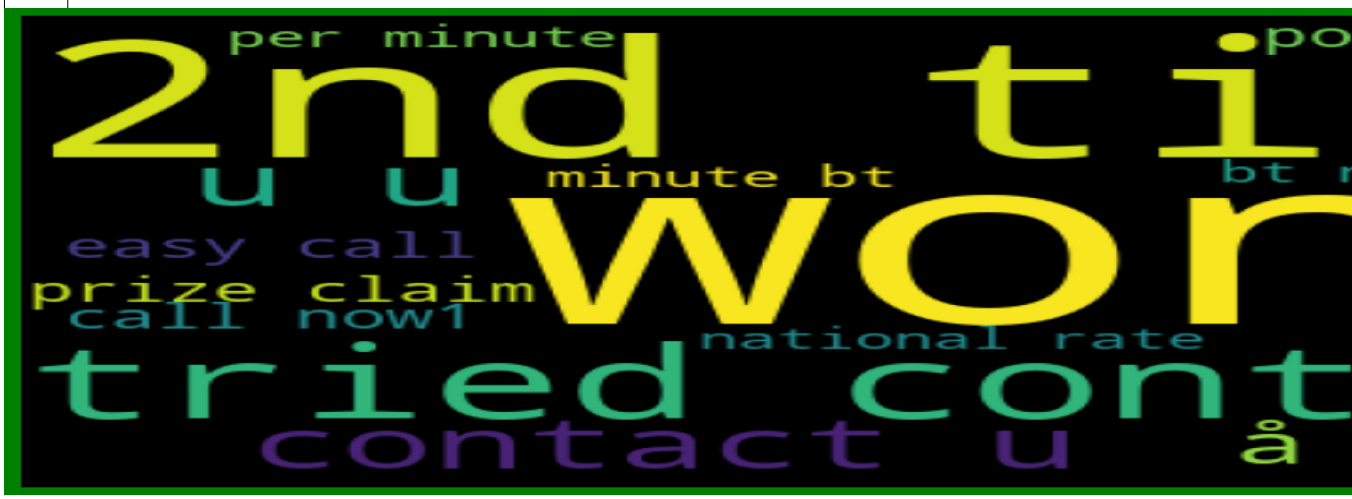
Data Consistency: Maintaining consistent data formatting, such as date formats or categorical values, streamlines the analysis and model training process, reducing errors and improving efficiency.

let's use the above functions to create Spam word cloud and ham word cloud.

```
spam_wordcloud = WordCloud(width=500,
height=300).generate(spam_words)ham_wordcloud =
WordCloud(width=500, height=300).generate(ham_words)

#Spam Word cloudplt.figure( figsize=(10,8),
facecolor='w')plt.imshow(spam_wordcloud)plt.axis("off
")plt.tight_layout(pad=0)plt.show()
```

```
#Creating Ham wordcloudplt.figure( figsize=(10,8), facecolor='g')plt.imshow(ham_wordcloud)plt.axis("off")plt.tight_layout(pad=0)plt.show()
```



from the spam word cloud, we can see that "free" is most often used in spam.

Now, we can convert the **spam** and **ham** into 0 and 1 respectively so that the machine can understand.

```
data = data.replace(['ham','spam'],[0, 1])data.head(10)
```

	label	text
0	0	Go until jurong point, crazy.. Available only ...
1	0	Ok lar... Joking wif u oni...
2	1	Free entry in 2 a wkly comp to win FA Cup fina...
3	0	U dun say so early hor... U c already then say...
4	0	Nah I don't think he goes to usf, he lives aro...
5	1	FreeMsg Hey there darling it's been 3 week's n...
6	0	Even my brother is not like to speak with me. ...
7	0	As per your request 'Melle Melle (Oru Minnamin...
8	1	WINNER!! As a valued network customer you have...
9	1	Had your mobile 11 months or more? U R entitle...

