Text Classification with Python: Comparing Three Classifiers

Spam Detection using
Machine Learning

Presented by: Sultan &
Ch Mubashir

Date: 04-12-2024

Text Classification with Python: Comparing Three Classifiers

In this project, we will compare the performance of three popular classifiers for text classification: Naive Bayes, Support Vector Machines (SVM), and Random Forests. We will train each classifier using a labeled dataset, and then evaluate their accuracy and efficiency on a holdout test set. By comparing the results, we aim to identify which classifier is best suited for our specific text classification task.

Introduction

Objective

Build and compare three text classifiers for spam detection.

Dataset

Sourced from Kaggle, containing labeled spam and non-spam messages.

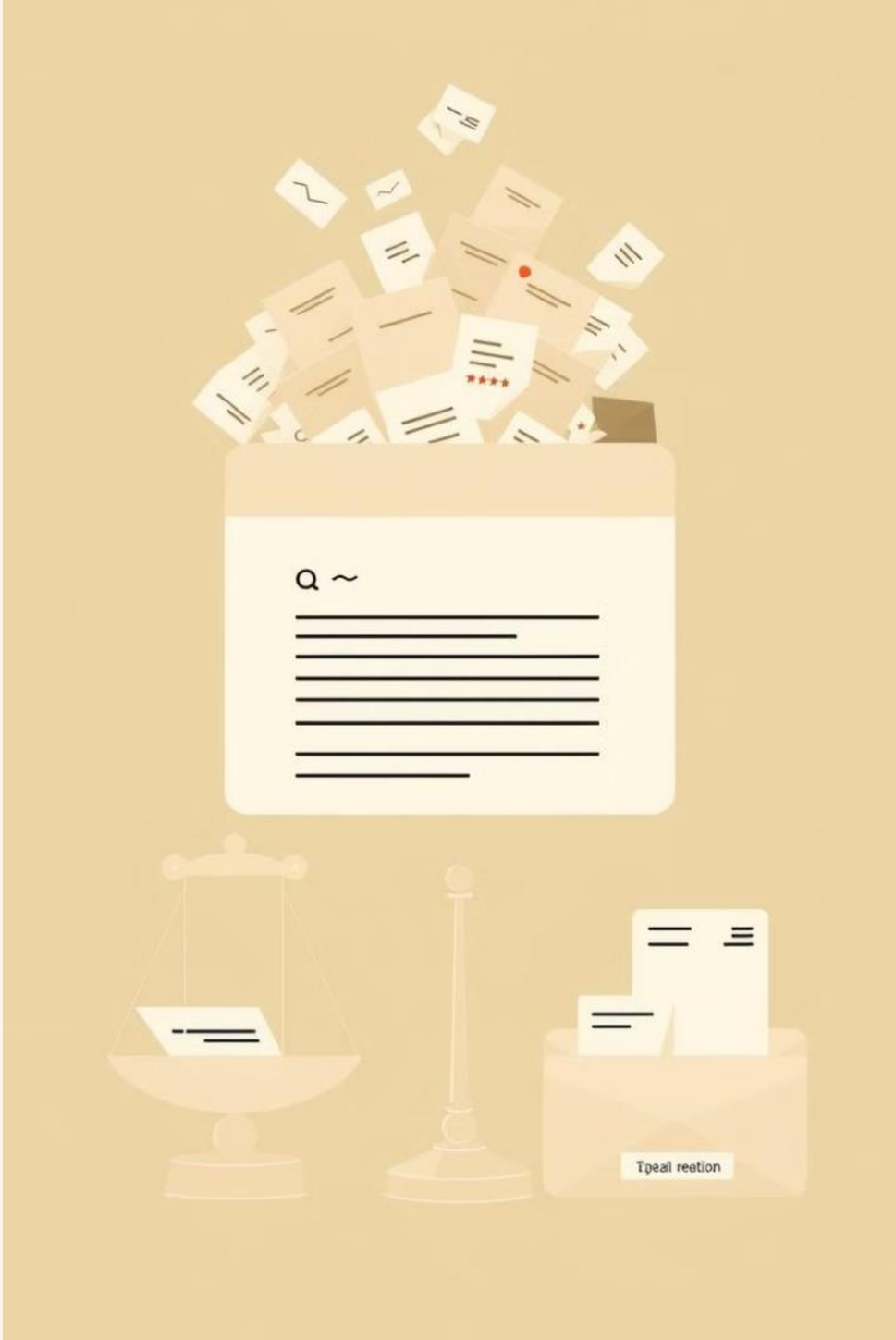
What is Text Classification?

Definition

Assigning labels to text based on its content or context.

Common Applications

- Spam Detection
- Sentiment Analysis
- Document Categorization



Dataset Preparation

Dataset Details

- Columns: Message (text) and Category (spam/ham).
- Distribution: Spam vs. Ham counts (use a pie chart or bar graph).

Preprocessing Steps

- Removing stop words.
- Transforming text into numerical format using TF-IDF.

Classifiers Used

1 **Multinomial Naive Bayes**

A probabilistic classifier that assumes independence between features.

2 **Complement Naive Bayes**

An alternative to Multinomial Naive Bayes, often better for imbalanced datasets.

