Detecting Deceptive Opinion Spam using Machine Learning: An Empirical Study

Jens de Jonge (5298259)
Utrecht University
Utrecht, The Netherlands
j.w.dejonge@students.uu.nl

Akshaj Agarwal (1296884)
Utrecht University
Utrecht, The Netherlands
a.agarwal2@students.uu.nl

Daan Brouwer (9131491)
Utrecht University
Utrecht, The Netherlands
d.brouwer4@students.uu.nl

ABSTRACT

This study addresses the task of detecting deceptive hotel reviews using machine learning. We compare and evaluate five classifiers within a TF–IDF pipeline: Multinomial Naive Bayes (MNB), L1-regularized Logistic Regression (LR), Decision Trees (DT), Random Forests (RF), and Gradient Boosting (GB), trained with and without bigrams. We used data existing of 800 negative reviews (400 truthful, 400 deceptive) from Ott et al. [1, 2]. MNB achieved the best performance (accuracy = 0.87, F1 = 0.88), followed by RF (F1 = 0.86). Bigrams improved recall for deceptive reviews, while single trees showed overfitting. Deceptive reviews contained in general more persuasive and exaggerated wording, whereas truthful ones used more concrete, experience-based language.

1 INTRODUCTION

Online reviews are becoming increasingly important in our daily lives. There are even platforms dedicated entirely to customer reviews, such as TripAdvisor and Yelp, which play an essential role in shaping consumer decisions. However, their popularity has also led to the rise of opinion spam: deceptive reviews written to sound like real reviews. These fake reviews, often used to promote or damage a business, undermine trust in online platforms [1]. Detecting such deceptive reviews is a challenging text classification problem because deceptive language mimics genuine writing.

This study aims to detect deceptive hotel reviews using machine learning techniques. We apply and compare several classification approaches, including generative and discriminative linear models, as well as tree-based and ensemble methods. The dataset used in this study is introduced in Section 2, while the preprocessing steps, model configurations, and evaluation procedure are described in Section 3. Section 4 presents and discusses the main experimental findings, and Section 5 concludes the paper with a summary of key insights.

The study addresses the following research questions:

(1) How does the performance of the generative linear model compare to the discriminative linear model?
(2) Do ensemble methods outperform the linear classifiers?
(3) Does performance improve when bigram features are included?
(4) Which linguistic features most strongly indicate deceptive versus truthful reviews?

Proc. of the 19th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2020), May 2020, Auckland, New Zealand

https://doi.org/doi

By comparing these models under identical conditions, the study aims to identify which learning approaches and linguistic cues are most effective for detecting deceptive opinion spam.

2 DATA

The dataset used in this study is from the work of Ott et al. [1, 2]. It contains hotel reviews collected to study deceptive opinion spam. The data include both *truthful* and *deceptive* reviews. The *truthful* reviews were written by real customers and obtained from popular online platforms such as TripAdvisor, while the *deceptive* reviews were fabricated by crowdworkers on Amazon Mechanical Turk. Each review is labeled as either *truthful* or *deceptive*, allowing the task to be framed as a binary text classification problem.

The complete dataset consists of 1,600 reviews divided equally across four categories: positive truthful, positive deceptive, negative truthful, and negative deceptive. For the purpose of this study, only the 800 *negative* reviews are analyzed, evenly divided between 400 truthful and 400 deceptive examples. All reviews are written in English and vary in length from a few sentences to short paragraphs.

2.1 Data Preprocessing

The data preprocessing stage of this empirical data mining study was designed to convert raw hotel review texts into a clean, standardized, and model-ready dataset. The goal was to minimize noise, prevent bias, and ensure that subsequent analysis focused on linguistic patterns rather than superficial identifiers.

