SEIS 763: Final Project Proposal

Daniel Walczak
Department of Electrical and Computer Engineering
University of St. Thomas
Saint Paul, MN, USA
walc3290@stthomas.edu

Ian Obert
Department of Electrical and Computer Engineering
University of St. Thomas
Saint Paul, MN, USA
ober7804@stthomas.edu

Dilip Singh Thakur
Department of Electrical and Computer Engineering
University of St. Thomas
Saint Paul, MN, USA
thak6503@stthomas.edu

Mark Anson
Department of Electrical and Computer Engineering
University of St. Thomas
Saint Paul, MN, USA
anso9374@stthomas.edu

I. Introduction

The primary objective of this project is to develop a supervised machine learning model that accurately predicts whether an email is spam or not spam based on the frequency of specific words appearing in the message body. This problem is practically important because modern email providers rely heavily on automated filters to reduce unwanted or malicious emails without requiring manual review. From an application perspective, our goal is to compare several classification algorithms and determine which approach produces the most reliable predictions on a high-dimensional, word-frequency dataset.

II. Dataset Description and Data Collection

The dataset used for this project is the Email Spam Classification Dataset from Kaggle, which contains 5,172 emails. Each sample is represented by 3,000 continuous features, where each feature captures the frequency of a specific word within that email. The target variable is a binary label, indicating whether the email is spam (1) or not spam (0). Because all predictors are already numeric and there are no missing values, the dataset is well-suited for a variety of machine learning algorithms and allows us to focus on model design, feature selection, and dimensionality reduction. In addition, the features are highly sparse and often skewed, which motivates the use of techniques such as feature selection and dimensionality reduction to improve computational efficiency and model stability.

III. Data Preparation

A. Train–Test Split

To evaluate model generalization, we split the data into training and testing sets using an 80/20 train–test split. All preprocessing steps were fit only on the training data and then applied to the test data, which prevents information leakage and provides an unbiased assessment of performance.

B. Handling Class Imbalance

The target distribution in the dataset is moderately imbalanced, with approximately 71% non-spam emails (3,672) and 29% spam emails (1,500). Although this imbalance is not extreme, a classifier trained directly on this distribution may still become biased toward predicting the majority class (non-spam), which could hurt recall for spam. To mitigate this, we applied SMOTE (Synthetic Minority Over-sampling Technique) to the training set only. SMOTE generates synthetic spam examples by mixing existing minority-class samples in feature space, increasing the representation of spam emails without simply duplicating existing rows. This helps the models pay more attention to the minority (spam) class and improve sensitivity and recall for spam samples.

C. Scaling

Given the large number of continuous predictors and the fact that many of our algorithms are sensitive to differences in scale, we standardized all features using StandardScaler. This scaling prevents features with large raw ranges from dominating distance-based methods, improves numerical stability for optimization algorithms, and ensures PCA and other variance-based methods are not overly influenced by high-variance features.

D. Feature Elimination and Feature Engineering

The original dataset contained an ID column that served only as a unique identifier for each email and carried no predictive information. This feature was removed prior to modeling. No additional manual feature elimination or complex feature engineering was performed. We deliberately started with all available features to respect the assignment requirement and allow backward elimination and forward selection to identify individually important predictors and dimensionality reduction methods to discover lower-dimensional representations that better capture structure in the data.

IV. Feature Scaling

To explore whether a smaller subset of features could improve or simplify model performance, we applied both backward elimination and forward selection on the word-frequency features. Backward Elimination starts with all features and iteratively removes the least significant predictor at each step until further removal would degrade performance. Forward Selection starts with no features and iteratively adds the most

informative predictor at each step, stopping when adding additional features no longer yields meaningful improvement. Because the dataset contains 3,000 predictors, these stepwise procedures are computationally expensive. In our workflow they were used primarily to Identify which word-frequency features appear most consistently useful across models and diagnose redundancy among predictors. In practice, the extreme sparsity and high dimensionality of the dataset made global dimensionality reduction, via PCA and related techniques, more impactful for model efficiency than strictly relying on manual, stepwise feature selection.

