

Department of Computer Engineering
BE Computer-B (2025-26 Sem I)
LP-III Machine Learning
Practical Assignment 2: Binary Classification (KNN and SVM)

[CO1-CO3, BT: L3 (Apply)] [Max Marks: 10]

Problem Definition: Classify the email using the binary classification method. Email Spam detection has two states: a) Normal State – Not Spam, b) Abnormal State – Spam. Use K-Nearest Neighbors and Support Vector Machine for classification. Analyze their performance.

Dataset link: The emails.csv dataset on the Kaggle

<https://www.kaggle.com/datasets/balaka18/email-spam-classification-dataset-csv>

Learning Outcomes:

By completing this practical, students will be able to:

Learning Outcome	BT Level
Understand the concept of email spam filtering and binary classification	Understand (L2)
Preprocess and vectorize textual email data	Apply (L3)
Implement KNN and SVM algorithms using Python/Scikit-learn	Apply (L3)
Compare classification results using metrics like accuracy, precision, recall	Analyze (L4)
Evaluate model performance to choose the best classifier	Evaluate (L5)

Software / Hardware Requirements

Software:

- Python 3.x
- Jupyter Notebook / Google Colab / VS Code
- Scikit-learn
- Pandas
- Numpy
- Matplotlib / Seaborn
- NLTK / SpaCy (for NLP preprocessing)

Hardware:

- A system with minimum 4 GB RAM
- Intel i3 or above processor
- Internet connection (for package installation and datasets)

Theory:

1. Dataset Description:

The csv file contains 5172 rows, each row for each email. There are 3002 columns. The first column indicates Email name. The name has been set with numbers and not recipients' name to protect privacy. The last column has the labels for prediction : 1 for spam, 0 for not spam. The remaining 3000 columns are the 3000 most common words in all the emails, *after excluding the non-alphabetical characters/words*. **For each row, the count of each word(column) in that email(row) is stored in the respective cells. Thus, information regarding all 5172 emails are stored in a compact dataframe rather than as separate text**

Acknowledgement:

The dataset is referred from Kaggle: <https://www.kaggle.com/balakal8/email-spam-classification-dataset-csv>

2. Email Spam Detection

Spam emails are unsolicited messages, often sent in bulk. Detecting spam is a binary classification task where each email is classified as:

- **Spam (1)** — Abnormal
- **Not Spam (0)** — Normal

3. Data Preprocessing:

Data preprocessing is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model.

A real-world data generally contains noises, missing values, and maybe in an unusable format which cannot be directly used for machine learning models. Data preprocessing is required tasks for cleaning the data and making it suitable for a machine learning model which also increases the accuracy and efficiency of a machine learning model.

4. Binary Classification:

- A supervised learning task where the output is restricted to two classes. In this case, "Spam" and "Not Spam".
- Popular algorithms that can be used for binary classification include:
 - k-Nearest Neighbors
 - Decision Trees
 - Support Vector Machine
 - Naive Bayes

4. K-Nearest Neighbors (KNN)

K-Nearest Neighbors (KNN) is a **supervised machine learning algorithm** used for **classification** and **regression** problems. In the context of **classification**, it assigns a class to a data point based on the **majority class among its k nearest neighbors**.

4.1 Steps in KNN Classification

1. **Choose** the number of neighbors k
2. **Calculate** the distance between the test point and all training points
3. **Sort** the distances and determine the k nearest neighbors
4. **Count** the labels among the k neighbors
5. **Assign** the most frequent label to the test point

4.2 Advantages of KNN

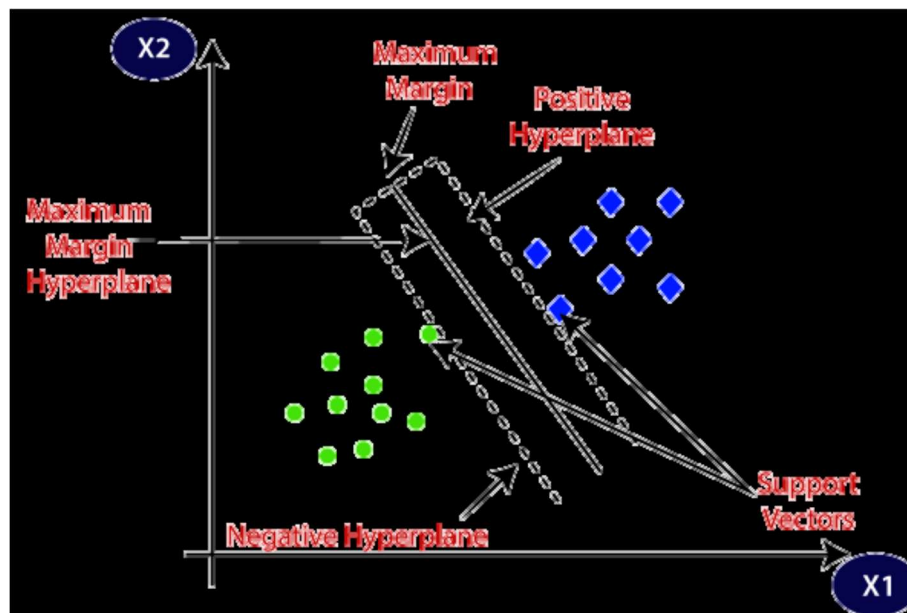
- Simple and easy to implement
- No assumptions about data distribution
- Works well with small, well-labeled datasets

