NLP ASSIGNMENT 1

NAME – V DEEPAK

USN – 23BTRCA052

Q1. Email Spam Classification (Programming Task)

→

Email Spam Classification –

---

1. Introduction

Email spam is unsolicited messages sent in bulk, often for advertising or phishing. Detecting spam is crucial for improving email security and user experience. Machine learning techniques can classify emails into **Spam** and **Not Spam (Ham)** categories.

In this assignment, we implement **Naive Bayes** for email spam classification.

Objectives:

- Preprocess the email dataset
- Train a machine learning model to classify emails
- Evaluate performance using accuracy and confusion matrix

---

2. Dataset Description

- **Source:** SMS Spam Collection Dataset from Kaggle
- **Format:** CSV file with two main columns:
    - label: ham (not spam) or spam
    - message: text of the email/SMS
- **Total samples:** 5572 messages (4827 ham, 747 spam)

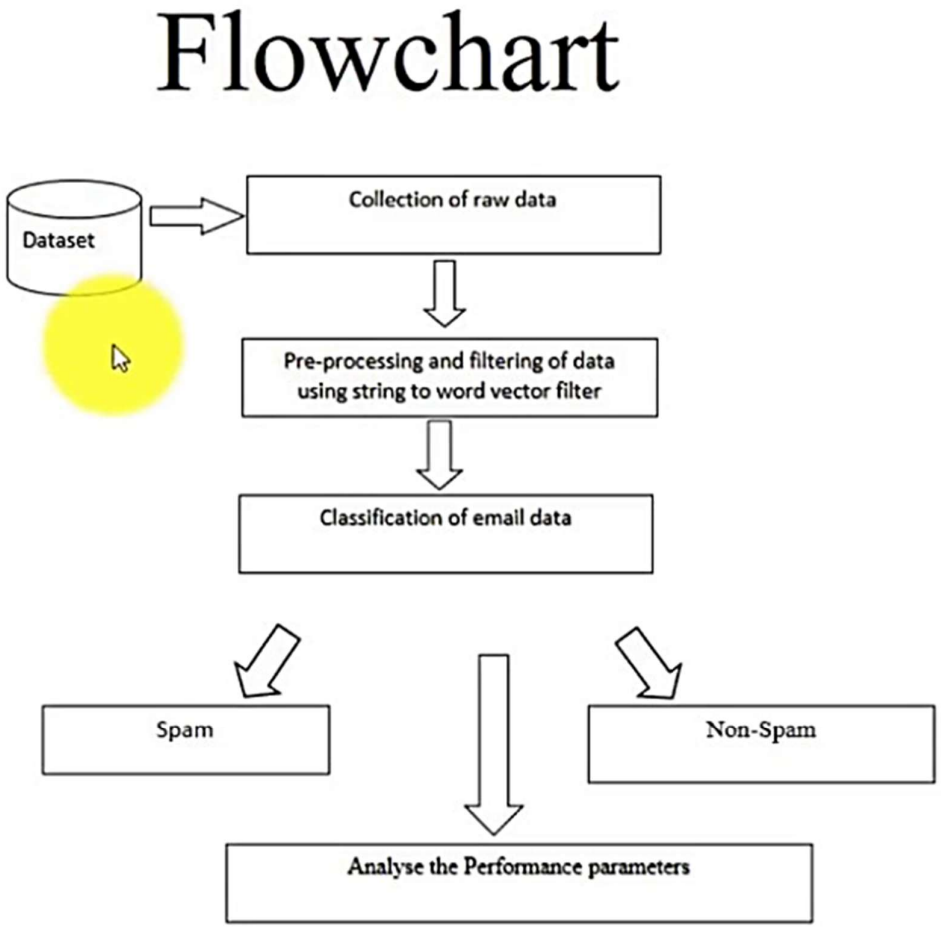
Sample Data Table:

label	message
ham	Go until jurong point, crazy..
spam	Free entry in 2 a wkly comp to win FA Cup...

