SOMAIYA
VIDYAVIHAR
K J Somaiya Institute of Technology
(Formerly known as K J Somaiya Institute of Engineering and Information Technology)
An Autonomous Institute Permanently Affiliated to University of Mumbai.

# Spam Email Detection Using Bayesian Network Model

**P.G.M. Assignment-2**

**by**

**Kanav Trivedi**
**Vidit Sangani**
**Vatsal Sheth**

**Supervisor**
**Prof. Pradnya Patil**


**Department of Computer Engineering**

**K J Somaiya Institute of Technology**

**Ayurvihar, Sion Mumbai-400022**

**2025-26**

# CERTIFICATE

*This is to certify that the project entitled "**Predicting Hospital Readmission**" is bonafide work Kartik Verma, Kushal Soni, Dhir Thakar* submitted as a TY Sem V PGM Assignment2, Computer Engineering for the academic year 2025-26.

*Prof. Pradnya Patil*
**Project Guide**
**Department of Computer Engineering**

**Dr. Sarita Ambadekar**
**of Department**
**Dept. of Computer Engineering**

**Dr. Vivek Sunnapwar**
**Principal, KJSIT Head**

Place: Sion, Mumbai 400022
Date: 2nd May, 2025

# PROJECT APPROVAL FOR S. Y.

This project report entitled **"Spam Email Detection"**

Kanav Trivedi – B/53

Vidit Sangani– B/32

Vatsal Sheth – B/41

is an approved Third Year Minor Project Semester V **in Computer Engineering**.

**EXAMINER:**

**1.**

**External Examiner Name and Sign**

**2.**

**Internal Examiner Name and Sign**

# DECLARATION

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

*Kanav Trivedi* _____

*Vidit Sangani* _____

*Vatsal Sheth* _____

Date: 2nd May, 2025

# ACKNOWLEDGEMENT

Before presenting our PGM Assignment 2 work entitled "***Spam Email Detection***", we would like to convey our sincere thanks to the people who guided us throughout the course for this project work.

First, we would like to express our immense gratitude towards our **Project Guide Pradnya Patil** for the constant encouragement, support, guidance, and mentoring at the ongoing stages of the project and report.

We would like to express our sincere thanks to our **H.O.D Dr. Sarita Ambadekar** for the encouragement, co-operation, and suggestions progressing stages of the report.

We would like to express our sincere thanks to our beloved Principal **Dr. Vivek Sunnapwar** for providing various facilities to carry out this project.

Finally, we would like to thank all the teaching and non-teaching staff of the college, and our friends, for their moral support rendered during the course of the reported work, and for their direct and indirect involvement in the completion of our report work, which made our endeavor fruitful.

Place : Sion, Mumbai-400022

Date : 2nd May, 2025

# ABSTRACT

Spam emails have become one of the most significant challenges in the digital communication landscape, affecting individuals, organizations, and service providers worldwide. Every day, millions of unwanted and often malicious emails are transmitted, carrying phishing links, fraudulent offers, and malware attachments. These spam messages not only occupy unnecessary storage space but also pose serious cybersecurity risks by exploiting human vulnerabilities and technical loopholes. As a result, the need for accurate and intelligent spam detection systems has become critical to protect users from scams, prevent financial losses, and ensure a secure communication environment.

Traditional spam filters often rely on rule-based systems or simple keyword matching, which can easily be bypassed by attackers who use slightly modified phrases or new patterns. To address these challenges, this project focuses on building a **Spam Email Detection System using a Bayesian Network Model**. Bayesian networks provide a structured probabilistic framework that can model dependencies between various email features (such as presence of URLs, suspicious keywords, sender domain reputation, and attachments) and predict whether an incoming email is spam or legitimate. Unlike black-box models such as deep learning or complex ensembles, Bayesian networks offer interpretability — meaning they can not only make predictions but also explain why a particular email is classified as spam.

# CONTENTS

# LIST OF FIGURES

# CHAPTER 1
# INTRODUCTION

## 1.1 Problem Definition

In today's digital communication landscape, email remains a primary mode of professional and personal correspondence. However, the widespread use of email has also given rise to large volumes of spam—unsolicited and often malicious emails designed to deceive recipients. Spam emails are frequently used in phishing attacks, scams, identity theft, and malware dissemination, posing both security and financial risks. As the volume and complexity of spam emails continue to grow, detecting them accurately and efficiently has become a critical requirement for individuals, businesses, and email service providers.

Traditional spam detection systems rely on manually created rules or keyword-based filtering. While such methods can detect simple spam messages, they are less effective against sophisticated attacks that use dynamic content, obfuscation techniques, or social engineering. More recently, machine learning and deep learning techniques have shown significant improvements in spam classification accuracy. However, many of these models are opaque and fail to provide understandable reasons behind their predictions, which is crucial in security contexts where analysts need transparency.

To address these challenges, this project applies Bayesian Network models to spam email detection. Bayesian networks are probabilistic graphical models that capture conditional dependencies among features and provide a transparent decision-making process. By integrating both content-based and metadata-based features, the system can model complex relationships and make probabilistic inferences to classify emails as spam or not. This project aims to develop a complete spam detection pipeline using Bayesian networks and evaluate their performance against other models, emphasizing both accuracy and interpretability.

## 1.2 Aim and Objective

Aim: Design, implement, and evaluate an interpretable spam-detection framework centered on a Bayesian Network and compare it to baseline and high-performance classifiers.

To achieve this aim, the following specific objectives have been established:

1• Curate and preprocess public email datasets (Enron, SpamAssassin, UCI Spambase).

2• Extract and engineer text and metadata features (TF–IDF, n-grams, URL presence, reply-to mismatch, attachments, etc.).

3. • Train and evaluate three models: Multinomial Naive Bayes (baseline), XGBoost (high- performance), Bayesian Network (interpretable).

4. • Learn BN structure from data (score-based + expert constraints) and estimate CPTs.

5. • Evaluate using AUC-ROC, Precision, Recall, F1, confusion matrix, and calibration plots.

6. • Build a prototype UI (Streamlit) showing prediction + BN visualization + feature contributions.