3 **Support Vector Classifier**

A powerful classifier that seeks to find the optimal hyperplane to separate classes.

Training and Testing

Dataset Split

80% for training, 20% for testing.

Implementation

Used sklearn for pipelines and model training.

Evaluation Metrics

Accuracy

Overall proportion of correctly classified instances.

Precision

Proportion of correctly classified spam messages among all predicted spam messages.

Recall

Proportion of correctly classified spam messages among all actual spam messages.

F1-Score

Harmonic mean of precision and recall, providing a balanced measure of performance.

Results

85%

Support Vector Classifier

Highest accuracy.

70%

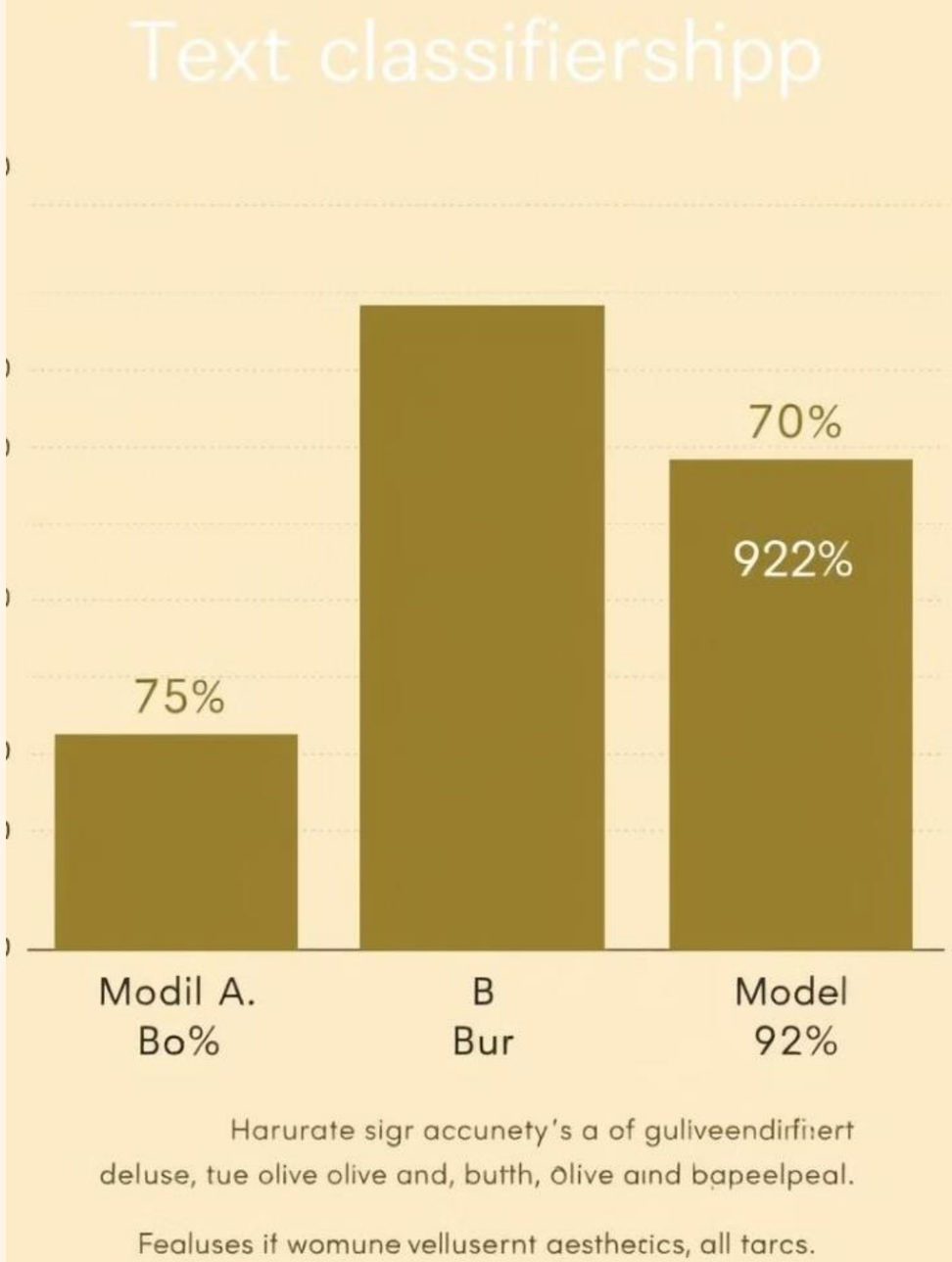
Complement Naive Bayes

Moderate performance.

60%

Multinomial Naive Bayes

Least accurate.



Questions for the Audience

- What are your top concerns about implementing text classification in your business?
- Have you encountered any challenges with **data quality** or **feature engineering** for text classification models?
- How important is model interpretability and explainability for your text classification use cases?
- Are you interested in exploring **advanced techniques** like **deep learning** for text classification?
- What are your thoughts on **deploying text classification models in production** and **monitoring their performance**?

Thank You

I appreciate your time and attention throughout this presentation. Your valuable feedback will help us refine our text classification models to better serve your needs.