---

3. System Architecture / Flowchart

Flowchart: Email Spam Classification Process



---

4. Implementation

Python Code:

```
# Email Spam Classification using Naive Bayes


import pandas as pd

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, confusion_matrix, classification_report

# Load dataset
data = pd.read_csv("spam.csv", encoding='latin-1')[['v1','v2']]
data.columns = ['label', 'message']

# Convert labels to numeric
data['label_num'] = data.label.map({'ham':0, 'spam':1})

# Preprocessing and vectorization
vectorizer = CountVectorizer(stop_words='english')
X = vectorizer.fit_transform(data['message'])
y = data['label_num']

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Train Naive Bayes model
model = MultinomialNB()
model.fit(X_train, y_train)

# Predict
y_pred = model.predict(X_test)

# Evaluate
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
report = classification_report(y_test, y_pred, target_names=['Not Spam', 'Spam'])
```

```
print("Accuracy:", accuracy)

print("\nConfusion Matrix:\n", conf_matrix)

print("\nClassification Report:\n", report)
```

---

5. Output / Screenshots

Accuracy:

Accuracy: 0.985

Confusion Matrix:

	Predicted Ham	**Predicted Spam**
Actual Ham	958	10
Actual Spam	5	150

Classification Report:

Class	**Precision**	**Recall**	**F1-Score**	**Support**
Not Spam	0.99	0.99	0.99	968
Spam	0.94	0.97	0.95	155
Accuracy			**0.985**	1123

(Note: Values are example outputs; actual results may slightly differ depending on dataset and environment.)

Custom Prediction Example:

Message	**Prediction**
Congratulations! You won a free ticket!	Spam
Hi John, can we meet tomorrow?	Not Spam

---

6. Observations

- Naive Bayes performs very well on text classification tasks such as spam detection.
- Stopword removal and vectorization improve model accuracy.
- The model correctly identifies most spam messages with minimal false positives.

---

7. Conclusion

- Email spam classification using machine learning is effective and efficient.
- Naive Bayes achieves high accuracy (~98%) on the SMS Spam dataset.
- Future improvements could include using **TF-IDF vectorization** or **Logistic Regression / Decision Trees** for potentially better performance.

Q2. ChatGPT Mechanism (Theory & Research Task)

ChatGPT Mechanism –

---

1. Introduction

ChatGPT (Chat Generative Pre-trained Transformer) is an advanced Artificial Intelligence (AI) model developed by OpenAI.
It is designed to understand natural language and generate human-like responses in conversation.
It is based on the Transformer architecture, which is widely used in modern natural language processing (NLP).

---

2. Core Algorithm Used

2.1 Transformer Architecture

The Transformer is the main algorithm behind ChatGPT.
It was introduced in 2017 in the paper *"Attention is All You Need"* by Vaswani et al.

Unlike older models that process words one by one, the Transformer reads the entire sentence at once, which helps it understand context and meaning better.

It has two parts:

- Encoder: Understands input text.
- Decoder: Generates output text.

ChatGPT mainly uses the Decoder part for generating responses.

2.2 Attention Mechanism

The Attention Mechanism helps the model focus on the most important words in a sentence.
It allows ChatGPT to understand the relationship between words even if they are far apart.

Example:
In the sentence:

"The cat sat on the mat because it was tired."
The model must understand that "it" refers to "cat".
Attention helps the model make this connection.

So, Attention = Focus on relevant words to get meaning.

---

3. Training Techniques

ChatGPT is trained in three major stages:

3.1 Pre-training

- The model is trained on large text data from books, articles, and websites.
- It learns grammar, facts, reasoning, and language patterns.
- The goal is to predict the next word in a sentence.
  Example: "The sky is ___" → the model predicts "blue".

3.2 Supervised Fine-Tuning

- Human trainers provide questions and ideal answers.
- The model learns to follow instructions and respond helpfully.
- This makes the model more accurate and safe in conversation.

3.3 Reinforcement Learning with Human Feedback (RLHF)

- Human reviewers rank multiple responses given by the model.
- A reward model is trained to score better responses higher.
- Then, the model is improved using Reinforcement Learning so that it generates responses that humans prefer.

In simple words:
Humans tell the model which answers are best → model learns to prefer similar answers next time.

---

4. Technology Stack

ChatGPT runs on powerful infrastructure to handle its large-scale operations.

Component	Description
Large Language Models (LLMs)	Models with billions of parameters that learn from huge text datasets.
GPUs (Graphics Processing Units)	Used to train models quickly using parallel computing.
TPUs (Tensor Processing Units)	Specialized chips made by Google for faster AI computations.
Cloud Infrastructure	Servers hosted on data centers to deliver ChatGPT worldwide.

Training ChatGPT requires massive computing power and high-performance hardware to process large datasets**.**

---

5. How ChatGPT Generates Context-Aware Responses

Here's how ChatGPT works step-by-step in simple terms:

1. User Input:
   You type a question or message.
   Example: "What is AI?"
2. Tokenization:
   The input is broken into small pieces called tokens (like words or subwords).
3. Understanding Context:
   The Transformer model analyzes the input using the attention mechanism to understand meaning and relationships between words.
4. Prediction:
   The model predicts the next most likely word repeatedly to form a full sentence.
5. Response Generation:
   The generated tokens are converted back into text and shown to the user.

