# Data Mining Assignment 2
# Classification for the Detection of Opinion Spam

**Lizzy Brans**
1110209

**Willemijn Barens**
2145480

**Anna de Wolff**
1498835

## 1 Introduction

This report examines the challenge of opinion spam detection, with a particular emphasis on differentiating between deceptive and truthful reviews within the context of online hotel reviews. Opinion spam, characterized by the presence of fake reviews designed to deceive readers, has become an issue with the rise of online platforms like TripAdvisor and Yelp, which have been increasingly used by businesses to influence customer perception. In recent years, the manipulation of reviews has created a significant challenge for consumers and platforms alike, as companies often attempt to strengthen their own reputation or tarnish competitors' standings through false testimonials. This study specifically targets the classification of negative reviews, as these reviews often carry notable weight in consumer decision-making and have been shown to impact brand reputations considerably when left unchecked [8; 9]. Our analysis uses several machine learning classification models, each executing the same task. For this, a well-structured and widely used dataset for spam detection research allows for focused experimentation on model performance and feature selection. Previous research has underscored the efficacy of using both generative and discriminative classifiers to detect opinion spam. Building upon this foundation, our approach explores both linear and non-linear classification techniques to show which model and feature set best capture the patterns associated with deceptive reviews. Additionally, the study employs unigram and bigram textual features to examine the role of word context in improving classification accuracy, a method that has shown promise in previous opinion spam detection studies [9].

## 2 Data

The dataset used for this evaluation is a corpus of 800 hotel reviews collected and annotated by Ott et al. [8; 9], known for its balanced composition across truthful and deceptive categories. Specifically, the dataset includes four distinct classes: positive truthful, positive deceptive, negative truthful, and negative deceptive reviews. Each category contains 200 reviews, ensuring an equal distribution of data points, which supports unbiased model training and validation. For the purpose of this study, we have narrowed our focus to 400 negative reviews, comprising 200 truthful and 200 deceptive reviews. Negative reviews were chosen to target cases where deceptive content could be potentially harmful to competitors, aligning with recent work by Ott et al. [9], which indicates that negative opinion spam can be more difficult to detect due to its similarity to genuine critiques. The truthful reviews in the dataset were sourced from well-known online review platforms, such as Expedia, Hotels.com, Orbitz, Priceline, TripAdvisor, and Yelp, reflecting authentic consumer feedback from these trusted sites. Conversely, the deceptive reviews were created via Mechanical Turk, where contributors were instructed to fabricate realistic, negative hotel reviews, imitating the language and structure of genuine reviews. This approach ensures a gold-standard corpus of deceptive content by simulating spam behaviour under controlled conditions, thereby enabling strict testing of machine learning models on genuinely deceptive content.

### 2.1 Training and Testing

The dataset is structured into five folds, an arrangement that facilitates the training and testing of classification models. Folds 1 through 4, comprising 640 reviews, are used for training and hyperparameter tuning. Fold 5, containing 160 reviews, is reserved for testing and evaluating the models. For the training phase, cross-validation is utilised to optimise hyperparameters and improve generalisation capabilities. Cross-validation enables the model to learn patterns across different subsets of data, thereby reducing overfitting and providing a more reliable estimate of model performance on new data. Each model's hyperparameter tuning is essential for achieving optimal accuracy, precision, and recall in detecting deceptive opinion spam.

### 2.2 Preprocessing

Preprocessing of textual data is critical for maximising the performance of machine learning models in natural language processing tasks. In preparing the dataset for analysis, we applied several preprocessing techniques to enhance the quality of the features used in classification. These steps reducing noise in the data and ensure that only relevant linguistic information is retained:

*Tokenization.* Each review was tokenized at the word level, splitting the text into individual words. Tokenization is a foundational step in natural language processing that facilitates the subsequent analysis of text by breaking down sentences into their component words, which can then be used as features for classification.

*Stopword Removal.* Common English stopwords, such as "the," "and," and "is," were removed from the reviews. Removing these frequently occurring but semantically trivial words enhances the signal-to-noise ratio in the dataset, as stopwords are unlikely to contribute meaningfully to the distinction between truthful and deceptive reviews [8].

*Lemmatisation.* Lemmatization was applied to reduce words to their base or root forms, ensuring consistency in word representation. This step is particularly important for reducing redundancy in the feature set, as it unifies different inflected forms of the same word (e.g., "running" and "ran" become "run"). By focusing on base forms, lemmatization improves the model's ability to generalize across variations in word usage. Compared to a similar approach such as stemming, lemmatization is a more sophisticated approach and preserves meaning better.

