# **Project Report: SMS Spam Classification**

### **I. Problem Statement**

The rapid growth of mobile communication has led to a surge in SMS messages, many of which are unsolicited or spam. Spam messages not only waste user time but also pose risks such as phishing, scams, and malware links. The problem is to design a system that can automatically classify SMS messages into spam or non-spam categories by using computational methods, reducing the need for manual intervention and improving user safety.

### **II. Literature Survey**

**A. Rule-based Approaches** Early SMS filtering methods used keyword spotting and rule-based heuristics.  
 *Limitation*: Easily bypassed by variations in wording and required manual updates.

**B. Traditional Machine Learning** Algorithms like Naïve Bayes, SVM, and Decision Trees have been widely applied.  
 They rely on features such as bag-of-words, TF-IDF, and n-grams.  
 Works well with labeled datasets but performance depends heavily on feature engineering.

**C. Deep Learning Methods** CNNs and RNNs capture sequential and contextual patterns in messages.  
 Better at handling variations in spam text compared to traditional models.  
 However, they require large amounts of labeled data and high computational resources.

**D. Transfer Learning & Pre-trained Models** Recent approaches leverage models like BERT and other transformers.  
 They achieve state-of-the-art performance by understanding semantic context.  
 The drawback is that training and fine-tuning are computationally expensive.

**E. Challenges in Literature**

* Manual labeling of SMS datasets is time-consuming.
* Class imbalance problem: spam messages are much fewer than normal messages.
* Dynamic nature of spam (new patterns keep emerging).

### **III. Expected Outcome**

A functional SMS Spam Classification system capable of:

* Parsing and processing SMS data (from raw backups like XML).
* Converting messages into structured formats for classification.
* Automatically labeling messages as spam or non-spam using an AI-based model.
* Improved accuracy compared to traditional keyword-based filtering.
* Reduced manual labeling effort and better scalability for large datasets.
* Practical applications in mobile devices, telecom services, and security systems.

### **IV. Objectives**

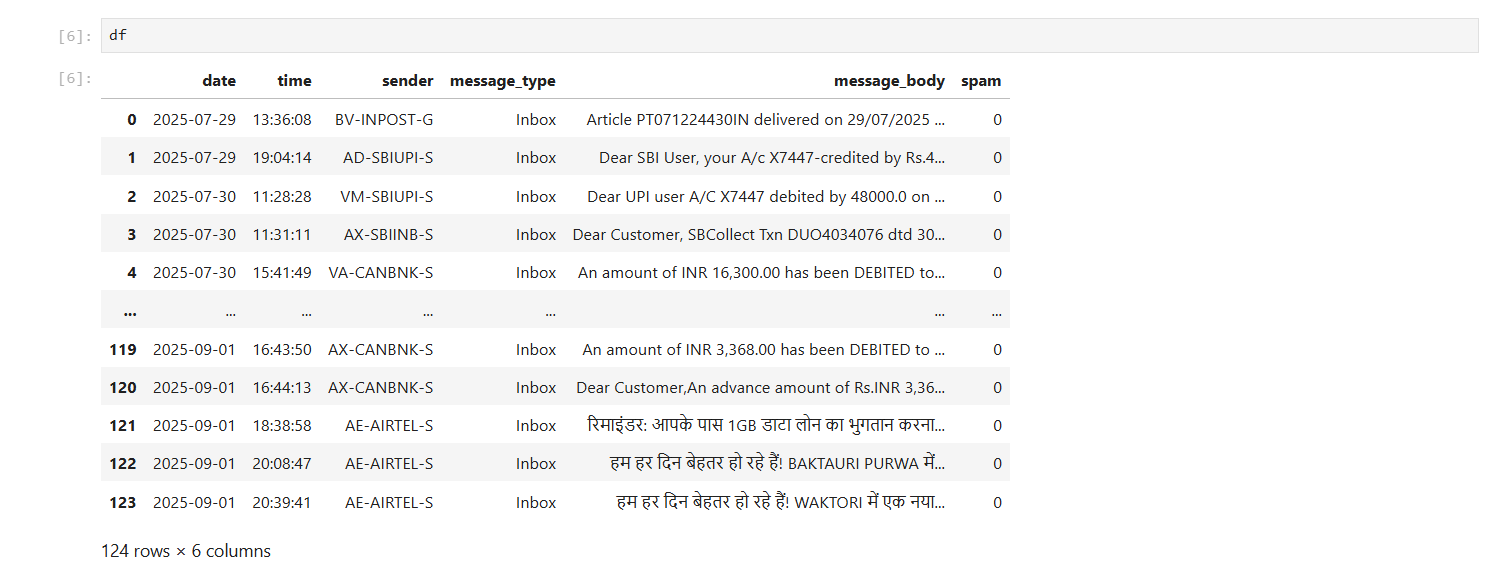
The objective of this project is to extract SMS messages from backups, process them into structured datasets, and apply automated classification to identify spam and scam messages reliably. The aim is to reduce manual effort, ensure consistency, and create a scalable solution for handling large SMS datasets.