4.3 Disadvantages

- **Slow at prediction** time with large data
- **Sensitive to irrelevant features** or unscaled data
- Struggles in **high-dimensional** spaces (curse of dimensionality)
- Requires good choice of k and distance metric

5. Support Vector Machine (SVM): Support Vector Machine is a **supervised learning algorithm** used for both **classification** and **regression**, but it is mostly used for **classification tasks**. SVM works by finding the **best decision boundary (hyperplane)** that separates data points of different classes with the **maximum margin**.

In email classification, SVM separates **spam** and **non-spam** emails based on patterns in the text. Imagine drawing a line between spam and non-spam emails on a 2D graph (based on features like word frequency). SVM finds the line (or hyperplane) that best separates the two classes and is farthest from both classes (maximum margin).



SVM can be of two types:

- **Linear SVM:** Linear SVM is used for linearly separable data, which means if a dataset can be classified into two classes by using a single straight line, then such data is termed as linearly separable data, and classifier is used called as Linear SVM classifier.
- **Non-linear SVM:** Non-Linear SVM is used for non-linearly separated data, which means if a dataset cannot be classified by using a straight line, then such data is termed as non-linear data and classifier used is called as Non-linear SVM classifier.

5.2 Advantages

- High accuracy for text classification tasks
- Works well in high-dimensional space
- Effective in cases with clear margin of separation

5.3 Disadvantages

- Slow training on large datasets
- Requires careful parameter tuning (e.g., kernel, C, gamma)
- Not very interpretable
- Performance drops with noisy datasets

6. Model Evaluation Metrics:

After building models, we evaluate how well they predict fares using:

Metric	Formula	Description
Accuracy	$(TP + TN) / (TP + TN + FP + FN)$	Proportion of correctly classified emails (both spam and not spam)
Precision	$TP / (TP + FP)$	Of all emails predicted as spam, how many are actually spam
Recall	$TP / (TP + FN)$	Of all actual spam emails, how many were correctly identified
F1 Score	$(2 \cdot \text{Precision} \cdot \text{Recall}) / (\text{Precision} + \text{Recall})$	Harmonic mean of Precision and Recall; balances false positives & false negatives
Specificity	$TN / (TN + FP)$	Proportion of actual non-spam emails correctly classified
Error Rate	$(FP + FN) / (TP + TN + FP + FN)$	Proportion of incorrectly classified emails

Where

- TP True Positive (Spam correctly classified as spam)
TN True Negative (Not spam correctly classified as not spam)
FP False Positive (Not spam incorrectly classified as spam)
FN False Negative (Spam incorrectly classified as not spam)

