Enhancing Naive Bayes using techniques like auxiliary features and TF-IDF weighting

Raghav Kamineni, Lavanya Krishna

Introduction

EMAIL SPAM & NAIIVE BAYES

- Email is a key communication tool for personal and business exchanges.
- Spam messages pose security risks, leading to phishing attacks and malware.
- Detecting spam is crucial for maintaining a safe and trustworthy digital environment.
- Naïve Bayes Classifier is widely used for spam detection:
  - Advantages: Simple, efficient, and effective for text classification.
  - Limitation: Assumes independence of words, reducing accuracy in real-world scenarios.

OUR RESEARCH

- Goal: Enhance Naïve Bayes for better email spam classification.
- Dataset: Using the SMS Spam Collection Dataset.
- Key Feature Enhancements:
  - TF-IDF Weighting: Emphasizes important words while reducing common noise.
  - Word2Vec Embeddings: Captures relationships between words for better context understanding.
  - Information Gain: Selects the most relevant features to improve classification.
- Process Overview:
  - Input: Raw spam dataset → Preprocessing (TF-IDF, Word2Vec, Information Gain) → Improved Naïve Bayes Classifier → Spam or Not Spam Output.
- Expected Outcome: Higher accuracy, better feature representation, and a more robust spam filter.

Related Research

RELATED RESEARCH

- Zhang & Gao (2011): Introduced an auxiliary feature method to improve classification accuracy.
  - Strengths: Dynamically adjusts conditional probabilities.
  - Limitations: Does not explicitly model word dependencies.
  - Our work: Incorporates dependency graphs to improve classification.
- Li & Li (2020): Proposed an improved TF-IDF weighting scheme.
  - Strengths: Increases weights of high-frequency spam words, reducing feature space.
  - Our work: Uses their TF-IDF improvement along with additional feature selection strategies.

RELATED RESEARCH

- Das & Chakraborty (2018): Showed TF-IDF improves classification by reducing the weight of common words.
  - Introduced Next Word Negation (NWN) to capture negated phrases.
  - Our work: Applies TF-IDF & explores negation-aware preprocessing for spam filtering.
- Aninditya et al. (2021): Applied Naïve Bayes + TF-IDF + N-grams for exam question classification.
  - Strengths: Achieved 85% precision & 80% recall using N-grams.
  - Our work: Uses N-gram analysis to capture repeated spam patterns.
- Rasjid & Setiawan (2017): Explored feature selection techniques (Information Gain, Chi-Squared).
  - Our work: Experiments with feature selection to remove redundant terms.

Dataset & Preprocessing

Dataset & Preprocessing

- Initial Dataset:
  - Contained 5 columns: v1, v2, and three empty columns.
  - First step: Removed the last three empty columns to clean the dataset.
- Processing v1 (Class Attribute):
  - Originally labeled as "spam" and "ham".
  - Converted to binary values: "spam" → 1, "ham" → 0.
  - Allows Naïve Bayes to process numerical values more efficiently.
- Processing v2 (Email Text):
  - Converted all text to lowercase to avoid duplicate word variations (e.g., “Free” vs. “free”).
  - Removed special characters (e.g., periods, commas) to prevent punctuation from misleading the classifier.
- Tokenized emails:
  - Broke text into individual words rather than treating an entire email as one entity.
  - Helps Naïve Bayes recognize important word frequencies.

Methods & Experimental Setup

Naive Bayes Classifier

Our beginnings

- Assumes independence of words → Not always true!
- Implementation (SKLearnNaiveBayes.py):
  - Used CategoricalNB classifier
  - Bag of Words (BoW) for text vectorization
  - Discretized features using KBinsDiscretizer
  - Limitation: Doesn't model dependencies between words

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)}$$

Bayesian Network with Dependency Graphs

- Fixing the independence assumption with Bayesian Networks
- Implementation (DependencyGraph.py):
  - Selected high-frequency words
  - Defined directed acyclic graph (DAG) for dependencies
  - Used Maximum Likelihood Estimation (MLE) for training
  - Performed Variable Elimination for inference
- Captures word co-occurrence and dependencies

$$P(Y, X_1, X_2, ..., X_n) = P(Y) \prod_i P(X_i | \text{Parents}(X_i))$$

Feature Engineering Enhancements

- TF-IDF Weighting
  - Replaces raw word counts to improve feature representation
  - Effect:
    - Reduces weight of common words like "is"
    - Highlights informative words

$$\text{TF-IDF}(w, d) = \text{TF}(w, d) \times \log \frac{N}{\text{DF}(w)}$$

- Word Embeddings (Word2Vec)
  - Captures word meaning and context
  - Converts emails into dense vector representations

$$P(w_t | w_{t-2}, w_{t-1}, w_{t+1}, w_{t+2})$$

- Feature Selection
  - Removes irrelevant features using:
    - Information Gain (IG) – Measures how much a word contributes to classification
    - Chi-Squared Test – Identifies words strongly associated with spam

EXPERIMENTAL SETUP

- Dataset: Standard spam classification dataset
- Train-Test Split:
  - 80% training, 20% testing
  - 5-fold cross-validation for hyperparameter tuning
- Implementation Tools:
  - Scikit-learn (Naïve Bayes, TF-IDF, feature selection)
  - pgmpy (Bayesian Networks)
  - Gensim (Word2Vec)

RESULTS

Our Metrics

- Accuracy
  - Proportion of Correct to total predictions
- Precision
  - Proportion of True Positives to all Positive Predictions
- Recall
  - Proportion of Correct Positive Prediction to total Positive Predictions
- F-1 Score
  - The Harmonic mean of precision and recall

Accuracy alone does not account for class imbalances, Important to use other metrics too.

	Initial Naive Bayes	SKLearn	Dependency Graphs	TF-IDF Weighting	Word2Vec	Combined model
Accuracy	94.8%	90.49%	86.73%	96.5%	97.49%	97.49%
Precision	85.3%	100%	53.8%	97.4%	89.6%	89.6%
Recall	74%	29.3%	9.3%	76%	92%	92%
F-1 Score	79.2%	45.4%	15.9%	85.3	90.8	90.8

STRENGTH

SKLearn

Very Precise, Every spam classified was actually spam. However too conservative and didn't detect other spams

Dependency Graphs

Probabilistic structure made it too unreliable, High False Negatives

TF-IDF Weighting

High accuracy, with low false positives. Good precision based option.

Word2vec

Best overall, 92% of all spam emails, with a low false positive rate. Best Recall focused option

Ensemble

Wasn't able to Aggregate algorithm to improve accuracy.

Conclusion
& Future Work

Conclusion/Future Work

Key Findings

- **Enhanced Naïve Bayes models** improved spam detection performance.
- **TF-IDF weighting** reduced noise from common words, increasing accuracy.
- **Word2Vec embeddings** captured semantic meaning, boosting performance.
- **Dependency graphs** modeled word relationships, refining predictions.
- **Feature selection** yielded the best results (**86.3% accuracy, 83.1% F1-score, 0.92 AUC-ROC**).

Future Directions

- **Evaluate on larger, multilingual, and informal text datasets** (e.g., social media).
- **Test deep learning models** (e.g., **LSTMs, Transformers**) for **long-range dependencies**.
- **Refine dependency graphs** using **Bayesian structure learning**.
- **Explore hybrid models** (e.g., **SVM + embeddings, BERT**).
- **Assess computational efficiency** of advanced models for real-time spam detection.

Thanks!

Questions?