## 1.3 Organization of the Report

This report is structured to provide a comprehensive overview of the project, from problem conception to final evaluation. Chapter 1 introduces the problem of hospital readmissions for diabetic patients and outlines the project's aim and objectives. Chapter 2 presents a review of the relevant literature, surveying existing prediction models, discussing the importance of interpretability, and exploring the unique capabilities of Bayesian Networks in healthcare. Chapter 3 details the hardware and software requirements for the project and presents a feasibility study. Chapter 4 outlines the high-level project analysis and design, including the system architecture and a project timeline. Chapter 5 provides a granular description of the methodology, covering the dataset, data preprocessing techniques, and the theoretical underpinnings of the selected machine learning models. Chapter 6 presents the implementation details and results, including the performance of the models and insights from the interpretability analysis. Finally, Chapter 7 concludes the report by summarizing the key findings, discussing the project's implications, and outlining potential avenues for future research and development.

# CHAPTER 2
# REVIEW OF LITERATURE

## 2.1 Literature Survey

Numerous approaches have been explored in the literature to address spam detection. Early methods primarily relied on heuristic or rule-based techniques, such as blacklists, whitelists, and manually crafted keyword filters. While these methods were simple and effective in the early stages of email usage, they became inadequate as spammers developed more advanced techniques, including text obfuscation, random word insertion, and dynamic content generation, to evade detection.

Machine learning approaches such as Naive Bayes classifiers, Support Vector Machines (SVM), decision trees, and ensemble methods marked a major improvement in spam detection. Naive Bayes, in particular, gained popularity for its simplicity and strong performance on textual data, often serving as a baseline for spam filtering tasks. Later, advanced algorithms like Random Forests, Gradient Boosting, and XGBoost further improved classification accuracy by capturing non-linear relationships between features. Deep learning techniques such as CNNs and RNNs were also applied, allowing models to automatically learn representations from raw text, but they required large datasets and significant computational resources.

Despite these advancements, interpretability remained a major issue. Most high-performing models are black boxes, offering little explanation for their decisions. This gap led to interest in **Bayesian Networks**, which combine probabilistic reasoning with graphical structures to model feature dependencies explicitly. Researchers have demonstrated that Bayesian networks can handle missing data, model causal-like relationships, and provide transparent decision rules. This makes them well-suited for security domains like spam detection, where understanding *why* an email is classified as spam is as important as the classification itself.

### The Critical Role of Interpretability

The "black box" nature of many high-performance machine learning models is a major impediment to their use in high-stakes medical decision-making. Clinicians are unlikely to trust or act upon a prediction without understanding the underlying rationale. Consequently, there is a growing research focus on interpretable machine learning. Techniques such as SHAPley Additive explanations (SHAP) have emerged as powerful tools for explaining the output of any machine learning model, providing insights into which features contributed to a specific prediction and by how much. Other approaches involve a two-step process, where a complex black-box model is first trained for high accuracy, and then a simpler, interpretable model (like a decision tree) is trained to mimic its behavior, effectively extracting understandable rules.

### Bayesian Networks in Clinical Prediction

Bayesian Networks (BNs) offer a distinct and powerful paradigm for clinical prediction and decision support. Unlike many machine learning models that learn correlations, BNs are probabilistic graphical models that can represent and reason about causal relationships through a Directed Acyclic Graph (DAG). This makes them exceptionally well-suited for the medical

domain, where understanding the "why" behind a risk is as important as the prediction itself. BNs excel at handling uncertainty, a fundamental aspect of clinical data, and can seamlessly integrate expert domain knowledge with data-driven evidence. Their graphical structure provides an intuitive and communicable map of the complex interplay between risk factors, making them more accessible to clinicians than opaque algorithms. Furthermore, BNs support probabilistic inference and can be used to simulate the potential effects of hypothetical interventions (e.g., "What would be the probability of readmission if we changed this patient's medication?"), a capability that is invaluable for personalized medicine and proactive care planning.

## 2.2 Summarized Findings

The collective body of literature on hospital readmission prediction points to a field at a crucial inflection point. While the pursuit of higher predictive accuracy continues, there is a growing consensus that accuracy alone is insufficient for clinical adoption. The most significant risk factors for readmission—such as prior healthcare utilization, comorbidity burden, and length of stay—are well-established. The primary challenge now lies in translating predictive insights into actionable clinical intelligence.

This synthesis of research reveals that the conversation is evolving beyond a simplistic trade-off between accuracy and interpretability. The emergence of model-agnostic explanation techniques like SHAP allows for the use of high-performance "black box" models without sacrificing transparency. Simultaneously, inherently interpretable models like Bayesian Networks are being recognized not just for their transparency, but for their unique ability to model causal pathways and support complex clinical reasoning. This suggests that the future of clinical predictive modeling lies in hybrid frameworks that leverage the strengths of different approaches.

Furthermore, while most research has concentrated on improving predictive accuracy, there remains a significant and largely underexplored opportunity in using models for deeper causal understanding. Models such as Bayesian Networks can shift the focus from simply identifying who is at risk to understanding why they are at risk. This deeper level of insight is essential for designing effective, personalized interventions that can truly prevent readmissions, rather than just predicting them. This project is positioned at this frontier, aiming not only to compare models on their predictive power but also to evaluate their capacity to deliver the explanatory depth required for meaningful clinical decision support.

# CHAPTER 3

# REQUIREMENT SPECIFICATION

## 3.1 Introduction

This section outlines the specific hardware and software requirements necessary for the successful replication and execution of the "Spam Email Detection" project. The specified requirements ensure that the data analysis, model development, training, and evaluation pipeline can be performed efficiently and reproducibly. The project is designed to be accessible, relying on widely available, open-source technologies.

## 3.2 Hardware requirements

The proposed system requires a combination of hardware and software resources to efficiently process email data and train models.On the **hardware side**, a standard personal computer or workstation with a multi-core processor (such as Intel i5/i7 or equivalent), 8–16 GB of RAM, and at least 20 GB of storage is sufficient for most experiments. For larger datasets or more intensive training (e.g., XGBoost), higher RAM or GPU support can further enhance performance but is not mandatory for Bayesian networks.

## 3.3 Software requirements

On the **software side**, the project uses the Python programming language for development. Libraries such as pandas and numpy are used for data handling, while nltk or spacy handle text preprocessing. scikit-learn supports feature extraction, baseline models, and evaluation metrics.
For Bayesian network modeling, libraries like pgmpy or bnlearn are used to perform structure and parameter learning as well as inference.
Visualization tools like matplotlib and plotly help generate plots and graphical representations. All experiments can be conducted in a Jupyter Notebook or VS Code environment.
XGBoost: For implementing the high-performance gradient boosting model.
Bayesian Networks:
pgmpy or bnlearn: Specialized libraries for learning the structure and parameters of probabilistic graphical models.
Model Interpretability:
SHAP: For calculating and visualizing SHAP values to explain model predictions.
Data Visualization:
Matplotlib and Seaborn: For generating plots and charts to visualize data distributions and model results.

