**SMS Spam Detection — Project Template**

# Title Page

* **Project Title**: SMS Spam Detection Using Machine Learning
* **Your Name:** (your name)
* **Institution/ Department :** University college of engineering ,Tindivanam/ B.E CSE
* **Date :** (submission date)
* **Github :**(your github link)

# Table of Contents

1. Title Page
2. Table of Contents
3. Project Overview
4. Objectives & Problem Statement  
    4.1 Problem Statement  
    4.2 Objectives
5. Proposed Solution
6. Features  
    6.1 Functional Features  
    6.2 Non-Functional Features
7. Technologies & Tools
8. System Architecture
9. Implementation Steps
10. Output / Screenshots
11. Advantages
12. Future Enhancements
13. Conclusion

# Project Overview

* **Background / Motivation**:

Spam messages are a major concern for users and mobile networks, as they often contain fraudulent, phishing, or promotional content. Detecting spam SMS automatically improves user experience, prevents scams, and strengthens communication security.

* **Scope**:

This project focuses only on SMS text messages, classifying them into two categories: Spam or Ham (Not Spam) using machine learning techniques. The project does not cover emails or multimedia spam.

* **High-level Description**:

The system reads raw SMS messages, preprocesses them, converts text into numerical features, trains multiple ML models, and predicts whether a new message is spam or not.

* **Dataset / Data Source Summary**

**Data set llink:** [**https://www.kaggle.com/datasets/uciml/sms-spam-collection-dataset**](https://www.kaggle.com/datasets/uciml/sms-spam-collection-dataset)

# Objectives & Problem Statement

## Problem Statement

Given a short SMS text message, classify it as **Spam** or **Ham** using machine learning.  
Challenges include:

* Short informal text
* Misspellings, abbreviations, and slang
* Class imbalance (fewer spam than ham)

**4.2 Objectives**

List specific goals your project will achieve.

Examples:

* Build a predictive model with high accuracy and recall for spam detection.
* Preprocess and clean text effectively.
* Compare multiple ML algorithms.
* Visualize model performance.
* Develop a reusable prediction pipeline

# Proposed Solution

* **Approach / Methodology**:

A supervised machine learning approach was used. The main pipeline:

Raw SMS → Preprocessing → Feature Extraction → Model Training → Prediction.

* **Pipeline Overview**: raw SMS → preprocessing → feature extraction → model → prediction
* **Justification of Choices**:
* TF-IDF Vectorizer to represent text efficiently.
* Multinomial Naive Bayes model for fast and accurate text classification.
* Alternative models (Logistic Regression, SVM) tested for comparison.

# Features

## Functional Features

* Accepts raw SMS text and outputs “Spam” or “Ham”.
* Provides classification probability.
* Can process multiple messages at once.
* Easy to extend into a GUI or API interface.

## Non-Functional Features

* High accuracy (above 95%).
* Fast and lightweight.
* Reliable and easy to maintain.
* User-friendly for non-technical users.

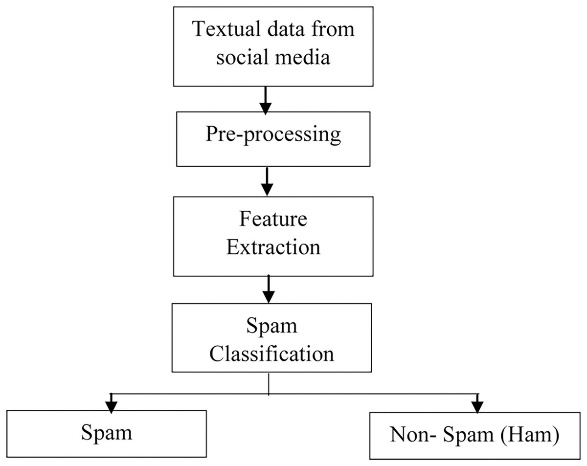
# Technologies & Tools

List all software, libraries, frameworks used. Examples:

* **Language**: Python
* **Data Handling**: pandas, numpy
* **Visualization**: matplotlib, seaborn
* **Text / NLP**: nltk, re (regex), spaCy (if used)
* **Feature extraction**: scikit-learn’sCountVectorizer, TfidfVectorizer
* **Machine Learning**: scikit-learn (Naive Bayes, Logistic Regression, Random Forest, SVM etc.)
* **Model persistence**: pickle, joblib
* **Environment / Tools**: Google colab / GitHub

# System Architecture

* **Architecture Diagram** (block diagram)



* **Component Descriptions**:
* **Preprocessing Module:** Cleans and tokenizes SMS text.
* **Feature Extraction:** Converts cleaned text into TF-IDF vectors.
* **Model Training:** Learns spam/ham patterns using ML.
* **Prediction Layer:** Outputs spam or ham for new messages.
* **Data Flow**: How data moves through the system (during training, and during inference)

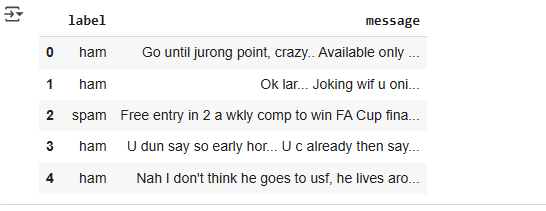
# Implementation Steps

Step-by-step how you built the system. Example:

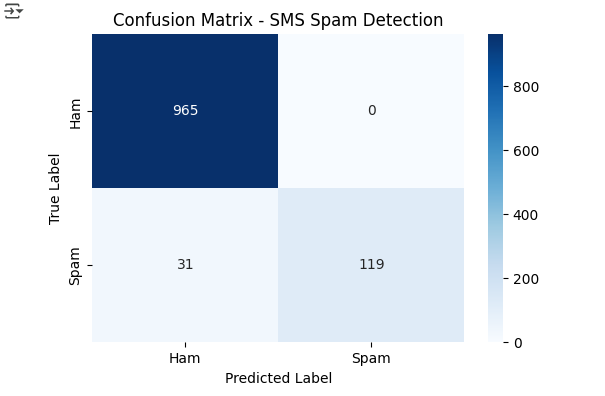
1. Load dataset (spam.csv).
2. Perform EDA — check spam vs ham count, message lengths, word clouds.
3. Preprocess text:
   * Lowercasing
   * Removing punctuation and stopwords
   * Tokenization and stemming
4. Convert to numeric features using TF-IDF.
5. Split data into train (80%) and test (20%).
6. Train models:
   * Multinomial Naive Bayes
   * Logistic Regression
   * SVM
7. Evaluate using Accuracy, Precision, Recall, F1-score.
8. Visualize confusion matrix and word frequencies.
9. Save final model and vectorizer using pickle.
10. Build prediction function / UI for new messages.

# Output / Screenshots

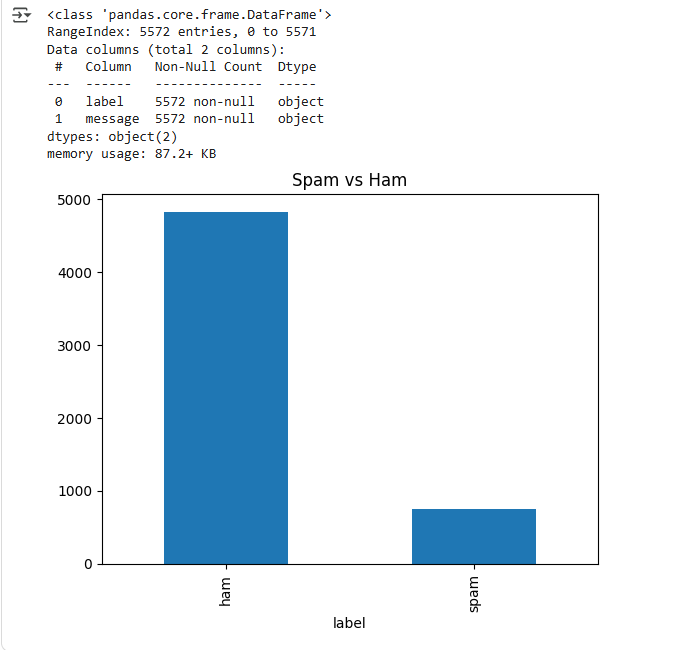
* Sample inputs & outputs (SMS text → prediction)



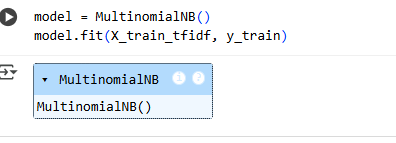
* Confusion matrix, classification report



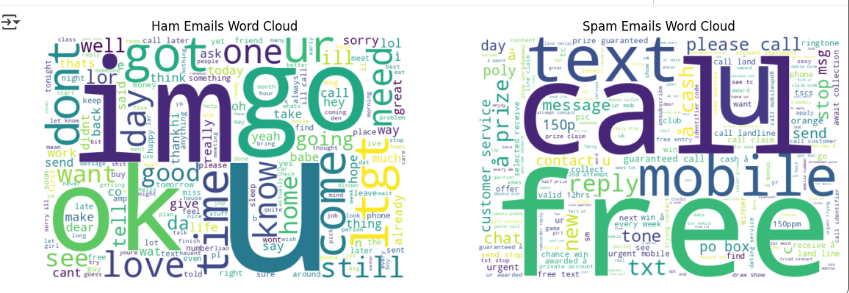
* Data Exploration



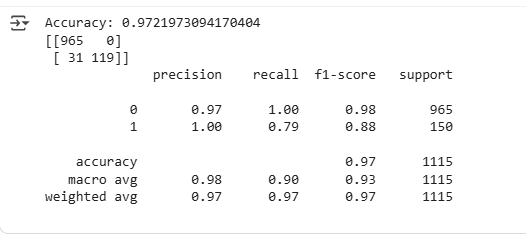
* Model Training



* Any feature importances / word clouds / insights visuals



* Evaluation



# Advantages

* Automatic and fast SMS classification.
* High accuracy and low false positives.
* Scalable for real-time detection.
* Lightweight and easy to deploy.
* Extendable for other text sources (emails, chat).

# Future Enhancements

* Use **Deep Learning** models (LSTM, BERT).
* Support **multilingual SMS.**
* Apply **SMOTE** for class balancing.
* Deploy as **web API or mobile app.**
* Integrate with **real-time spam filtering systems.**

# Conclusion

* This project successfully demonstrates the use of machine learning for SMS spam detection.
* The final model achieves high accuracy and reliability using simple NLP preprocessing and TF-IDF features.
* It provides a foundation for advanced spam detection systems, with potential for deployment as an online service or integrated app.