V. Dimensionality Reduction

With more than 3,000 word-frequency features, reducing dimensionality was an important step to explore as part of our pre-processing pipeline. Many of the original features are sparse or highly correlated, so applying dimensionality-reduction techniques had the potential to simplify the learning task, limit noise, and improve model stability. Although our later results show that models ultimately performed best on the full feature set, evaluating several dimensionality-reduction methods allowed us to understand how much of the predictive signal could be preserved in a compressed representation. PCA was applied to the standardized training data to transform the high-dimensional input into a smaller set of uncorrelated components. The method identifies directions in the data that contain the most variation and projects each observation into this new feature space. PCA allowed us to compress thousands of original features into a much smaller representation, remove components associated with very low variance, and improve generalization by working with orthogonal features. We retained enough components to preserve 99.9% of the variance, which resulted in 196 principal components and provided a compact representation of the dataset without losing meaningful information.

To examine whether nonlinear patterns in the data could enhance classification, we also tested Kernel PCA using an RBF kernel. Unlike standard PCA, Kernel PCA can capture more complex, nonlinear relationships by mapping the data into a higher-dimensional space before performing PCA. Because Kernel PCA does not report explained-variance ratios, it requires the number of components to be specified manually. For consistency, we set this value equal to the 196 components selected by standard PCA, allowing us to directly compare linear and nonlinear feature extraction methods while holding dimensionality constant. This helped determine whether the additional complexity of a nonlinear method offered any practical benefit for our dataset. We also explored Linear Discriminant Analysis as a supervised dimensionality-reduction technique. LDA differs from PCA-based methods by incorporating class labels and seeking the projection that maximizes separation between spam and non-spam emails. Since LDA can produce at most one fewer component than the number of classes, our binary classification task resulted in a single discriminant component. This feature represents the direction that emphasizes the boundary between the two classes. Including LDA provided a useful contrast to the unsupervised, variance-focused methods and helped highlight the trade-off between preserving variance and explicitly maximizing class separation in reduced-dimensional space. To evaluate the effect of dimensionality reduction on model performance, we trained each classifier using four versions of the dataset: the full feature set, PCA-transformed features, Kernel PCA–transformed features, and the one-dimensional LDA representation. The accuracy results are summarized in Figure 3. Across nearly all models, the highest accuracy was achieved when training directly on the full 3,000-word feature space. Logistic Regression, Random Forest, XGBoost, and Naive Bayes all performed best with no dimensionality reduction, suggesting that meaningful predictive signal is distributed across many of the original features and is lost when compressing the data into a lower-dimensional space. Among the dimensionality-reduction approaches, Kernel PCA generally provided stronger accuracy than standard PCA or LDA, especially for tree-based models and the polynomial SVM. However, even in these cases, Kernel PCA did not surpass the performance of using the full feature set. LDA, which reduces the dataset to a single discriminant component, produced consistent but lower accuracy across all models due to its extreme compression. Overall, the results show that dimensionality reduction did not improve predictive accuracy for this dataset, and retaining the full set of word-frequency features enabled the models to capture more of the subtle patterns associated with spam detection. While PCA, Kernel PCA, and LDA provided useful insight into the structure of the feature space, the visualizations and accuracy comparisons make it clear that the full-dimensional representation remains the most effective input for classification.

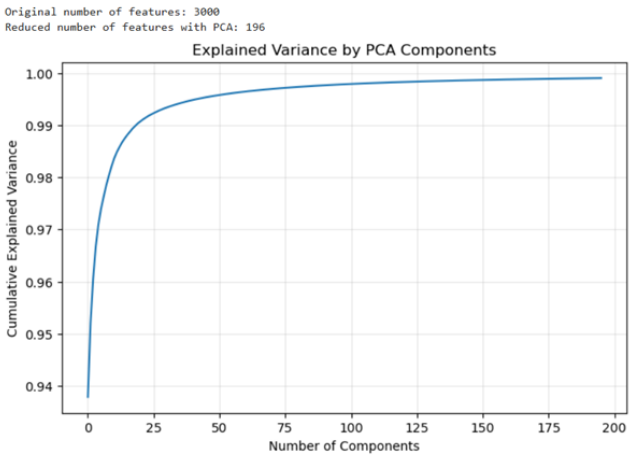

Fig. 1. Variance of PCA Components

A. Models Attempted and Methodology

This section describes the machine learning models implemented for email classification, focusing on Random Forest, Decision Tree, XGBoost, Naïve Bayes, and three variations