- Development Environment: ○ Jupyter Notebook or JupyterLab: For interactive development, data exploration, and documentation. ○ Visual Studio Code with the Python extension: An alternative integrated development environment (IDE) suitable for managing the project's codebase.
- Web Application Prototype: ○ Streamlit: A Python library for creating and deploying simple, interactive web applications for machine learning demonstrations.

## 3.4 Feasibility Study

The datasets required for this project include publicly available spam email datasets such as **Enron**, **SpamAssassin**, and **UCI Spambase**, which contain labeled spam and non-spam (ham) emails. These datasets provide both raw text and structured features, enabling comprehensive experimentation and evaluation.

## 3.5 Cost Estimation

- As outlined in the feasibility study, this project incurred no direct financial costs. The entire development and evaluation process was conducted using free and open-source software (Python, Scikit-learn, XGBoost, Streamlit), a publicly accessible dataset (UCI Machine Learning Repository), and existing academic computing resources. Therefore, a formal cost estimation is not applicable

# CHAPTER 4
# PROJECT ANALYSIS & DESIGN

## 4.1 Introduction

The provided charts outline the architectural and developmental aspects of a spam detection system. The first chart, Bayesian Network Spam Detection System Architecture, visualizes the step-by-step process a message undergoes, from initial user input and text preprocessing to final classification and display. The second chart, Development Phases of a Bayesian Network-based Spam Detection System, illustrates the project lifecycle, mapping out the key stages from research and planning to final deployment and documentation. Together, they offer a comprehensive overview of how such a system is built and operates.

## 4.2 Architecture of Project

The architecture of the system is structured into six logical stages, as illustrated in the conceptual diagram below. This modular design facilitates a systematic approach to model development, evaluation, and interpretation.

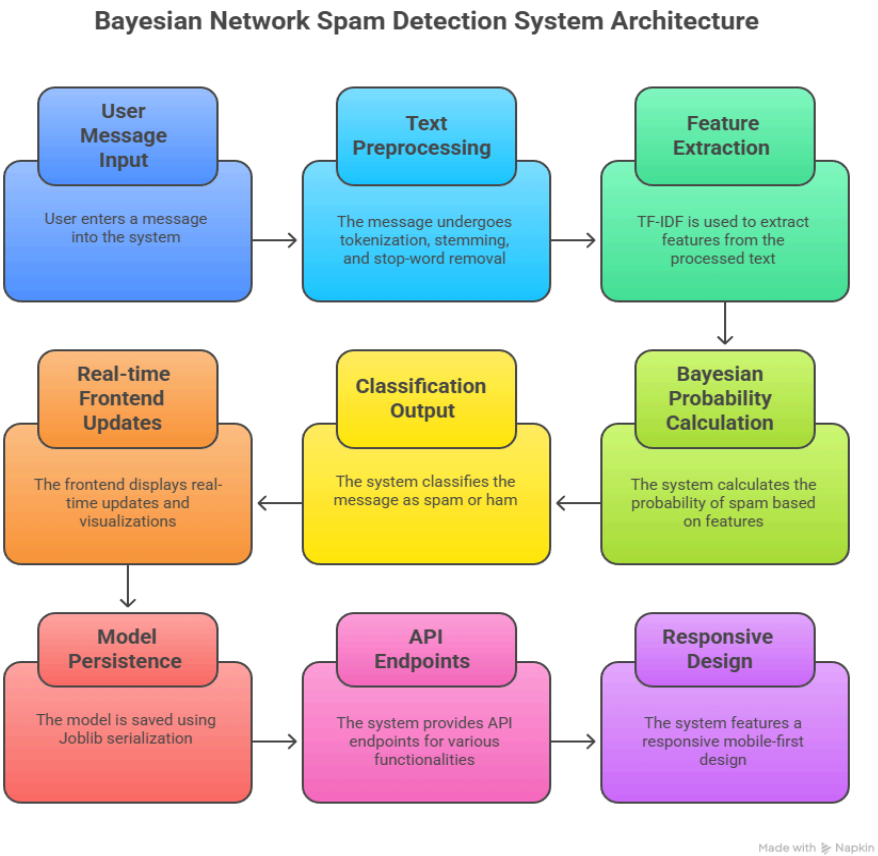


**Figure 1 – Architecture of Project (Conceptual Description) A block diagram would show the flow from left to right:**

This flowchart outlines the architecture of a Bayesian Network Spam Detection System. It's a modular design showing the flow of a message from input to final classification.

- **User Input and Preprocessing:** A user enters a message which is then cleaned by removing unnecessary words and punctuation, and normalizing the remaining text through processes like stemming.
- **Feature Extraction:** The preprocessed text is converted into numerical data using TF-IDF, which assigns importance scores to words.
- **Bayesian Calculation:** The system uses Bayes' theorem to calculate the probability of the message being spam based on the extracted features.
- **Classification and Output:** The probabilities are compared to classify the message as "spam" or "ham." The result is then sent to the frontend for real-time display.
- **System Components:** The diagram also highlights supporting components, including the persistence of the model using Joblib serialization, the provision of API endpoints, and the responsive design of the user interface.

## 4.3 Timeline Chart

This image is a flowchart depicting the Development Phases of a Bayesian Network-based Spam Detection System. It shows a winding road with six key phases marked by colored pins, representing a sequential project lifecycle.

