SMS Spam Detection - Model Documentation

1. Introduction

SMS spam detection is a crucial task in filtering unwanted messages. This project aims to classify SMS messages as **Spam** or **Not Spam (Ham)** using both traditional machine learning techniques and Transformer-based models.

2. Dataset Overview

The dataset consists of 10,000 SMS messages labeled as either "ham" (not spam) or "spam".

2.1 Data Structure

Each row contains:

- **Message:** The text content of the SMS.
- **Label:** "ham" for not spam, "spam" for spam messages.

2.2 Data Distribution

- **Ham (Not Spam):** 6,621 messages
- **Spam:** 3,379 messages

2.3 Message Length Statistics

Label	Count	Mean Length	Min	Max
Ham	6621	64.04	1	910
Spam	3379	164.53	13	453

Spam messages tend to be significantly longer than ham messages.

3. Data Preprocessing & Feature Engineering

3.1 Label Encoding

Labels were converted to numerical values:

- **"ham" → 0**

- **"spam" → 1**

3.2 Text Processing

- **TF-IDF Vectorization**: Extracts numerical features from text by transforming messages into weighted term frequency representations.
- **Message Length Feature**: The character length of each message was added as a feature.
- **Multilingual TF-IDF Vectorization**: Ensured compatibility with multiple languages for better spam detection.

4. Model Training & Evaluation

4.1 Machine Learning Models

We experimented with four traditional machine learning models:

Model	Accuracy
Logistic Regression	97.55%
Multinomial Naive Bayes	95.60%
Linear SVC	**98.65%**
Random Forest	98.25%

4.2 Best Performing Model

- **Linear SVC (98.65% accuracy)** had the best performance, with high precision and recall for both spam and ham classifications.

5. Transformer-Based Model Attempt

5.1 Approach

- Used **Multilingual BERT (mBERT)** for feature extraction.
- Included message length as an additional input feature.
- Defined a custom classifier on top of mBERT embeddings.

5.2 Challenges

- Due to **computational constraints**, training with transformers was slow and inefficient on the available hardware.
- A larger GPU with higher VRAM would be required for full fine-tuning, as training on CPU would take a long time.

6. Conclusion

- **Traditional ML models performed exceptionally well** and required less computational power.
- **Linear SVC achieved the highest accuracy (98.65%)** and is recommended for deployment.
- Transformer-based models could improve performance further but require better hardware.

7. Future Improvements

- Implement **efficient transformer models like DistilBERT** for better performance with lower resource requirements.
- Experiment with **hybrid approaches**, combining traditional ML with pre-trained embeddings.
- Improve handling of multilingual data with domain-specific preprocessing.