### **V. Parsing SMS Data from XML Backup**

**A. Purpose** This step prepares the raw SMS dataset for classification by extracting messages from an XML backup file (e.g., from “SMS Backup & Restore”).

**B. Workflow**

1. Read XML File → Loaded the backup file (Rishabh.xml).
2. Message Type Mapping → Converted numeric message types (Inbox, Sent, Draft, etc.) into readable categories.
3. Parse and Clean Data → Extracted sender, timestamp (converted to date/time), message body, and type.
4. Default Spam Label → Assigned spam = 0 initially (to be updated later via classification).
5. Output → Converted all messages into a structured Pandas DataFrame and displayed sample rows.



**C. Error Handling**

* Handled invalid/missing dates gracefully.
* Replaced bad encoding characters to prevent crashes.
* Could fail if XML structure differs (e.g., <message> instead of <sms>).

### **VI. SMS Classification**

**A. Purpose** This step focuses on automatically classifying SMS messages into predefined categories such as:

* Phishing
* Smishing
* Promotional
* Loan/Financial Scam
* Job Scam
* Crypto/Investment Scam
* No Spam

**B. Workflow**

1. Load Dataset → CSV file containing SMS texts (message\_body).
2. Define Categories → Prepared a set of seven classification labels.
3. Classification Function → Used a language model to classify each SMS.
4. Iterative Processing → Processed messages one by one, adding a short delay to avoid rate limits.
5. Output → Exported results to annotated\_dataset.csv with predicted labels.

**C. Error Encountered** The script hit an API rate limit error (429):

Rate limit reached for requests per minute...

Limit 3 requests per minute.

This happened because the free tier allows only 3 requests/minute, but the dataset required many more queries.

### **VII. Challenges Faced**

* API rate limits restricted the number of SMS messages that could be classified per minute.
* XML backup formats can vary, and any deviation in structure may cause parsing issues.
* Some SMS messages may have missing or corrupted timestamps, requiring error handling.
* Large dataset sizes increase processing time and require automation.

### **VIII. Applications**

* Mobile spam filtering systems to protect users from scams and fraud.
* Telecom providers analyzing SMS data for fraudulent activity detection.
* Enterprise communication monitoring to prevent phishing or scam attempts.
* Research projects involving natural language processing on SMS datasets.

### **IX. Limitations**

* Classification relies on external language models which may have usage limits.
* Mixed or ambiguous SMS content can lead to misclassification.
* Dataset parsing depends on the specific XML structure, which may not be universal.
* The current setup does not support real-time SMS filtering.

### **X. Why Automated Labeling is Needed**

* **Volume of Data** → A typical SMS backup may contain thousands of messages. Manually reading and labeling each one is impractical.
* **Human Error** → Manual labeling is prone to mistakes and inconsistent category assignment.
* **Scalability** → As the dataset grows, the effort required increases exponentially, making manual labeling infeasible.
* **Time Constraint** → Labeling each message individually would take an enormous amount of time, which is not realistic for projects with deadlines.

### **XI. Conclusion**

This project shows how raw SMS data can be transformed into structured datasets and then automatically classified into spam and scam categories. By combining parsing and classification, the process becomes scalable and reliable. Manual labeling was avoided due to the high volume of data, inconsistency risks, and strict time constraints, making automation the practical choice.

### **XII. References**

[SMS Spam Collection Dataset](https://www.kaggle.com/datasets/uciml/sms-spam-collection-dataset)

[SMS Spam Collection - UCI Machine Learning Repository](https://archive.ics.uci.edu/dataset/228/sms+spam+collection)

[SMS Spam Detection using TensorFlow in Python - GeeksforGeeks](https://www.geeksforgeeks.org/deep-learning/sms-spam-detection-using-tensorflow-in-python/)

[SMS Spam Classification Using Machine Learning Techniques | IEEE Conference Publication](https://ieeexplore.ieee.org/document/9734128)

[📧SMS Spam Classification using Machine and Deep Learning | by Tahir | Medium](https://medium.com/@tahirbalarabe2/sms-spam-classification-using-machine-and-deep-learning-bee133e28bdb)

[Email Spam Classification Dataset CSV](https://www.kaggle.com/datasets/balaka18/email-spam-classification-dataset-csv)

[Email Spam Detection with Machine Learning: A Comprehensive Guide | by Azim Khan | Medium](https://medium.com/@azimkhan8018/email-spam-detection-with-machine-learning-a-comprehensive-guide-b65c6936678b)