*TF-IDF Vectorisation.* The reviews were transformed into numerical vectors using Term Frequency-Inverse Document Frequency (TF-IDF). TF-IDF assigns a weight to each word based on its frequency in a review and its rarity across the entire dataset, thus capturing the relative importance of each term within the context of opinion spam detection. This avoids a common problem in text analysis where the most frequent words in one review are usually also the most frequent words in other reviews [5]. Both unigram and bigram features were extracted during TF-IDF vectorization. Unigrams represent single words, while bigrams capture pairs of consecutive words, allowing for the analysis of word pairings that may indicate deceptive language patterns, as suggested in Ott et al.'s findings on bigram effectiveness in opinion spam detection [9] This was used for all models.

These steps collectively contribute to a more clean, interpretable and structured input for our models. Irrelevant and redundant information is reduced, while preserving essential linguistic characteristics that point to deceptive and truthful content.

## 3 Modeling and analysis

The employed models can be categorized into linear classifiers: Multinomial Naive Bayes (Generative) and Logistic Regression with Lasso Penalty (Discriminative). On the other hand, we have nonlinear classifiers: Classification Trees and Random Forests (Ensemble of classifiers).

*Multinomial Naive Bayes.* The multinomial Naive Bayes model calculates the probability of observing each word in the document for each class [3]. The name naive stems from the assumption of independence among features, meaning the presence of one word does not affect the presence of another. Naive Bayes is computationally efficient and handles high-dimensional data like text effectively. In text classification especially, this is a serious oversiplification. However, Naive Bayes still performs surprisingly well despite its assumptions, making it a strong baseline model in text classification tasks [3]. Additionally, feature selection methods like TF-IDF helps to prevent overfitting in this model, as it's able to capture the relevant features of the data [7]. A last consideration of Naive Bayes is the *zero probability problem*; When a feature appears in the test data, but not in the training data, it can result in zero probability estimates. For this reason, we made sure to apply Laplace smoothing, mitigating this problem [2].

*Logistic Regression with Lasso Penalty.* Logistic regression is a discriminative model that, similarly to naive Bayes, predicts the probability of a binary class outcome based on input features. The addition of Lasso (L1) regularisation introduces a penalty term that not only prevents overfitting but also aids in feature selection by shrinking the coefficients of less relevant features to zero, thereby simplifying the model [11]. While Lasso aids in improving model accuracy, it may exclude potentially useful features if the penalty is set too high, which necessitates tuning of such hyperparameters [13]. Another drawback is its sensitivity to outliers, where the model can disproportionately influence the results, leading to inaccurate predictions [11]. TF-IDF can help mitigate this drawback by balancing the influence of rare words.

*Classification tree.* The classification tree algorithm partitions data by creating a series of binary splits based on feature values [12]. In text classification, this may be the presence, absence, or frequency of certain words. Classification trees are interpretable, as the resulting splits reveal the decision-making process of the model. Unlike the two aforementioned models, classification trees can capture non-linear relationships between features, which can be beneficial for more complex relations [11]. On the other hand, classification trees are prone to overfitting, especially when trees are deep and complex. In the current application, we included constraints on maximal tree depth, minimum amount of cases required for a split, minimum number of samples that a leaf node must contain and a cost-complexity parameter alpha. These constraints aim to mitigate the risk of overfitting by promoting simpler trees.

*Random forests.* Random forests is an ensemble learning method that combines multiple decision trees to form a "forest," resulting in improved prediction accuracy and robustness compared to a single decision tree [10]. By aggregating predictions from numerous trees, random forests reduce overfitting and are less susceptible to noise in the data, making them a strong choice for classification tasks like opinion spam detection. Another advantage is that random forests can create diverse trees and reduce correlation between them by randomly selecting a subset of features when splitting nodes in the trees [11]. However, these advances trade with computational cost, as more trees are grown. Furthermore, random forests may lose some interpretability compared to individual decision trees, considering the aggregated decision-making process across numerous trees. The hyperparameters of the random forests were maximum depth of the tree, one for the number of trees included in the ensemble, minimal samples required for a plit, minimal samples allowed in a leaf, maximum amount of features that were used and a cost-complexity function.

Lastly, each model was run twice, once using unigram and once including both uni-and bigram features, resulting in a comparison of eight different models. In the unigram models, only single words or tokens are used as features. Likewise, the bigram models use both unigrams and pairs of consecutive words as features. By comparing unigram and bigram representations, we aim to determine the influence of word context on model performance.