Method	No Reduction	PCA	Kernel PCA	LDA
Model				
Logistic Regression	0.973582	0.957474	0.868557	0.807990
Random Forest	0.975515	0.908505	0.872423	0.804768
XGBoost	0.969716	0.937500	0.886598	0.805412
Decision Tree	0.942010	0.784794	0.797036	0.804768
Naive Bayes	0.949742	0.423969	0.713273	0.804768
SVM (RBF)	0.866624	0.869845	0.862113	0.783505
SVM (Poly)	0.764820	0.750000	0.864046	0.806057

Fig. 2. Variance of PCA Components

of Support Vector Machines (SVM): RBF kernel, Polynomial kernel, and Linear kernel.

B. Random Forest

Random Forest is an ensemble learning method that constructs multiple decision trees using randomly selected subsets of data and features. The final prediction is determined by majority voting across the individual trees. This approach reduces overfitting and handles high-dimensional, noisy text data effectively. Random Forest demonstrated robust performance with high precision and recall for both classes, ensuring reliable classification of spam and non-spam emails.

TABLE I
RANDOM FOREST PERFORMANCE METRICS

Metric	Class 0	Class 1
Precision	0.98	0.96
Recall	0.98	0.95
F1-Score	0.98	0.96
Accuracy: 0.97		

C. Decision Tree

Decision Trees are greedy algorithms designed to split nodes based on which feature best reduces impurity within the node generating information gain. Our Decision Tree model utilized GridSearch Cross Validation, which indicated optimal parameters of: Max Depth of 10, Min Samples Leaf of 5, and Min Samples Leaf of 12. Model performance was stronger for the negative class, which may be preferred if it is more important to minimize false positives. However, overall model performance was still below several other models tested.

D. XGBoost

XGBoost is an ensemble, gradient boosting method that sequentially builds decision trees each correcting from the errors of the previous models. Our XGBoost model obtained its highest performance utilizing RandomSearch Cross Validation adjusting the hyperparameter Max Depth. XGBoost obtained impressive results with an 0.98 F1-Score for the negative class, 0.94 F1-Score for the positive, and 0.97 Accuracy.

TABLE II
DECISION TREE PERFORMANCE METRICS

Metric	Class 0	Class 1
Precision	0.94	0.89
Recall	0.95	0.87
F1-Score	0.95	0.88
Accuracy: 0.93		

TABLE III
XGBOOST PERFORMANCE METRICS

Metric	Class 0	Class 1
Precision	0.97	0.96
Recall	0.98	0.92
F1-Score	0.98	0.94
Accuracy: 0.97		

E. Naïve Bayes

Naïve Bayes applies Bayes' theorem under the assumption that features are conditionally independent. Despite this simplification, Naïve Bayes is effective for text classification due to the high occurrence of indicative keywords in spam emails. The model exhibited strong recall for the spam class, effectively identifying most spam emails in the dataset.

TABLE IV
NAÏVE BAYES PERFORMANCE METRICS

Metric	Class 0	Class 1
Precision	0.98	0.88
Recall	0.95	0.96
F1-Score	0.96	0.92
Accuracy: 0.95		

F. Support Vector Machines (SVM)

SVM aims to find an optimal hyperplane that maximizes the margin between classes. Three different kernels were evaluated to assess their effectiveness for email classification. The RBF kernel projects data into a higher-dimensional space to model non-linear relationships.

1) SVM with RBF Kernel: This configuration favored the non-spam class and exhibited poor recall for the spam class.

2) SVM with Polynomial Kernel: The polynomial kernel incorporates polynomial interactions to capture more complex decision boundaries. The model displayed a strong bias toward non-spam predictions, resulting in low performance for spam detection.