1. Data Loading and Labeling
   - The raw dataset consisted of multiple text files stored in a ZIP archive.
   - Each review was automatically labeled as “deceptive” or “truthful” based on the file path.
   - The hotel name was extracted from the filename to later remove references that might bias the model.
   - This automated labeling ensured consistency and reproducibility without manual intervention.
2. Basic Text Cleaning
   - To standardize and normalize the text:
   - Expanded common contractions (e.g., “don’t” → “do not”).
   - Removed HTML tags, punctuation, digits, and extra spaces.
   - Converted all text to lowercase.
   - These steps reduced variability in the data and ensured uniformity for further processing.
3. Linguistic Processing
   - A lightweight spaCy pipeline was used (tagger + named entity recognizer only) for efficient text parsing.

- Lemmatization with NLTK's WordNet lemmatizer converted words to their root forms (e.g., "running" → "run"), reducing lexical diversity and improving model generalization.

4. Stopword and Negation Handling

- Combined NLTK and spaCy stopword lists to remove uninformative words.
- Negation words such as not, no, never, and nor were retained to preserve sentiment and meaning.
- This ensured that the model would capture important contextual nuances in the reviews.

5. Noise and Leakage Reduction

- To prevent the model from learning spurious patterns:
- Removed proper nouns, named entities (e.g., people, locations), and hotel names.
- Filtered out tokens that were numeric, contained digits, or were too short (e.g., "th").
- These steps reduced the risk of data leakage and improved the reliability of linguistic features.

6. Final Dataset Creation

- Cleaned tokens were joined back into processed review texts.
- The results were stored in a structured CSV file with two columns:
- Review: the preprocessed text.
- Label: either "deceptive" or "truthful."

In summary, the text preprocessing pipeline effectively transformed raw hotel reviews into a clean, standardized, and unbiased dataset. By applying cleaning, lemmatization, stopword removal (while retaining negations), and filtering of names and entities, the process reduced noise and prevented data leakage. These steps ensured that the dataset captured meaningful linguistic patterns essential for accurate and reliable data mining and model training.

3 METHODS / EXPERIMENTAL SETUP

Each review was vectorized using a *CountVectorizer* followed by a *TfidfTransformer*. Two n-gram configurations were tested: unigrams (1,1) and unigrams+bigrams (1,2). Terms appearing in fewer than two documents were removed (`min_df = 2`), and the vocabulary was limited to 20,000 features. Both models were evaluated using the binary F1-score with

`make_scorer(f1_score, pos_label="DECEPTIVE")`.

3.1 Multinomial Naive Bayes

Feature Extraction

Each review was vectorized using a TF–IDF representation generated by a `CountVectorizer` followed by a `TfidfTransformer`. Two configurations were tested: unigrams (1,1) and unigrams+bigrams (1,2). Terms appearing in fewer than two documents were removed (`min_df = 2`), and the vocabulary size was limited to 20,000 features. Optionally, χ^2 feature selection with `SelectKBest` was applied to retain the top-k most informative terms.

Model Description

The `MultinomialNB` classifier from `scikit-learn` was used as the generative baseline for text classification. It estimates class-conditional probabilities under a multinomial distribution and applies Bayes' theorem to compute posterior class probabilities. This approach is well-suited for word count–based features and performs effectively on high-dimensional sparse text data.

Hyperparameter Tuning

The smoothing parameter $\alpha \in \{0.1, 0.5, 1.0\}$ and the number of top features $k \in \{500, 1000, 2000\}$ were tuned through 5-fold cross-validation on folds 1–4 of the dataset. The optimal configuration (`alpha = 1.0`, `k = 2000`) was selected based on the highest mean F1-score.

Evaluation Metrics

Model performance was assessed on the held-out fold 5 using accuracy, precision, recall, and F1-score for the `DECEPTIVE` class. The final model achieved an accuracy of 0.87 and an F1-score of 0.88. The confusion matrix is shown in Table 1.

Table 1: Confusion Matrix – Multinomial Naive Bayes

	Pred TRUTHFUL	Pred DECEPTIVE
True TRUTHFUL	65	15
True DECEPTIVE	6	74

3.2 Logistic Regression (L1 / Lasso)

Feature Extraction

The same TF–IDF feature extraction pipeline as used for Naive Bayes was applied. Both unigram and bigram features were tested to capture local word dependencies that may signal deceptive phrasing.