### 3.1 Model Performance Metrics

For each model, 10-fold cross-validation was performed on the training set (folds 1-4, 640 reviews) to select the best hyperparameters. We did so using grid search, an optimization technique that exhaustively tries every combination of pre-specified hyper-parameter values to find the best model [6]. The final performance of the models was evaluated using fold 5 (160 reviews). Next, performance was assessed using several metrics. First, accuracy was used to capture the overall correctness of the model, which is the amount of correct predictions out of all predictions. Next, precision determines the number of correct positive predictions out of all predicted positives. Recall estimates the correct positive predictions out of all actual positives. Lastly, the F1 score captures the balance between precision and recall.

### 3.2 Model comparison

To properly compare models and determine whether they differ significantly in accuracy, we conducted a Cochran's Q test. The Cochran's Q test is a non-parametric test that evaluates whether the success rates across classifiers is statistically different. Cochran's Q test has two assumptions: samples should be related, in this case the same observations are tested by different models. Next, the outcomes should be binary which in our case are the correct/incorrect predictions [4]. Therefore, this test seems like a good fit to our goal. A significant Q statistic indicates there is a significant difference between the models. If this is indeed the case, we then use a pairwise McNemar's tests for post-hoc analysis. The McNemar reveals whether the performance of two models differ in a meaningful way [1]. Similarly to Cochran's Q test, it is designed for binary outcomes and therefore applicable to the current problem [1].

## 4 Results

### 4.1 Multinomial Naive Bayes

*Unigrams.* The Naive Bayes classifier with unigrams yielded an overall accuracy of 82.50%. With a precision of 77% for the deceptive class and 92% for the truthful class. However, the recall for the deceptive class was notably high at 94%, whereas the recall for the truthful class was lower at 71%. The F1-Score was 0.84 for the deceptive class and 0.80 for the truthful class.

| | **Precision** | **Recall** | **F1-Score** |
|---|---|---|---|
| Deceptive | 0.77 | 0.94 | 0.84 |
| Truthful | 0.92 | 0.71 | 0.80 |
| Accuracy | 0.82 | | |

Table 1: Multinomial Naive Bayes metrics, unigrams

*Bigrams.* With bigrams as features, the multinomial naive bayes yielded an overall accuracy of 87.50%. The classifier achieved a precision of 88% for the deceptive class and 87% for the truthful class. The recall values were 86% for deceptive and 89% for truthful. The F1 scores was 0.87 for the deceptive class and 0.88 for the truthful class.

| | **Precision** | **Recall** | **F1-Score** |
|---|---|---|---|
| Deceptive | 0.88 | 0.86 | 0.87 |
| Truthful | 0.87 | 0.89 | 0.88 |
| Accuracy | 0.88 | | |

Table 2: Multinomial Naive Bayes metrics, bigrams

### 4.2 Logistic Regression with Lasso Penalty

*Unigrams.* The logistic regression model with unigrams resulted in an accuracy of 84%. It achieved a precision of 88% for the deceptive class, and 81% for the truthful class. The recall values were 79% for the deceptive class and higher for the truthful class at 89%. The F1 score was 0.83 and 0.85 for the deceptive and truthful class respectively, striking a very good balance between recall and precision.

| | **Precision** | **Recall** | **F1-Score** |
|---|---|---|---|
| Deceptive | 0.88 | 0.79 | 0.83 |
| Truthful | 0.81 | 0.89 | 0.85 |
| Accuracy | 0.84 | | |

Table 3: Logistic regression metrics, unigrams

*Bigrams.* when including bigrams, the logistic regression yields an accuracy of 85%. The precision for the deceptive class was 86%, and for the truthful class this was 84%. For recall, the deceptive class received 84% and for the truthful class this was slightly higher at 86%. The F1-score for both classes was 0.85.

| | **Precision** | **Recall** | **F1-Score** |
|---|---|---|---|
| Deceptive | 0.86 | 0.84 | 0.85 |
| Truthful | 0.84 | 0.86 | 0.85 |
| Accuracy | 0.85 | | |

Table 4: Logistic regression metrics, bigrams

### 4.3 Classification Trees

*Unigrams.* The classification tree with unigram features scored a relatively low but acceptable accuracy of 66.87%. Its precision was 68% for the deceptive and 66% for the truthful class. It had a recall score of 62% for the deceptive and 71% for the truthful class. The F1-score was 0.65 for the deceptive, and 0.68 for the truthful classes. These values reflect a moderate balance between precision and recall.