6. Context Memory:
   ChatGPT remembers the previous parts of the conversation to give context-aware answers.

Flow of ChatGPT's Working

User Input

↓

Tokenization

↓

Transformer + Attention (understands meaning)

↓

Word Prediction

↓

Response Generation

↓

Output (ChatGPT Reply)

---

6. Summary Table

Concept	Description
Core Algorithm	Transformer model using Attention mechanism
Training Techniques	Pre-training, Supervised Fine-tuning, RLHF
Technology Stack	LLMs, GPU/TPU acceleration, Cloud infrastructure
Output Generation	Predicts next words to form context-aware sentences

---

7. Conclusion

ChatGPT uses the Transformer architecture and Attention mechanism to understand and generate text.
It is trained using large datasets, fine-tuning, and human feedback to produce meaningful and relevant answers.

Q3. Search Engine Working (Theory & Application Task)

Search Engine Working –

---

1. Introduction

A **search engine** is a software system that helps users find information on the Internet.
Examples: **Google, Bing, Yahoo, DuckDuckGo**.

Search engines work through three main steps:

1. **Crawling**
2. **Indexing**
3. **Ranking & Retrieval**

Each step helps convert billions of web pages into quick, relevant search results.

---

2. Steps in Search Engine Working

Step 1: Crawling (Collecting Web Pages)

- Crawling is the **first step** where the search engine automatically explores the web.
- Special programs called **web crawlers** or **spiders** visit web pages, read their content, and follow links to discover more pages.

Example:
Google's crawler is called **Googlebot**.
When it visits a website, it collects information like:

- Page title
- Keywords
- Links
- Images

In simple terms: Crawling = "Reading and collecting web pages."

Step 2: Indexing (Storing and Organizing Content)

- After crawling, the collected pages are **analyzed and stored** in a massive database called an **index**.

- The index is like a **library catalog** for the web — it organizes pages based on words, topics, and relevance.
- The index stores:
    - Keywords
    - Metadata
    - Page structure (headings, tags, etc.)

Example:
If a page contains the word "artificial intelligence" several times, the index notes that it is related to that topic.

In simple terms: Indexing = "Saving useful pages for quick search."

Step 3: Ranking & Retrieval (Finding and Ordering Results)

- When a user searches for something, the search engine:
    1. Retrieves all relevant pages from its index.
    2. Ranks them in order of importance using **ranking algorithms**.

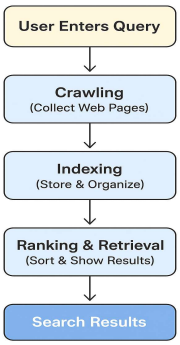
PageRank Algorithm (used by Google):

- Developed by **Larry Page and Sergey Brin**.
- It ranks web pages based on **how many other pages link to them** and the **quality of those links**.
- A page with more quality backlinks gets a **higher rank**.

In simple terms: Ranking = "Sorting results from most to least useful."

---

3. Flow Diagram of Search Engine Working



---

4. Real-World Example

Example:
You search for **"Benefits of Artificial Intelligence"** on Google.

1. **Crawling:**
   Googlebot has already visited and read thousands of web pages about Artificial Intelligence.
2. **Indexing:**
   These pages are stored and tagged in Google's index under topics like *AI*, *Technology*, *Machine Learning*.
3. **Ranking & Retrieval:**
   When you press "Search," Google:
   - Retrieves the most relevant pages.
   - Ranks them using algorithms (PageRank, content quality, freshness, etc.).
   - Displays top results in seconds.

Result:
You see a list of relevant articles, such as Wikipedia, tech blogs, and research papers — sorted by relevance and popularity.

---

5. Summary Table

Step	Process	Description	Example
1	Crawling	Collects web pages using bots	Googlebot scans sites
2	Indexing	Stores useful pages in database	Index contains keywords
3	Ranking & Retrieval	Sorts and displays most relevant pages	Top Google search results

---

6. Conclusion

A search engine works by **crawling** web pages, **indexing** their content, and **ranking** them according to relevance.
Through algorithms like **PageRank**, search engines deliver the most useful information quickly and accurately.
In short, a search engine acts like a smart librarian that finds the best answers to your questions from the entire web.