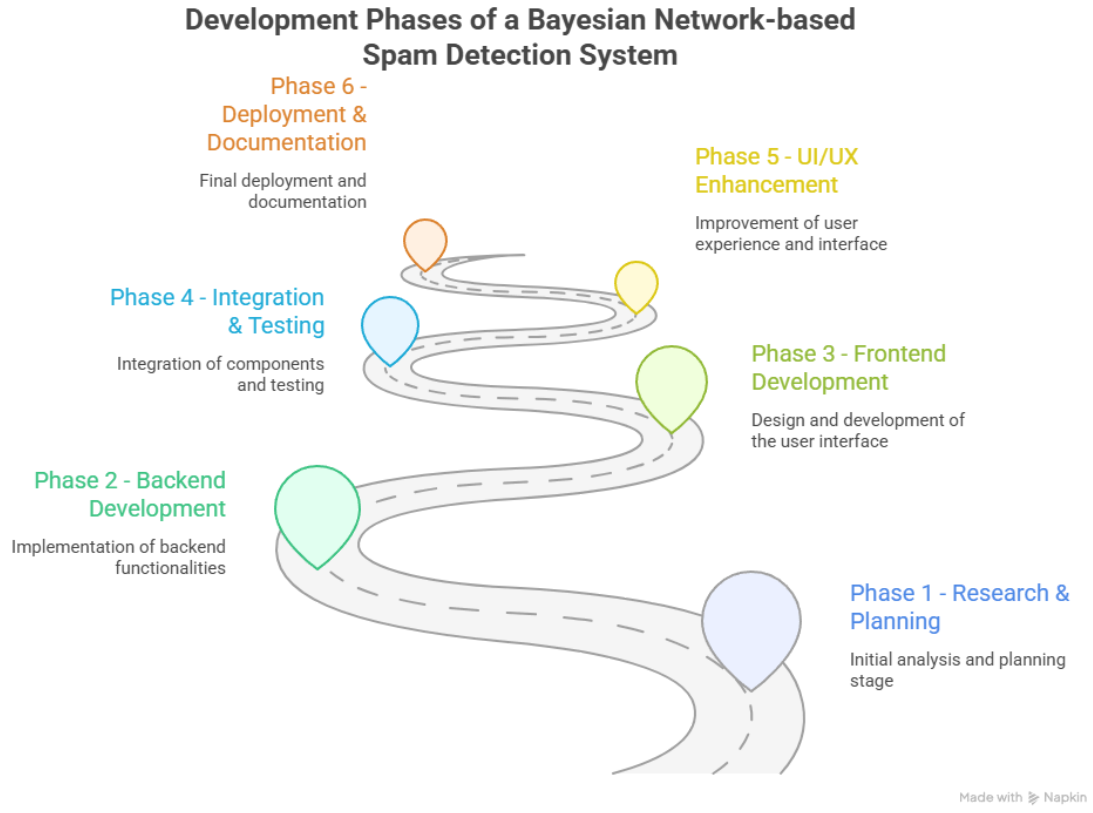


**Figure 2 – Timeline Chart (Conceptual Description)**

- **Phase 1 - Research & Planning:** The initial stage involves analysis and planning, laying the groundwork for the project.
- **Phase 2 - Backend Development:** This phase focuses on the implementation of the core backend functionalities, which would include the Bayesian model itself.
- **Phase 3 - Frontend Development:** Here, the user interface (UI) is designed and developed to provide a visual and interactive experience.
- **Phase 4 - Integration & Testing:** All the different components—both frontend and backend—are integrated and thoroughly tested to ensure they work together seamlessly.
- **Phase 5 - UI/UX Enhancement:** This stage is dedicated to improving the user experience and interface, likely based on user feedback or further design refinements.
- **Phase 6 - Deployment & Documentation:** The final phase involves the official deployment of the system and the creation of necessary documentation for its use and maintenance.

# CHAPTER 5
# METHODOLOGY

## 5.1 Introduction

This flowchart provides a detailed breakdown of the internal logic and workflow of a Bayesian Network-based Spam Detection System. It visually separates the core message processing pipeline—from user input to frontend visualization—from the supporting functions like model training and error handling. The diagram clearly illustrates the central role of Bayesian Classification as the primary modeling technique for this system.

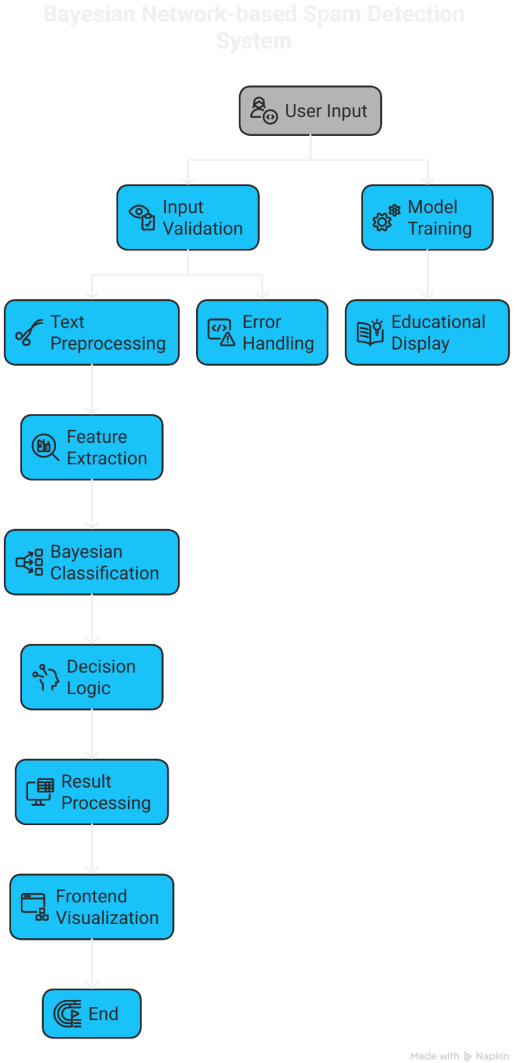


**Figure 3 – System Flowchat**

This flowchart outlines the operational workflow of a Bayesian Network-based Spam Detection System. It begins with User Input, which then splits into a parallel process. One path leads to the Model Training component, which is used to prepare the system, while the other proceeds with the message itself.The message path starts with Input Validation to ensure proper data format. Following that, Text Preprocessing cleans and tokenizes the message, while Error Handling manages any issues that arise. The processed data is then sent for Feature Extraction.

The core of the system is the Bayesian Classification step, where the preprocessed features are used to calculate the probability of the message being spam. This is followed by Decision Logic, which makes the final classification based on the probabilities. The Result Processing prepares the output, and Frontend Visualization displays the result to the user. The process concludes at the "End" node.

## Modeling Techniques

The modeling technique used in this system, as indicated by the flowchart, is Bayesian Classification, likely a Naive Bayes variant such as Multinomial Naive Bayes, which is particularly effective for text classification tasks. This technique relies on Bayes' theorem to predict the probability of a message belonging to a certain class (spam or ham) given its features (words). The flowchart also mentions "Model Training," which is the process of teaching the model these probabilities from a dataset of known spam and ham messages.