| | **Precision** | **Recall** | **F1-Score** |
|---|---|---|---|
| Deceptive | 0.68 | 0.62 | 0.65 |
| Truthful | 0.66 | 0.71 | 0.68 |
| Accuracy | 0.66 | | |

Table 5: Classification tree metrics, unigrams

*Bigrams.* Including bigram features, the classification tree algorithm scored an accuracy of 68.75%. With a precision in the deceptive class of 70% and 67% of the truthful class. Recall was relatively low for the deceptive class 65% compared to the truthful class 72%. The F1-score for the deceptive class was 0.68, while for the truthful class this was 0.70.

### 4.4 Random Forests

*Unigrams.* For unigrams, the random forest obtained an accuracy of 85%. The precision was 88% for the deceptive class, and 83% for the truthful class. Its recall was 81% and 89% for the truthful class. Its F1-score was 0.84 and 0.86 for the de-

| | **Precision** | **Recall** | **F1-Score** |
|---|---|---|---|
| Deceptive | 0.70 | 0.65 | 0.68 |
| Truthful | 0.67 | 0.72 | 0.70 |
| Accuracy | 0.69 | | |

Table 6: Classification tree metrics, bigrams

ceptive and truthful class respectively, indicating a good balance between precision and recall.

| | **Precision** | **Recall** | **F1-Score** |
|---|---|---|---|
| Deceptive | 0.88 | 0.81 | 0.84 |
| Truthful | 0.83 | 0.89 | 0.86 |
| Accuracy | 0.85 | | |

Table 7: Random forests metrics, unigrams

*Bigrams.* When employing the models with bigrams instead, the accuracy got to 83.12%. The model yielded a precision score of 92% for the deceptive class and slightly lower precision 77% for the truthful class. With a recall of 72% and 94% for both classes respectively, we observe an improvement in precision in the truthful class compared to unigrams. The F1-score was 0.81 for the deceptive class and 0.85 for the truthful class, showing a well balanced prediction.

| | **Precision** | **Recall** | **F1-Score** |
|---|---|---|---|
| Deceptive | 0.92 | 0.72 | 0.81 |
| Truthful | 0.77 | 0.94 | 0.85 |
| Accuracy | 0.83 | | |

Table 8: Random forest metrics, bigrams

For a comprehensive summary of the precision, recall, F1 and accuracy tables across models, see Appendix B.

### 4.5 Statistical tests

The Chochran's Q test yielded a Q statistic of 59.31, with a corresponding P-value of $2.08 \times 10^{-10}$. This indicates that there is a significant difference between the models.

The follow-up McNemar test yielded several significant results, predominantly showing that the classification tree with both uni- and bigram features is statistically different from all other models (except in comparison with itself). All significant results can be found found in table 9. For the all results, see Appendix A.

| Comparison | P-value |
|---|---|
| NB_Uni vs CT_Uni | 0.0012 |
| NB_Uni vs CT_Bi | 0.0092 |
| NB_Bi vs CT_Uni | 5.55e-06 |
| NB_Bi vs CT_Bi | 7.33e-05 |
| LogReg_Uni vs CT_Uni | 9.85e-05 |
| LogReg_Uni vs CT_Bi | 0.0005 |
| LogReg_Bi vs CT_Uni | 8.17e-05 |
| LogReg_Bi vs CT_Bi | 0.0004 |
| CT_Uni vs RF_Uni | 1.54e-05 |
| CT_Uni vs RF_Bi | 0.0002 |
| CT_Bi vs RF_Uni | 0.0002 |
| CT_Bi vs RF_Bi | 0.001 |

Table 9: Significant comparisons between models(p < 0.05)

### 4.6 Feature importance

In our analysis, we identified the top five most significant terms (features) associated with both fake and genuine reviews across different models and feature types. These important terms were extracted based on the feature importance scores generated by each classification model. We examined unigram and bigram features separately, utilising Naive Bayes, Logistic Regression, Decision Tree, and Random Forest classifiers. These features reveal patterns in the language used in fake versus genuine reviews, providing insight into each model's linguistic markers for classification. For a detailed breakdown of feature importance score across the models, please refer to Appendix C.