TABLE V
SVM (RBF KERNEL) PERFORMANCE METRICS

Metric	Class 0	Class 1
Precision	0.79	0.92
Recall	0.99	0.36
F1-Score	0.88	0.52
Accuracy: 0.80		

TABLE VI
SVM (POLYNOMIAL KERNEL) PERFORMANCE METRICS

Metric	Class 0	Class 1
Precision	0.74	0.91
Recall	0.99	0.16
F1-Score	0.85	0.27
Accuracy: 0.75		

3) SVM with Linear Kernel: The linear kernel is particularly effective for high-dimensional text data, such as TF-IDF features. This model performed exceptionally well, achieving metrics comparable to Random Forest.

TABLE VII
SVM (LINEAR KERNEL) PERFORMANCE METRICS

Metric	Class 0	Class 1
Precision	0.97	0.92
Recall	0.97	0.93
F1-Score	0.97	0.93
Accuracy: 0.96		

TABLE VIII
BACKWARD ELIMINATION ($p = 0.05$) PERFORMANCE SUMMARY

Metric	Value
Iterations	1,655
Features Retained	1,346
Accuracy	0.82

G. Backward Elimination

Backward elimination was evaluated to determine whether reducing the number of word-frequency features could improve the performance of email classification. In this process, features are iteratively removed based on their statistical significance, and the reduced feature set is then fit to the logistic regression model.

The hyperparameters used for the elimination threshold were set to $p = 0.05$ and $p = 0.10$. These values were chosen because $p = 0.05$ aligned with the class assignment example, while $p = 0.10$ served to illustrate the effect of a less strict parameter.

1) Elimination with $p = 0.05$: Using a significance threshold of $p = 0.05$, the backward elimination procedure ran for a total of 1,655 iterations and retained 1,346 features. The resulting logistic regression model achieved an accuracy of 0.82.

2) Elimination with $p = 0.10$: When the threshold was set to $p = 0.10$, the elimination process completed 1,255 iterations and retained 1,746 features. In this configuration, model accuracy decreased slightly from 0.82 to 0.80.

TABLE IX
BACKWARD ELIMINATION ($p = 0.10$) PERFORMANCE SUMMARY

Metric	Value
Iterations	1,255
Features Retained	1,746
Accuracy	0.80

3) Runtime Comparison: A notable aspect of backward elimination is the computational cost. The stricter threshold of $p = 0.05$ required 200 minutes to complete, while the relaxed threshold of $p = 0.10$ reduced runtime to 175 minutes. This indicates a smaller-than-expected reduction in runtime despite eliminating fewer features.

VI. MODEL DEPLOYMENT

The final model was selected based on both performance and interpretability, particularly in the context of our dimensionality-reduction experiments. As shown in Section 2, techniques such as PCA, Kernel PCA, and LDA were extensively evaluated to determine whether compressing more than 3,000 word-frequency features could improve classification efficiency or accuracy. However, our results consistently demonstrated that reducing dimensionality led to a loss of predictive consistency, for each attempted model (Random Forest, Naïve Bayes, and the SVM variants) achieved their highest accuracy when trained on all features. Because meaningful information was distributed across many sparse textual features, dimensionality reduction removed subtle but important patterns relevant to spam detection. For this reason, we selected Logistic Regression with standardized full-dimensional inputs as the final deployed model: it preserved the complete feature representation, avoided the performance degradation introduced by feature compression, and maintained strong predictive reliability without the need for oversampling or additional preprocessing complexity with a final output accuracy of 96% from a test set of 1,035 emails.

A. Confusion Matrix

The final deployed model combined a StandardScaler with a Logistic Regression classifier. Notably, this configuration performed effectively without requiring any oversampling techniques such as SMOTE, indicating that the learned decision boundary generalized well to the imbalanced class distribution. The dataset was split using an 80/20 train–test split.