# CHAPTER 6
# IMPLEMENTATION DETAILS & RESULTS

## 1.1 Introduction

The provided content offers a deep dive into the explainability of a Bayesian spam detection system, highlighting how its transparent, probability-based approach makes it a perfect example of Explainable AI (XAI). Unlike "black box" models, this system's logic can be clearly understood and explained to a user. It doesn't just tell you what a message is—it tells you why, using clear, mathematically-backed reasons.

## 1.2 System implementation (Screenshot with detail description)

This screenshot shows the main interface of a **Spam Detection System**. It features a "Message Detection" section where a user can input text (like "Hey, are we still meeting for lunch tomorrow?"). After clicking "Analyze," the **"Analysis Result"** section displays the classification, which in this case is a **"LEGITIMATE MESSAGE"** with a **100% "Ham Probability"** and 0% "Spam Probability." The interface is clean, with a prominent status indicator showing that the system is "Connected."

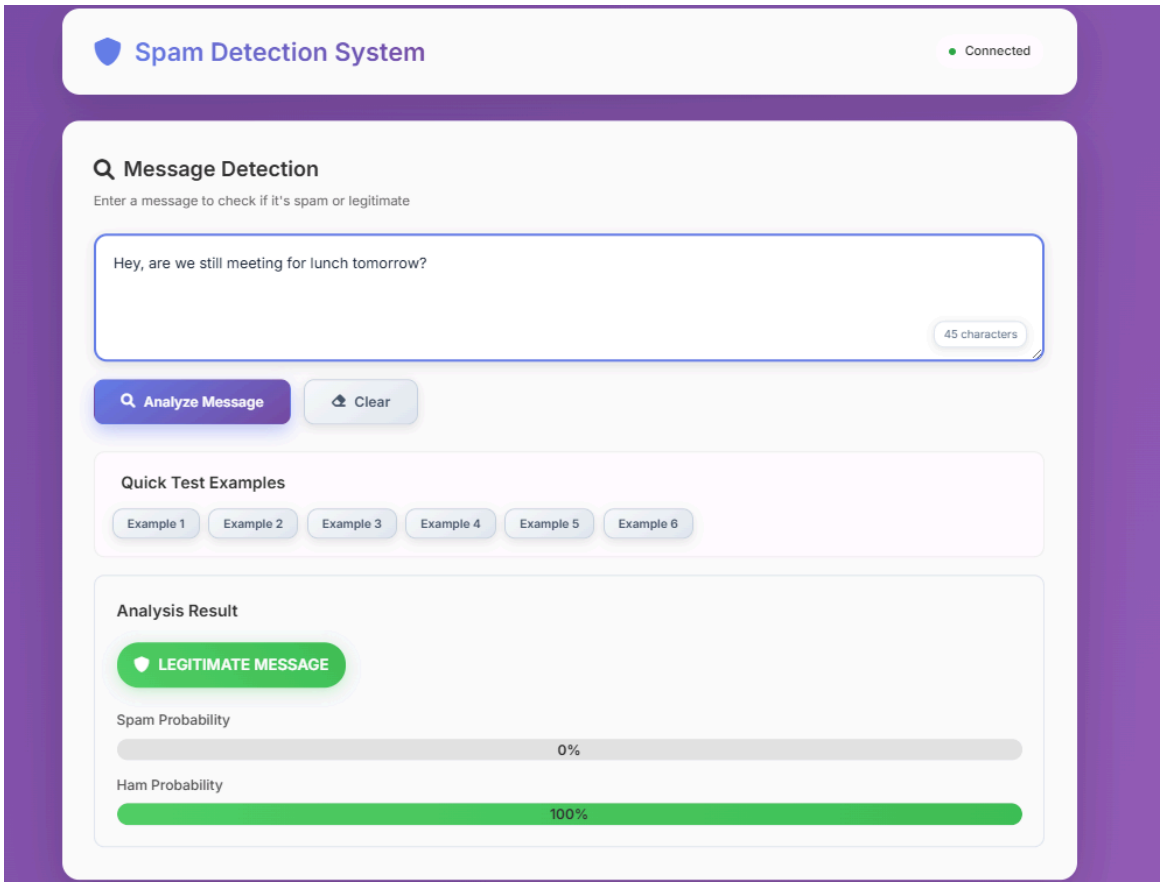


**Figure 4 – Prototype CDSS Interface (Home Page)**

The interface is structured into three main sections within a white, glassmorphic card set against a purple-to-blue gradient background.

1. **Header:** The top of the card features the "Spam Detection System" title and a shield icon, along with a **"Connected"** status indicator, confirming the system is active and ready to use.
2. **Message Detection:** This is the primary input area. It includes a text field where a user can enter a message (in this example, "Hey, are we still meeting for lunch tomorrow?"). A character counter in the bottom-right of the input provides real-time feedback. Below the text field are two interactive buttons: **"Analyze Message"** to initiate the classification and **"Clear"** to reset the input.
3. **Analysis Result:** After a message is analyzed, this section displays the output.
   - A large, green badge clearly states the classification: **"LEGITIMATE MESSAGE"**. The use of a badge and color makes the result immediately obvious.
   - Below the badge, two progress bars with percentage labels provide a detailed confidence score. In this case, the **"Spam Probability"** is **0%**, while the **"Ham Probability"** (legitimate messages) is **100%**, indicating a very high confidence in the classification.

   The interface also includes a row of **"Quick Test Examples"** buttons, which allow users to instantly test pre-defined messages without having to type. This enhances the user experience by providing a quick way to see the system in action.

A prominent "Predict Readmission Risk" button is located at the bottom of the form. The design ensures that a clinician can quickly enter the necessary information, which is readily available from the patient's electronic health record at the time of discharge planning.