Model Description

The discriminative model was implemented using `LogisticRegression` from `scikit-learn` with an L1 (Lasso) penalty [3]. This model directly estimates $P(y|x)$ using the logistic function applied to a linear combination of input features. L1 regularization promotes sparsity by forcing less informative coefficients toward zero, which enhances interpretability and prevents overfitting in high-dimensional text data [4].

Hyperparameter Tuning

The inverse regularization strength $C = 1/\lambda$ was tuned over $\{0.01, 0.1, 1, 10, 100\}$ using 5-fold cross-validation on folds 1–4. The `liblinear` solver was selected for its compatibility with L1 regularization, and the maximum iteration limit was set to 5000. The best-performing model used `C = 10`.

Evaluation Metrics

Evaluation was performed on the held-out test fold 5 using accuracy, precision, recall, and F1-score. The model achieved an accuracy of 0.78 and an F1-score of 0.78. The confusion matrix is shown in Table 2.

Table 2: Confusion Matrix – Logistic Regression (L1)

	Pred TRUTHFUL	Pred DECEPTIVE
True TRUTHFUL	64	16
True DECEPTIVE	19	61

3.3 Single Classification trees (non-linear)

Feature Extraction
The model uses a TF-IDF Vectorizer to convert text reviews into numerical features. It generates both unigram and bigram representations of the text. The text is converted to lowercase, stopwords are removed, and the feature count is limited to 15,000. The resulting TF-IDF matrix is then converted to a dense format for use with the decision tree classifier.

Model Description
A Decision Tree Classifier is used to categorize reviews as either "DECEPTIVE" or "TRUTHFUL." Two models are trained: one using unigrams (DT-uni) and another using both unigrams and bigrams (DT-uni+bi). The data is divided into five folds, with four used for training and cross-validation, and one reserved for testing.

Hyperparameter Tuning
Hyperparameters such as max depth, minimum samples per leaf, and ccp alpha are tuned using grid search with 5-fold cross-validation. The model with the best parameters is retrained and compared with another model using a different ccp alpha value to observe the effects of pruning.

Evaluation Metrics
Model performance is measured using accuracy, precision, recall, and F1-score. Confusion matrices are created for each model, and the results are saved in a CSV file. The model with the highest F1-score is saved as the final decision tree model for further evaluation.

Table 3: Confusion Matrices for Decision Tree Models

Model	TT	TD	DT	DD
DT_uni ($\alpha = 1e{-}02$)	50	30	28	52
DT_uni ($\alpha = 1e{-}03$)	54	26	31	49
DT_uni+bi ($\alpha = 0$)	55	25	25	55
DT_uni+bi ($\alpha = 1e{-}02$)	50	30	20	60

3.4 Random forests (ensemble of trees)

Model description
The random forest classifier from *sklearn.ensemble* was used to build a model made up of many decision trees. Each tree is trained on a random part of the training data and a random selection of features. By combining the predictions of all trees using majority voting, the model becomes more stable and less likely to overfit.

Hyperparameter tuning
For both feature representations, the following hyperparameters were chosen to be optimized:

- Number of trees (*n_estimators*) for values: { 40, 80, 120, 160, 200, 240, 280 }
- Number of selected features (*max_features*) for values: { 10, 30, 50, 70, 90, 110, 130 }

The optimization was performed using out-of-bag (OOB) evaluation, selecting the parameter combination with the highest internal OOB score after training for the final predictions. In testing, the best combination of hyperparameters for the unigram model consisted of around 160 trees and 70 selected features, while for the bigram model this increased to 200 trees and 90 selected features.

Evaluation metrics
Under these respective parameters, the unigram model achieved an OOB score of 0.80 and a test accuracy of 0.78, while the inclusion of bigrams further improved the OOB score to 0.81 and test accuracy to 0.84. Moreover, the model with bigrams achieved higher recall and F1 scores, while slightly decreasing in precision. See Table 8 for all achieved scores. Additionally, both models identified the features 'location', 'recently', and 'decided' among their top five most influential indicators of deceptive text. Features more indicative of truthful reviews included 'declined', 'overcharge', and 'finished' for the unigram model, and 'historic hotel', 'smoking not', and 'towel took' for the bigram model.