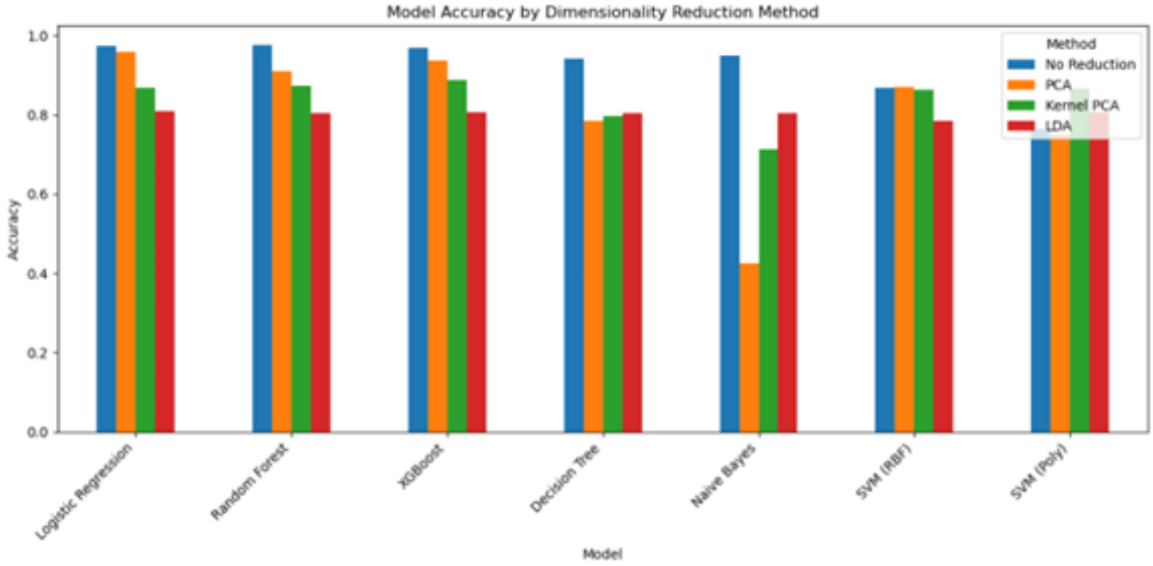

Fig. 3. Model Scores by Method

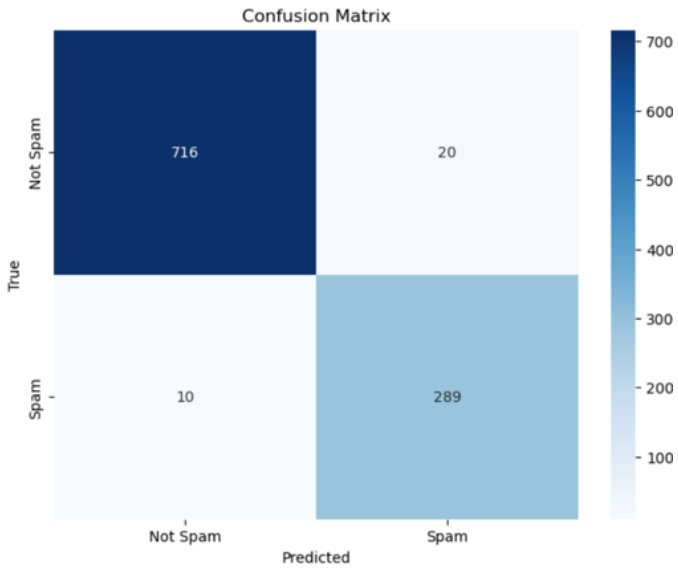

Fig. 4. Confusion Matrix for Logistic Regression

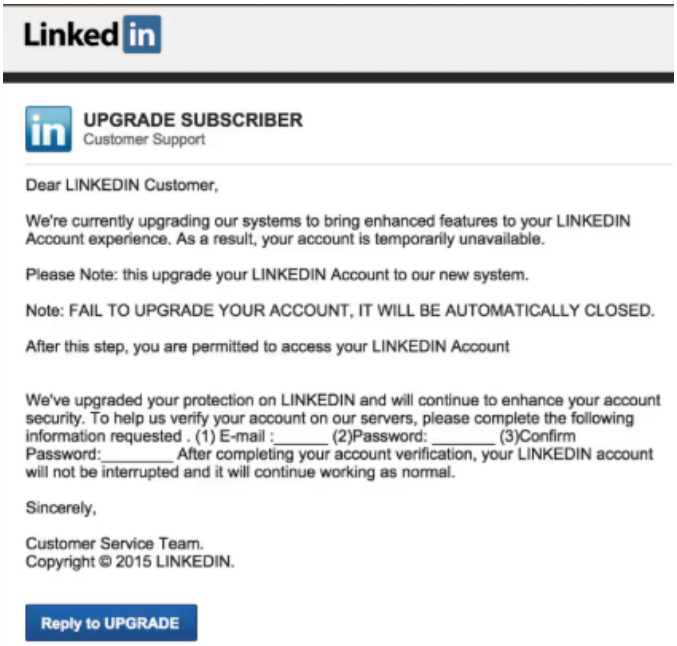

Fig. 5. SPAM LinkedIn Email

From a test set of 1,035 samples, the model produced the following outcomes:

- 716 True Positives (correctly identified non-spam emails)
- 289 True Negatives (correctly identified spam emails)
- 20 False Positives (spam misclassified as non-spam)
- 10 False Negatives (non-spam misclassified as spam)

This demonstrates that Logistic Regression, when paired with appropriate feature scaling, provides a robust and interpretable approach for the email classification task.

Additionally, the model achieved a Mean Squared Error (MSE) of 0.03, further indicating strong predictive performance with minimal deviation between predicted and true labels.

VII. Model Demonstration on Emails

To truly demonstrate how the model works, a program was written that could input text from an email and a probability (or confidence) rating was output and classified if the email was Spam or Not Spam with a range of 0-100% confidence.

A. Example Email: LinkedIn

Given here was an example email that was attempted. This email which claims to be from the hiring job site LinkedIn, claims that a user must upgrade their account or the account will be shut down. The email then goes on to request the user

Password and Email. Upon entering this text into the spam detection program, the model was able to detect the email as Spam with a confidence rating of 98.16%

B. Example Email: University of St. Thomas Sports

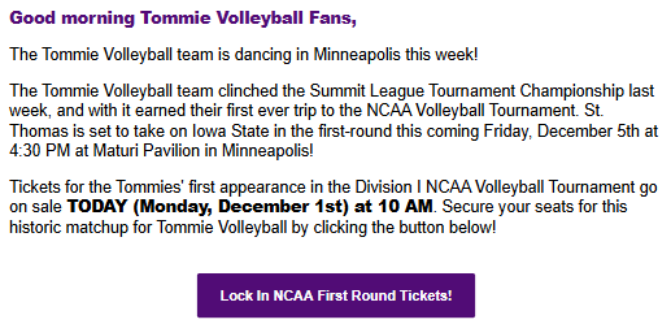

Fig. 6. UST Volleyball Email

The next email that will be discussed is an email that was sent out to students and faculty at the University of St. Thomas. This email is regarding the recent win for the university's women's volleyball team. In this email, there is a prompt to buy new tickets for the upcoming tournament. This email is viewed as promotional for the school's sports teams, and the model developed predicted this email as "Not Spam" with a confidence rating of 99.98%.

VIII. Conclusion

This project was able to develop an operational email detection with high accuracy. By attempting the various models from sklearn's library, it was determined that the Logistic Regression model fitting proved to be the most consistent and produced the best results. Furthermore, it was also found that dimensionality reduction was removed because this lowered the accuracy of each tested model. One theory of this happening was that PCA does not understand semantic structure and only eliminates information based on variance. Future work on this project would include focusing on whether a dimensionality reduction approach could be applied with more reference to semantic meaning or sentence structure.

References

[1] A. McCallum and K. Nigam, "A comparison of event models for naive Bayes text classification," in *Proc. AAAI Workshop on Learning for Text Categorization*, 1998. [Online]. Available: https://www.cs.cmu.edu/~mccallum/papers/eventmodels.pdf

[2] I. T. Jolliffe and J. Cadima, "Principal component analysis: A review and recent developments," *Philosophical Transactions of the Royal Society A*, vol. 374, no. 2065, 2016. [Online]. Available: https://royalsocietypublishing.org/doi/10.1098/rsta.2015.0202

[3] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001. [Online]. Available: https://link.springer.com/article/10.1023/A:1010933404324

[4] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An Introduction to Statistical Learning*, 2nd ed. New York, NY, USA: Springer, 2021, ch. 6. [Online]. Available: https://www.statlearning.com

[5] D. W. Hosmer, S. Lemeshow, and R. X. Sturdivant, *Applied Logistic Regression*, 3rd ed. Hoboken, NJ, USA: Wiley, 2013. [Online]. Available: https://onlinelibrary.wiley.com/doi/book/10.1002/9781118548387