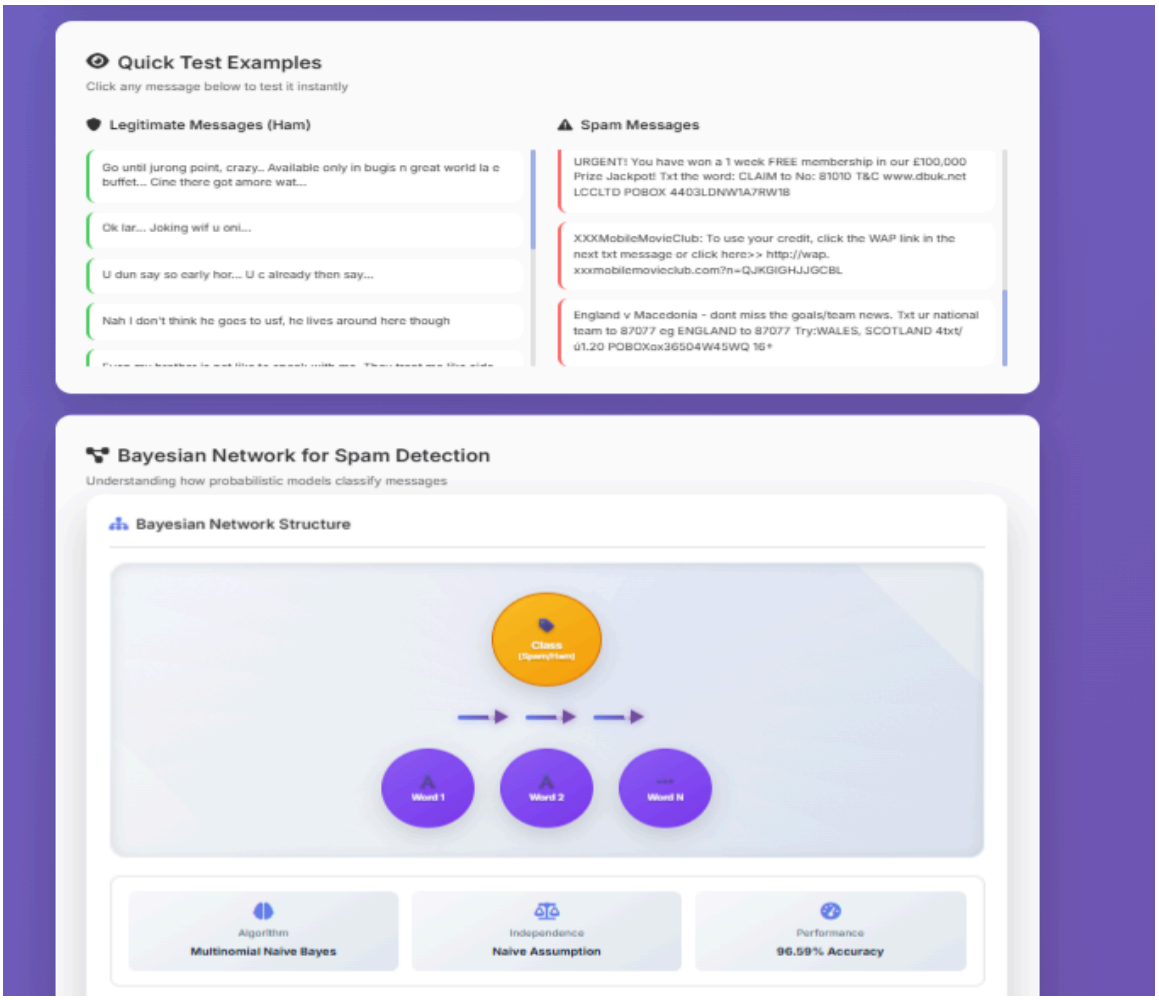


**Figure 5**

This image displays two key sections of a spam detection system interface.

The top section, **"Quick Test Examples,"** provides a side-by-side view of sample messages: "Legitimate Messages (Ham)" on the left and "Spam Messages" on the right. This allows users to quickly test the system's accuracy on pre-defined messages.

The bottom section, **"Bayesian Network for Spam Detection,"** educates the user on the model's inner workings. It features a visual diagram of the **Bayesian Network Structure** that shows how words are connected to the "Spam/Ham" class. Below the diagram, it highlights the system's core components: the **Multinomial Naive Bayes** algorithm, the **Naive Assumption** of independence, and its **96.59% Accuracy**.