#### 4.6.1 Terms indicating fake reviews

The most important features that suggest a review might be fake were identified through the various models, with terms consistently pointing towards fabricated reviews. Based on the feature importance scores, the top five terms indicating fake reviews are "*elevator*", "*adult*", "*turned*", "*priceline*", and "*star*". These terms appear prominently in deceptive content, likely due to their association with specific details or exaggerated experiences that deceptive reviewers attempt to replicate. For instance, "*elevator*" and "*priceline*" are terms that might be inserted to lend a sense of authenticity by referencing commonly reviewed aspects of hotel experiences (e.g., amenities or booking platforms). Similarly, words like "*star*" could reflect an attempt to mimic typical language associated with hotel ratings. At the same time, terms such as "*turned*" may be used in narrative constructions to simulate a personal storytelling style that readers associate with genuine reviews. These linguistic markers serve as distinguishing factors that models associate with deceptive reviews.

#### 4.6.2 Terms indicating genuine reviews

In contrast, terms indicating a review's genuineness are often tied to sensory descriptions, temporal markers, and location-specific language. The most significant truthful features identified across models are "*chicago*", "*smell*", "*luxury*", "*millennium*", and "*recently*". These terms consistently appear in genuine reviews, with "*chicago*" and "*millennium*" reflecting specific location or hotel names that are more naturally incorporated into authentic experiences. Sensory words such as "*smell*" and descriptive terms like "*luxury*" tend to signify genuine feedback as they are rooted in subjective experiences that deceptive reviewers might struggle to replicate convincingly. Temporal terms such as "*recently*" also add a sense of real-time progression or reflection, which may further distinguish truthful reviews from fabricated ones. The recurrence of these terms across different models reinforces the hypothesis that genuine reviews include specific, experiential language that is difficult for deceptive reviewers to mimic authentically.

#### 4.6.3 Important terms comparing linear and non-linear models

We also compare the feature importance results across linear and non-linear models, and differences in term selection and directional emphasis emerge. In the linear models (Naive Bayes and Logistic Regression), terms like "*smell*", "*luxury*", and "*millennium*" are identified as indicators of truthful reviews. These terms predominantly relate to specific hotel names, sensory descriptions, and experiential markers, reinforcing the linear models' focus on direct, content-related words commonly associated with authentic experiences. The features "*elevator*" and "*adult*" are consistently identified as indicators of deceptive reviews.

In contrast, the non-linear models (Decision Tree and Random Forest) provide a more nuanced perspective. For example, "*chicago*" appears as an important term for both truthful and deceptive reviews in non-linear models, reflecting a more complex interpretation of this location-specific term. Additionally, terms such as "*location*", "*smell*", and "*finally*" are highlighted in non-linear models as indicators of truthfulness, while "*turned*" is associated with deception. This suggests that non-linear models may capture more subtle contextual or positional cues within the reviews that are not as explicitly weighted in linear models.

While both linear and non-linear models identify similar core truthful indicators (e.g., "*chicago*" and "*smell*"), the non-linear models incorporate a broader range of terms across truthfulness and deception categories. This could indicate that non-linear models can capture layered linguistic patterns and more complex relationships in the data, which is in line with previous research [14].

This comparison highlights the different strengths of linear and non-linear approaches in opinion spam detection: linear models excel at identifying direct, content-driven markers, whereas non-linear models offer flexibility in capturing context-dependent nuances.

## 5 Discussion

In this section, we discuss the different models, the outcomes of the statistical comparisons and the impact of unigram versus bigram features.

Firstly, the accuracy of the generative linear model (Multinomial Naive Bayes) did not significantly differ from the discriminative linear model (Logistic Regression with Lasso Penalty), for neither unigrams nor bigrams. According to our results, the Logistic regression got an accuracy of 84% while Naive Bayes model yielded an accuracy of 83% (both unigrams). With bigrams, Naive Bayes outperformed Logistic regression. According to previous findings, logistic regression has a tendency to achieve higher accuracy in cases with non-independent features, in for example text classification [11]. Our results did not corroborate this. A lack of a significant difference may be due to the fact that our dataset was relatively limited and text data is known for its sparse features, and may not provide enough variance to highlight differences between the models. Additionally, regularisation of the Logistic regression may have constrained the parameters, essentially simplifying the model. Overall, the models performed compatibly well.