PROCEDURE

Building a smarter AI-powered spam classifier involves a multi-step process. Here's a high-level procedure:

Data Collection:
Gather a diverse and extensive dataset of emails or messages, including both spam and non-spam (ham) examples. This data will be used to train and evaluate your AI model.

Data Preprocessing:
Clean and preprocess the data. This includes removing irrelevant information, standardizing text (e.g., lowercasing), and tokenizing the text into words or phrases.

Feature Engineering:
Extract relevant features from the text data. Common features include word frequency, character n-grams, and metadata such as sender information and timestamps.

Data Splitting:
Divide your dataset into training, validation, and testing sets. Typically, this might be an 80-10-10 or 70-15-15 split.

Model Selection:
Choose the type of AI model you want to use. Common choices include Naive Bayes, Support Vector Machines, or more advanced models like deep learning-based neural networks.

Model Training:
Train your chosen model on the training dataset. The model should learn to distinguish between spam and non-spam messages based on the features you've extracted.

Hyperparameter Tuning:
Optimize the hyperparameters of your model to achieve better performance. This might involve adjusting learning rates, batch sizes, and architectural parameters.

Validation:
Evaluate your model's performance on the validation set to ensure it's not overfitting and is generalizing well to unseen data.

Testing:
Assess your model's performance on the test dataset. This provides a final estimate of how well your AI-powered spam classifier will perform in the real world.

Model Evaluation:
Use appropriate metrics such as accuracy, precision, recall, and F1-score to measure the model's performance. You may need to iterate on the previous steps to improve the model's accuracy

FEATURE ENGINEERING

Feature engineering is a crucial step in building a smarter AI-powered spam classifier. Here are some feature engineering techniques you can consider:

Text-Based Features:

Word Frequency: Calculate the frequency of each word in the text, as spam messages often contain specific keywords.
Character Count: Longer messages may be indicative of spam.
Use of Symbols: Count special characters and symbols that are common in spam.
Capitalization: Count the number of uppercase letters, as spam messages may use excessive capitalization.
NLP-Based Features:

N-grams: Analyze word combinations (bigrams, trigrams) to capture context.
Sentiment Analysis: Determine the sentiment of the message. Spam messages might use overly positive or negative language.
Part-of-Speech Tagging: Analyze the distribution of parts of speech in the text.
Structural Features:

Message Length: Determine the length of the message, subject line, or URL.
Number of Links: Count the number of hyperlinks in the message.
HTML Tags: Check for the presence of HTML tags, which are common in email spam.
Header Information:

Sender Information: Analyze sender details, including the email address and domain.
Received Headers: Examine the path the email took, as spammers often use multiple servers to hide their identity.
Time-Based Features:

Timestamps: Consider the time and date the message was sent. Spam patterns may vary at different times.
User Behavior:

User Interaction: If you have user feedback (e.g., users marking emails as spam), incorporate this data into your features.
External Data:

Blacklists and Whitelists: Utilize external databases or services that maintain lists of known spammers or trusted senders.
Machine Learning Features:

TF-IDF: Use TF-IDF (Term Frequency-Inverse Document Frequency) to weigh the importance of words in the message.
Word Embeddings: Represent words as dense vectors using techniques like Word2Vec or FastText.
Domain Analysis:

Domain Reputation: Check the reputation of the sending domain.
Image Analysis:

If applicable, analyze the content of images within emails.
Remember to perform feature selection to remove irrelevant or highly correlated features. Experiment with different combinations of features and machine learning algorithms to find the most effective model for spam classification.

Regularly update and retrain your model to adapt to evolving spam patterns.

Conclusion

In summary, loading and preprocessing a dataset are indispensable steps in the data analysis and modeling workflow. Properly prepared data is the foundation for building accurate and reliable machine learning and AI models. It enhances the efficiency and effectiveness of the modeling process and plays a pivotal role in various domains, including data science, natural language processing, image recognition, and many others.