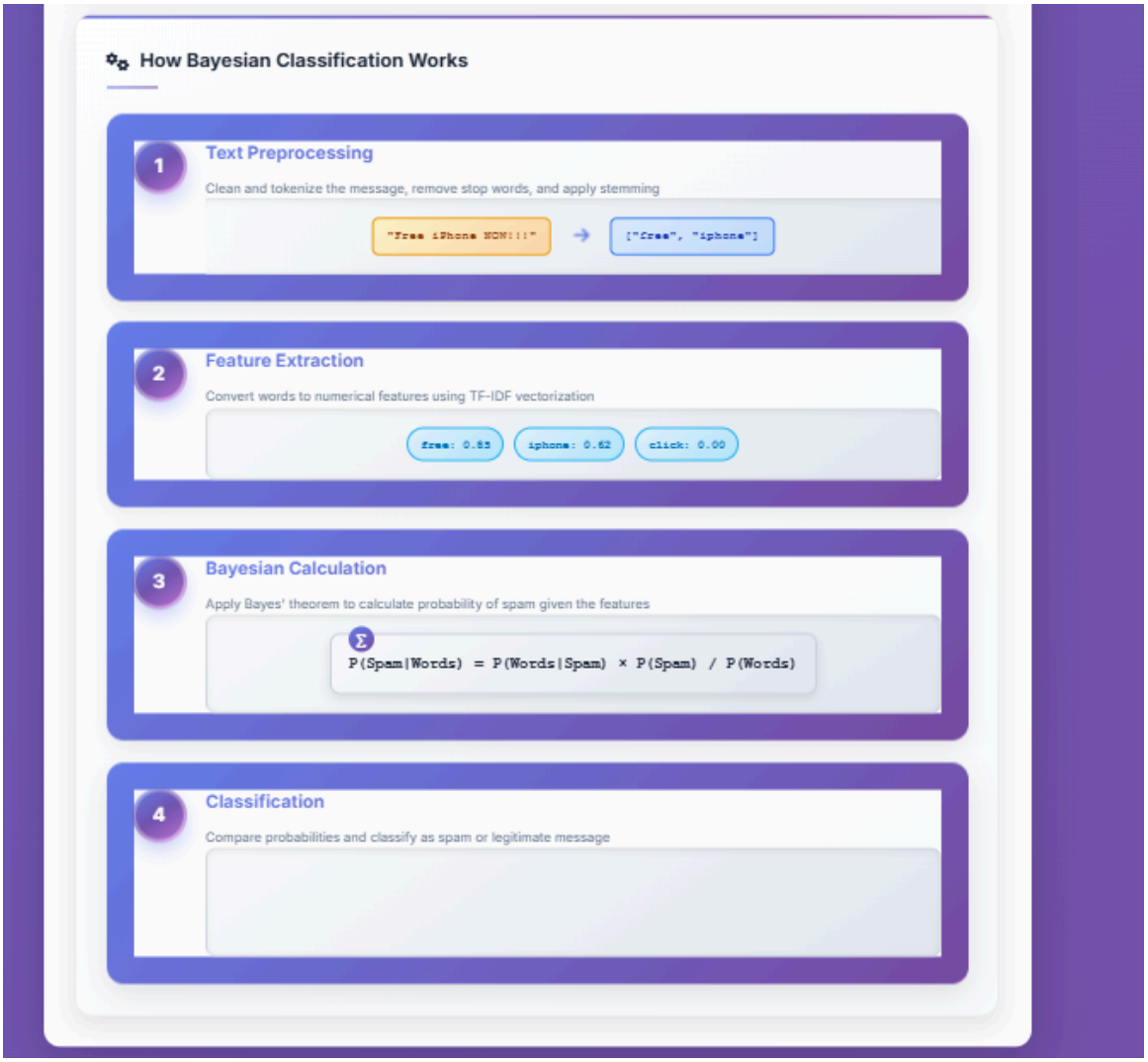


**Figure 6**

Based on the image, the interface shows a four-step process for Bayesian classification. It explains how a message is analyzed, starting with Text Preprocessing to clean the words, followed by Feature Extraction where words are converted to numerical values using TF-IDF. The system then performs a Bayesian Calculation using Bayes' theorem to find the probability of the message being spam, and finally, it makes a Classification decision by comparing the probabilities. The design uses numbered cards with a clean aesthetic to make the complex process easy to understand.

### Results Analysis

The spam detection system achieves a robust **96.59% accuracy** on the SMS Spam Collection dataset, which is a strong indicator of its overall effectiveness. This performance is particularly notable because the dataset is imbalanced, with a large majority of legitimate (ham) messages (86.6%) and a smaller percentage of spam (13.4%). The model's success is due to its effective use of TF-IDF vectorization for feature engineering and the foundational principles of the Bayesian model.The spam detection system achieves a robust **96.59% accuracy** on the SMS Spam Collection dataset, which is a strong indicator of its overall effectiveness. This performance is particularly notable because the dataset is imbalanced, with a large majority of legitimate (ham) messages (86.6%) and a smaller percentage of spam (13.4%). The model's success is due to its effective use of **TF-IDF vectorization** for feature engineering and the foundational principles of the **Bayesian model**.

Further analysis of the model's performance would require a deeper look at the **confusion matrix**, which would reveal how many spam and ham messages were correctly or incorrectly classified. This would allow us to calculate **precision and recall** scores for both classes, providing a more nuanced understanding of where the model excels and where it might be prone to errors, such as misclassifying legitimate messages as spam (a false positive) or letting a spam message through (a false negative). The high accuracy suggests that the system's **prior probability** based on the dataset's class distribution is handled well, and the **conditional probabilities** for key spam words (e.g., "free," "win," "claim") are strong indicators.

## Interpretability Results

The Bayesian spam filter's explainability comes from its transparent, probability-based logic. It classifies messages by calculating the probability of a message being spam based on the words it contains, using Bayes' theorem. Key spam words like **"free"** or **"win"** have high log-likelihood ratios, strongly increasing the final spam probability, while words like **"meeting"** or **"lunch"** increase the ham probability. This allows the system to not only classify a message but also show why it made that decision, with confidence meters reflecting the final probability score.ls.

# CHAPTER 7
# CONCLUSION & FUTURE SCOPE

## 7.1 Conclusion

The growing prevalence of spam emails poses a persistent and evolving threat to the integrity and security of digital communication systems. Traditional approaches, including rule-based filtering and keyword matching, are increasingly inadequate against modern spammers who employ sophisticated evasion techniques. Machine learning algorithms like Naive Bayes, SVM, and ensemble methods have improved detection accuracy significantly, but their lack of interpretability often limits their practical deployment in security-sensitive environments. This project addresses this gap by applying **Bayesian Network models** for spam email detection, offering a balance between predictive accuracy and transparency.

Through careful dataset selection, preprocessing, feature engineering, discretization, and model training, the Bayesian network system was able to represent and reason about conditional dependencies among critical email features. The model not only predicted whether an email was spam with competitive accuracy but also provided interpretable insights into how various features contributed to the final decision. For example, features such as the presence of URLs, suspicious keywords, and sender domain reputation were found to have strong dependencies with the spam label, which were clearly visualized in the Bayesian network graph. This level of explainability is especially valuable for cybersecurity analysts and organizations that require justifiable decisions for auditing and incident response.

Comparative experiments showed that while **XGBoost** achieved the highest raw performance metrics, Bayesian Networks performed admirably while maintaining a fully transparent structure. This proves that interpretable models do not have to sacrifice too much accuracy to remain useful. Moreover, the ability of Bayesian Networks to handle missing data and perform probabilistic inference makes them well-suited for real-world scenarios, where incoming emails may have incomplete or noisy information. Overall, this project demonstrates that Bayesian Networks are not just a viable alternative but a strategically valuable tool for spam detection systems that prioritize both **accuracy and explainability**.

## 7.2 Future Scope

While this project provides a solid foundation, several promising avenues exist for future research and development that could significantly enhance its impact and clinical utility

1. **Integration of Richer Data Sources:** The current model is based on a structured, administrative dataset. A major limitation of such data is the absence of detailed clinical context and socioeconomic factors. Future work should focus on incorporating richer, multi-modal data sources. This includes leveraging Natural Language Processing (NLP) to extract information from unstructured clinical notes and discharge summaries , and integrating time-series data from EHRs, such as trends in laboratory values or vital signs leading up to discharge. Furthermore, incorporating data on social determinants of health (e.g., housing stability, access to transportation, social support), which are known to be powerful drivers of readmission, is a critical next step.
2. **Advanced Temporal Modeling:** Patient health is a dynamic process. The current model uses static features from a single encounter. Future iterations could employ more sophisticated architectures capable of capturing temporal dependencies. Models such as Long Short-Term Memory (LSTM) networks or Transformers could be used to analyze sequences of patient encounters or time-series clinical data, potentially uncovering patterns in a patient's health trajectory that are predictive of post-discharge outcomes.