Next, there was no systematic performance improvement seen from non-linear classifiers. The random forest with unigrams did perfom marginally better, though this difference was insignificant (see Table 10). This slight advantage may be explained by the fact that random forests have the ability to generalize from complex, non-linear patterns in high-dimensional data while avoiding overfitting. This contrasts starkly to the classification tree, which consistently underperformed. The McNemar results confirm that the classification tree had a significant lower accuracy compared to all other models, for unigram and bigram models (see Appendix A). Decision Trees, despite their ability to also capture non-linear relationships, may have been fit to noise in the training data rather than the underlying distribution [15]. This notion underscores why the aggregation of multiple trees in the Random Forest model allowed for a more nuanced representation of feature importance. Overall, Naive Bayes and Random Forest were the best performing models across all metrics. Logistic regression performed moderately well, as the logistic regression with unigrams performed significantly better than random forests with unigrams.

The addition of bigram features generally enhanced performance for models that struggled with word context, such as Naive Bayes and Random Forests. Specifically, the Naive Bayes demonstrated an improved performance with bigram features, altough insignificant. Perhaps, the Naive Bayes model benefits from the additional contextual information provided by bigrams, enabling it to better distinguish between truthful and deceptive language patterns. Only for the random forests the addition of bigrams decreased accuracy. Perhaps, this happened Therefore, the use of bigrams varies depending on the classifier's inherent strengths in handling contextual information.

### 5.1 Limitations

Despite the promising results obtained from our analysis, certain limitations must be acknowledged. Firstly, the dataset used in this study consists of a balanced set of truthful and deceptive reviews, which may not reflect real-world data distributions. In practical applications, the prevalence of deceptive reviews is likely lower, and model performance might vary under imbalanced conditions. Additionally, while we explored unigram and bigram features, more complex feature representations, such as n-grams with n > 2 or word embeddings, could provide further insights and improve classification accuracy.

Next, the reliance on grid-search for tuning hyperparameters for non-linear models was suboptimal. While cross-validation was used to optimize these parameters, more sophisticated methods, such as Bayesian optimization, may have yielded improved results efficiently. Similarly, the computational complexity of Random Forests proved to be a bottleneck in scalability, particularly with added bigram features, as the model training time increased significantly.

Finally, the current study focused on conventional machine learning models without incorporating neural network-based approaches, which have shown potential in recent spam detection research. Future work could explore deep learning models such as Recurrent Neural Networks (RNNs) or Transformers, which are capable of capturing complex dependencies in text data but require larger datasets and computational resources.

## 6 Conclusion

This report reviewed the systematic detection of opinion spam in hotel reviews, focusing on distinguishing between deceptive and truthful reviews. A well-balanced dataset of 800 annotated hotel reviews was used to explore the performance of various classification models. These models were evaluated using both unigram and bigram features to study the effect of contextual information. Our findings show that both linear and non-linear classifiers can detect opinion spam effectively. The generative linear model (Multinomial Naive Bayes) did not significantly differ from the discriminative linear model (Logistic Regression with Lasso Penalty). There was no systematic performance improvement from non-linear classifiers compared to the linear classifiers; Random Forest with unigrams performed only marginally better than the linear models, though this difference was insignificant. Classification trees were structurally underperforming, likely due to overfitting. The addition of bigram features in almost all cases enhanced performance. This was especially the case for Naive Bayes and Random Forests. An analysis on feature importance showed truthful reviews were often related to location, sensory experiences, and specific hotel names, while deceptive reviews contained terms indicating detailed or specific amenities.

In conclusion, this study demonstrated the potential of machine learning classification models to detect opinion spam, where each model offers its own advantages. Future work could explore larger datasets that are more reflective of real-world distributions. Furthermore, more advanced models, such as neural networks and/or transformer architectures, could benefit from additional research and hold the potential to enhance text classification performance even further.

## References

[1] Jan De Leeuw, H Jia, L Yang, X Liu, K Schmidt, and AK Skidmore. Comparing accuracy assessments to infer superiority of image classification methods. *International Journal of Remote Sensing*, 27(1):223–232, 2006.

[2] David A Field. Laplacian smoothing and delaunay triangulations. *Communications in applied numerical methods*, 4(6):709–712, 1988.

[3] Ashraf M Kibriya, Eibe Frank, Bernhard Pfahringer, and Geoffrey Holmes. Multinomial naive bayes for text categorization revisited. In *AI 2004: Advances in Artificial Intelligence: 17th Australian Joint Conference on Artificial Intelligence, Cairns, Australia, December 4-6, 2004. Proceedings 17*, pages 488–499. Springer, 2005.

[4] Patrice Latinne, Olivier Debeir, and Christine Decaestecker. Different ways of weakening decision trees and their impact on classification accuracy of dt combination. In *Multiple Classifier Systems: First International Workshop, MCS 2000 Cagliari, Italy, June 21–23, 2000 Proceedings 1*, pages 200–209. Springer, 2000.