7. Algorithm

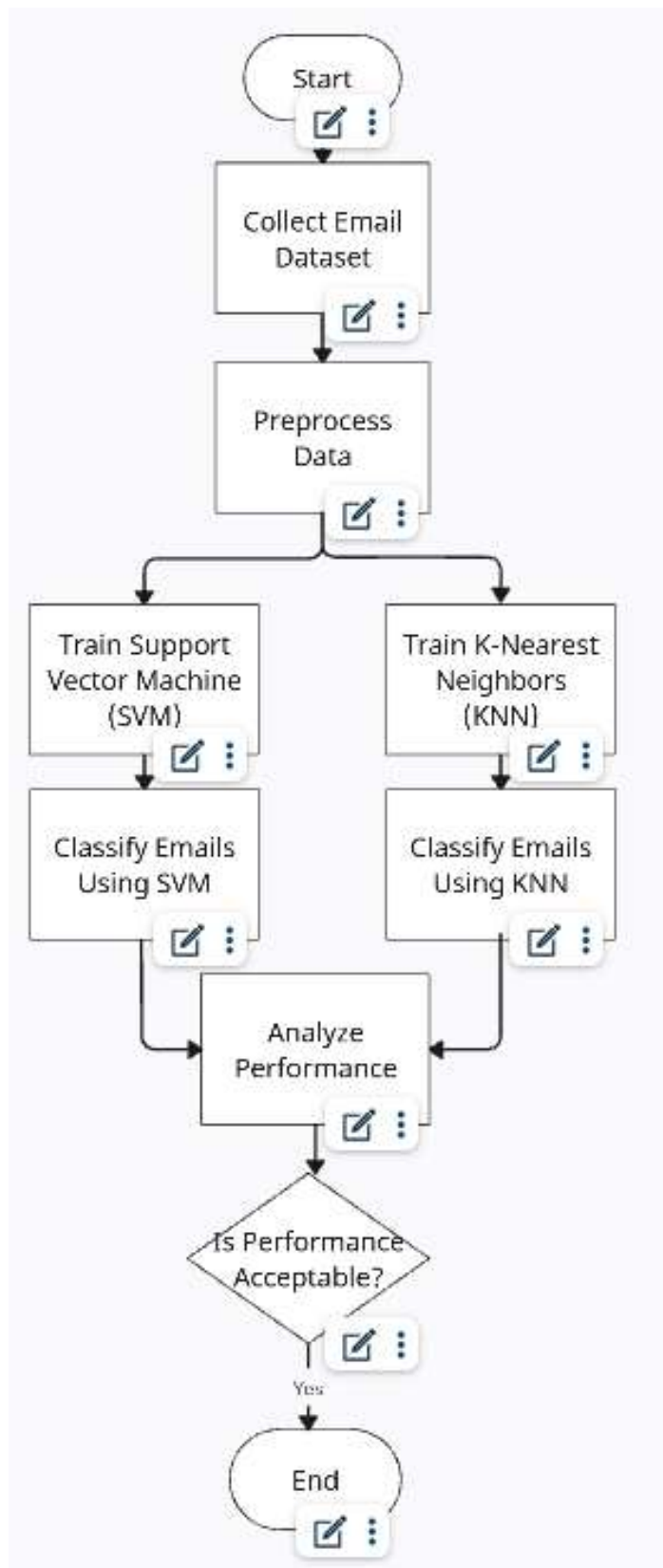
7.1 KNN Algorithm for Email Spam Detection

1. Input: Dataset D, value of k, email to classify x
2. Preprocess email text into numeric features using TF-IDF.
3. Compute distance between x and all training samples.
4. Identify the k-nearest neighbors.
5. Count the class labels of the neighbors.
6. Return the majority class as prediction.

7.2 SVM Algorithm for Email Spam Detection

1. Input: Dataset D
2. Preprocess emails using TF-IDF vectorization.
3. Choose a kernel (e.g., linear).
4. Train SVM to find optimal hyperplane.
5. For new email x, compute:
 - $f(x) = w^T x + b$
6. Classify as:
 - Spam if $f(x) > 0$
 - Not spam if $f(x) < 0$

8. Flow Chart



Result:

(Give Comparative Performance analysis of KNN and SVM using various evaluation metrics in Tabular Form)

Conclusion:

In this experiment, students successfully:

- Preprocessed textual email data
- Applied **KNN** and **SVM** classifiers
- Compared their performance using evaluation metrics

Viva Questions

Question	BT Level
What is binary classification?	Remember (L1)
Explain the difference between KNN and SVM	Understand (L2)
How is Euclidean distance calculated in KNN?	Apply (L3)
Why is TF-IDF better than CountVectorizer in spam detection?	Analyze (L4)
How would you decide which classifier performs better?	Evaluate (L5)
Can you suggest a way to handle class imbalance in spam detection?	Create (L6)

Rubrics for Evaluation (Example)

Question No.	Criteria	Best	Good	Average	Poor
Q1	Correctness of Program / Demonstration of Program [4]	Perfect [4]	Can be better [3]	Satisfied [2]	Poor [0]
	File Submission/Documentation [2]	Perfect [2]	Can be better [1.5]	Satisfied [1]	Poor [0]
	Timely Submission [2]	On Time [2]	After 1 week [1.5]	After 1.5 week [1]	At the end of semester [0]
	Viva [2]	Perfect [2]	Can be better [1.5]	Satisfied [1]	Poor [0]

Dr. Nikita Singhal
(Subject Incharge)

DR SR Dhore
(HoD)