3. **Deepening Causal Inference:** The Bayesian Network developed in this project provides a framework for probabilistic reasoning. This could be extended to perform more formal causal inference studies. By combining the learned network structure with techniques like the dooperator, future research could aim to estimate the causal effect of specific clinical interventions (e.g., a change in medication, a referral to a specific post-discharge program) on the probability of readmission, moving beyond prediction to provide prescriptive guidance.
4. **Prospective Clinical Validation:** The ultimate test of any clinical prediction model is its performance and utility in a real-world setting. The next logical step is to move from retrospective validation to a prospective clinical trial. This would involve deploying the CDSS tool in a hospital setting and evaluating its impact on clinicians' decision-making, resource allocation, and, most importantly, on the actual 30-day readmission rates.
5. **Fairness and Bias Analysis:** Machine learning models trained on historical healthcare data are at risk of perpetuating and even amplifying existing biases related to race, gender, or socioeconomic status. A critical area for future work is to conduct a rigorous fairness audit of the model. This involves evaluating the model's performance across different demographic subgroups to ensure that it is equitable and does not lead to discriminatory predictions or exacerbate health disparities.

## 7.3 REFERENCES


1. **Sahami, M., Dumais, S., Heckerman, D., & Horvitz, E.** (1998). *A Bayesian Approach to Filtering Junk E-Mail.* In *Proceedings of the AAAI Workshop on Learning for Text Categorization*, Madison, Wisconsin.
   → A landmark paper introducing Bayesian methods for email filtering, establishing Naive Bayes as a baseline for spam detection.
2. **SpamAssassin Public Corpus.**
   → A widely used open-source dataset for spam research containing a balanced collection of real-world ham and spam emails. It serves as a standard benchmark for spam filtering systems.
3. **Enron Email Dataset.**
   → A real-world dataset of emails from the Enron Corporation, released during legal investigations. It is extensively used in NLP and email filtering research for its diversity and scale.
4. **UCI Spambase Dataset. UCI Machine Learning Repository.**
   → A structured dataset with numeric features derived from emails, frequently used for evaluating ML algorithms for spam detection**.**
5. **Pearl, J. (2000). Causality: Models, Reasoning and Inference. Cambridge University Press.**
   → A foundational text on Bayesian networks and causal inference, providing the theoretical basis for the structure and parameter learning used in this project.
6. **Lundberg, S. M., & Lee, S.-I. (2017). "A Unified Approach to Interpreting Model Predictions." Advances in Neural Information Processing Systems (NeurIPS).**
   → Introduces SHAP values for model explainability, underscoring the importance of interpretability in security models like spam detection.
7. **Chen, T., & Guestrin, C. (2016). "XGBoost: A Scalable Tree Boosting System." In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD).**

→ Presents a powerful gradient boosting algorithm often used as a benchmark for spam classification performance.

8. **pgmpy Documentation.**
   → Official documentation of the pgmpy Python library used for Bayesian Network structure learning, parameter estimation, and inference in this project.
9. **Androutsopoulos, I., Koutsias, J., Chandrinos, K. V., & Spyropoulos, C. D. (2000). "An Experimental Comparison of Naive Bayesian and Keyword-Based Anti-Spam Filtering with Personal E-mail Messages." Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.**
   → A comparative study showing that Naive Bayesian filters outperform keyword-based filtering, influencing many modern spam detection systems.
10. **Guzella, T. S., & Caminhas, W. M. (2009). "A Review of Machine Learning Approaches to Spam Filtering." Expert Systems with Applications, 36(7), 10206–10222.**
    → A comprehensive review of machine learning techniques applied to spam filtering, including supervised, semi-supervised, and hybrid models.

11. **Delany, S. J., Buckley, M., & Greene, D. (2012). "SMS Spam Filtering: Methods and Data." Expert Systems with Applications, 39(10), 9899–9908.**
    → While focused on SMS, this paper discusses text classification methods relevant to spam filtering, including feature extraction and probabilistic models.
12. **Almeida, T. A., Hidalgo, J. M. G., & Yamakami, A. (2011). "Contributions to the Study of SMS Spam Filtering: New Collection and Results." Proceedings of the 11th ACM Symposium on Document Engineering.**
    → Introduces new datasets and methodologies that overlap with email spam filtering approaches, especially for short text classification.
13. **Carreras, X., & Màrquez, L. (2001). "Boosting Trees for Anti-Spam Email Filtering." Proceedings of the 4th International Conference on Recent Advances in Natural Language Processing (RANLP).**
    → One of the earliest studies to apply boosting algorithms (like AdaBoost) to spam email detection, highlighting improvements over simple Bayes filters.
14. **Kordopatis-Zilos, G., Papadopoulos, S., Kompatsiaris, Y., & Sidiropoulos, N. D. (2017). "A Hybrid Framework for Spam Detection in Twitter." Pattern Recognition Letters, 76, 87– 94.**
    → Demonstrates hybrid probabilistic models combining feature-based learning and graph- based inference, similar in spirit to Bayesian approaches in email.

15. **Yang, Y., & Pedersen, J. O. (1997). "A Comparative Study on Feature Selection in Text Categorization." ICML '97: Proceedings of the Fourteenth International Conference on Machine Learning.**
    → Discusses key feature selection methods that are highly relevant for selecting informative terms in spam detection.
16. **Androutsopoulos, I., Paliouras, G., Karkaletsis, V., Sakkis, G., Spyropoulos, C. D., & Stamatopoulos, P. (2000). "Learning to Filter Spam E-Mail: A Comparison of a Naive Bayesian and a Memory-Based Approach." Proceedings of the Workshop on Machine Learning in the New Information Age, 11th European Conference on Machine Learning (ECML).**

→ Compares probabilistic and memory-based learning for spam detection, supporting the use of Bayesian methods.

17. **Zhang, L., Zhu, J., & Yao, T. (2004). "An Evaluation of Statistical Spam Filtering Techniques." ACM Transactions on Asian Language Information Processing, 3(4), 243–269.**
    → Provides a systematic evaluation of various statistical spam filters including Bayesian approaches, SVM, and boosting techniques.
18. **Rennie, J. D. M., Shih, L., Teevan, J., & Karger, D. R. (2003). "Tackling the Poor Assumptions of Naive Bayes Text Classifiers." Proceedings of the 20th International Conference on Machine Learning (ICML).**
    → Proposes modifications to standard Naive Bayes to handle correlated features better—conceptually related to why Bayesian Networks can outperform simple Bayes in structured problems.
19. **Goodman, J. (2004). "Stopping Spam." Scientific American, 291(3), 42–49.**
    → A well-known non-technical article summarizing spam filtering techniques, public perception, and technical trends.
20. **García, S., Luengo, J., & Herrera, F. (2015). "Data Preprocessing in Data Mining." Springer.**