[5] Matthew Lavin. Analyzing documents with tf-idf. 2019.

[6] Petro Liashchynskyi and Pavlo Liashchynskyi. Grid search, random search, genetic algorithm: a big comparison for nas. *arXiv preprint arXiv:1912.06059*, 2019.

[7] Yonglian Luo and Cailin Lu. Tf-idf combined rank factor naive bayesian algorithm for intelligent language classification recommendation systems. *Systems and Soft Computing*, 6:200136, 2024.

[8] Myle Ott, Yejin Choi, Claire Cardie, and Jeffrey T Hancock. Finding deceptive opinion spam by any stretch of the imagination. *arXiv preprint arXiv:1107.4557*, 2011.

[9] Myle Ott, Claire Cardie, and Jeffrey T Hancock. Negative deceptive opinion spam. In *Proceedings of the 2013 conference of the north american chapter of the association for computational linguistics: human language technologies*, pages 497–501, 2013.

[10] Aakash Parmar, Rakesh Katariya, and Vatsal Patel. A review on random forest: An ensemble classifier. In *International conference on intelligent data communication technologies and internet of things (ICICI) 2018*, pages 758–763. Springer, 2019.

[11] Tomas Pranckevičius and Virginijus Marcinkevičius. Comparison of naive bayes, random forest, decision tree, support vector machines, and logistic regression classifiers for text reviews classification. *Baltic Journal of Modern Computing*, 5(2):221, 2017.

[12] Lior Rokach and Oded Maimon. Top-down induction of decision trees classifiers-a survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 35(4):476–487, 2005.

[13] Fariborz Salehi, Ehsan Abbasi, and Babak Hassibi. The impact of regularization on high-dimensional logistic regression. *Advances in Neural Information Processing Systems*, 32, 2019.

[14] Manpreet Singh. User-centered spam detection using linear and non-linear machine learning models. 2019.

[15] Yan-Yan Song and LU Ying. Decision tree methods: applications for classification and prediction. *Shanghai archives of psychiatry*, 27(2):130, 2015.

# A Results model comparison: McNemar's test

| Comparison | P-value | < 0.05 |
|---|---|---|
| NB_Uni vs NB_Bi | 0.13 | n |
| NB_Uni vs LogReg_Uni | 0.87 | n |
| NB_Uni vs LogReg_Bi | 0.60 | n |
| NB_Uni vs CT_Uni | 0.00 | y |
| NB_Uni vs CT_Bi | 0.00 | y |
| NB_Uni vs RF_Uni | 0.60 | n |
| NB_Uni vs RF_Bi | 1.00 | n |
| NB_Bi vs LogReg_Uni | 0.33 | n |
| NB_Bi vs LogReg_Bi | 0.52 | n |
| NB_Bi vs CT_Uni | 0.00 | y |
| NB_Bi vs CT_Bi | 0.00 | y |
| NB_Bi vs RF_Uni | 0.56 | n |
| NB_Bi vs RF_Bi | 0.14 | n |
| LogReg_Uni vs LogReg_Bi | 0.77 | n |
| LogReg_Uni vs CT_Uni | 0.00 | y |
| LogReg_Uni vs CT_Bi | 0.00 | y |
| LogReg_Uni vs RF_Uni | 0.84 | n |
| LogReg_Uni vs RF_Bi | 1.00 | n |
| LogReg_Bi vs CT_Uni | 0.00 | y |
| LogReg_Bi vs CT_Bi | 0.00 | y |
| LogReg_Bi vs RF_Uni | 1.00 | n |
| LogReg_Bi vs RF_Bi | 0.68 | n |
| CT_Uni vs CT_Bi | 0.77 | n |
| CT_Uni vs RF_Uni | 0.00 | y |
| CT_Uni vs RF_Bi | 0.00 | y |
| CT_Bi vs RF_Uni | 0.00 | y |
| CT_Bi vs RF_Bi | 0.00 | y |
| RF_Uni vs RF_Bi | 0.69 | n |

Table 10: Model Comparisons with Corresponding P-values

## B Performance metrics of all models

| Model | | Precision | Recall | F1-Score | Accuracy |
|---|---|---|---|---|---|
| Multinomial Naive Bayes: Unigrams | Deceptive | 0.92 | 0.72 | 0.81 | 0.83 |
| | Truthful | 0.77 | 0.94 | 0.85 | |
| Multinomial Naive Bayes: Bigrams | Deceptive | 0.88 | 0.86 | 0.87 | 0.88 |
| | Truthful | 0.87 | 0.89 | 0.88 | |
| Logistic regression: Unigrams | Deceptive | 0.88 | 0.79 | 0.83 | 0.84 |
| | Truthful | 0.81 | 0.89 | 0.85 | |
| Logistic regression: Bigrams | Deceptive | 0.86 | 0.84 | 0.85 | 0.85 |
| | Truthful | 0.84 | 0.86 | 0.85 | |
| Classification tree: Unigrams | Deceptive | 0.68 | 0.62 | 0.65 | 0.66 |
| | Truthful | 0.66 | 0.71 | 0.68 | |
| Classification tree: Bigrams | Deceptive | 0.70 | 0.65 | 0.68 | 0.69 |
| | Truthful | 0.67 | 0.72 | 0.70 | |
| Random forest: Unigrams | Deceptive | 0.88 | 0.81 | 0.84 | 0.85 |
| | Truthful | 0.83 | 0.89 | 0.86 | |
| Random forest: Bigrams | Deceptive | 0.92 | 0.72 | 0.81 | 0.83 |
| | Truthful | 0.77 | 0.94 | 0.85 | |

Table 11: Precision, Recall, F1 and Accuracy of All Models

## C Most important features

| Model | Feature | Majority direction | Models |
|---|---|---|---|
| **Linear Models** | | | |
| | elevator | Deceptive | [NB_unigram, NB_bigram] |
| | adult | Deceptive | [lg_unigram, lg_bigram] |
| | smell | Truthful | [NB_unigram, NB_bigram] |
| | luxury | Truthful | [NB_unigram, NB_bigram] |
| | millennium | Truthful | [NB_unigram, NB_bigram] |
| **Non-Linear Models** | | | |
| | chicago | Truthful | [dt_unigram, dt_bigram, rf_unigram, rf_bigram] |
| | turned | Deceptive | [dt_unigram, dt_bigram] |
| | location | Truthful | [rf_unigram, rf_bigram] |
| | smell | Truthful | [rf_unigram, rf_bigram] |
| | finally | Truthful | [rf_unigram, rf_bigram] |

Table 12: Top 5 Important Features for Linear and Non-Linear Models

| **Model** | **Feature** | **Direction** | **Importance** |
|---|---|---|---|
| Naive Bayes: Unigrams | smell | truthful | 1.4531 |
| | luxury | truthful | 1.4381 |
| | millennium | truthful | 1.4112 |
| | elevator | deceptive | 1.3290 |
| | priceline | deceptive | 1.2213 |
| Naive Bayes: Bigrams | elevator | deceptive | 0.3880 |
| | chicago millennium | truthful | 0.1182 |
| | millennium | truthful | 0.0645 |
| | smell | truthful | 0.0324 |
| | luxury | truthful | 0.0009 |
| Logistic Regression: Unigrams | chicago | truthful | 30.5963 |
| | smelled | truthful | 16.1574 |
| | recently | truthful | 15.6213 |
| | adult | deceptive | 14.5296 |
| | star | deceptive | 14.3043 |
| Logistic Regression: Bigrams | chicago | truthful | 27.2998 |
| | relax | truthful | 17.9409 |
| | booked hotel | deceptive | 17.7726 |
| | homewood suite | truthful | 16.7083 |
| | adult | deceptive | 16.5876 |
| Decision Tree: Unigrams | chicago | deceptive | 0.2958 |
| | turned | deceptive | 0.0506 |
| | decided | deceptive | 0.0505 |
| | east | deceptive | 0.0374 |
| | michigan | truthful | 0.0345 |
| Decision Tree: Bigrams | chicago | deceptive | 0.3167 |
| | turned | deceptive | 0.0419 |
| | finally got | deceptive | 0.0387 |
| | cool | truthful | 0.0386 |
| | ambassador east | deceptive | 0.0363 |
| Random Forest: Unigrams | chicago | truthful | 0.0569 |
| | location | truthful | 0.0230 |
| | smell | truthful | 0.0169 |
| | recently | truthful | 0.0157 |
| | finally | truthful | 0.0157 |
| Random Forest: Bigrams | chicago | truthful | 0.0205 |
| | location | truthful | 0.0070 |
| | smell | truthful | 0.0069 |
| | finally | truthful | 0.0066 |
| | seemed | truthful | 0.0062 |

Table 13: Top 5 Important Features for Each Model by Direction and Importance