Table 4: Confusion Matrix - Random Forest

	Pred TRUTHFUL	Pred DECEPTIVE
True TRUTHFUL	61	19
True DECEPTIVE	6	74

3.5 Gradient boosting (ensemble of trees)

Model description
The Gradient Boosting classifier from *sklearn.ensemble* was used to build an ensemble model that combines many shallow decision trees. Each tree in the sequence tries to correct the errors made by the previous ones, allowing the model to gradually improve its predictions. With this "boosting" approach the model aims to capture the more complex relationships in the data.

Hyperparameter tuning
To find the most effective parameter settings, a grid search was performed over the following hyperparameters:

- Tree depth (*max_depth*) for values: { 2, 4, 6, 8, 10 }
- Learning rate (*learning_rate*) for values: { 0.1, 0.3, 0.5, 0.7, 0.9 }

Ten-fold cross-validation was used during the grid search to find the best parameter combination for generalization. For both the unigram and unigram+bigram models, the best parameters were found to be a learning rate of 0.3 and a maximum tree depth of 2.

Evaluation metrics
Using these optimal parameters, the unigram model achieved a best cross-validation score of 0.79 and a test accuracy of 0.75, while the

unigram+bigram model reached a cross-validation score of 0.79 and a test accuracy of 0.76. The model using bigrams also achieved slightly higher precision, recall, and F1 scores, as shown in Table 8. Feature importance analysis showed consistent results across both models, with words such as 'location', 'luxury', 'recently', and 'smell' being strong indicators of deceptive texts. For the most truthful text, the unigram model found features 'ability', 'predicament', and 'prefer', while the unigram+bigram model found 'previously stayed' and 'previously opened'.

Table 5: Confusion Matrix - Gradient Boosting

	Pred TRUTHFUL	Pred DECEPTIVE
True TRUTHFUL	62	18
True DECEPTIVE	20	60

4 RESULTS AND DISCUSSION

Table 6: Top Deceptive Terms Across Models

Model	Top Deceptive Terms
DT	relax, fancy, egg, considered, sorely
RF	location, finally, decided, luxury, recently
GB	location, luxury, recently, smell, floor
LR (L1)	security, construction, location, rate, floor
NB	luxury, recently stayed, recently, smell, decided

From the Table 6 of top deceptive terms, it can be inferred that deceptive reviews tend to use more exaggerated, descriptive, and emotionally charged language. These reviews often focus on promoting the hotel experience, emphasizing positive qualities and creating an idealized impression rather than providing balanced feedback. The patterns across models suggest that deceptive writers attempt to sound persuasive by overusing adjectives or situational details, reflecting a crafted narrative rather than a genuine experience. Overall, the models consistently identify linguistic cues that indicate an intention to impress or convince, which are typical markers of deceptive communication.

Table 7: Top Truthful Terms Across Models

Model	Top Truthful Terms
DT	security, priceline, sofa, gift, directly
RF	absolutely no, work computer, worst customer, good staff, world class
GB	yeah right, able, able check, able find, able relax
LR (L1)	recently, decided, luxury, smelled, food
NB	elevator, security, construction, open, priceline

From the table of top truthful terms, it can be inferred that truthful reviews use more concrete, experience-based, and practical language. Genuine reviewers tend to describe specific aspects of their stay, such as services, facilities, or surroundings, without excessive emotional emphasis. The language appears more neutral and factual, focusing on personal experiences rather than promotional statements. Across models, the patterns indicate that truthful reviews are characterized by authenticity, detail, and straightforwardness, which help differentiate them from the overly expressive tone of deceptive reviews.

Table 8: Model Performance Summary (Rounded to Two Decimals)

Model	Accuracy	Precision	Recall	F1-score
NB_uni+bi	0.87	0.83	0.93	0.88
NB_uni+bi_topk	0.86	0.83	0.91	0.87
NB_uni_topk	0.85	0.80	0.94	0.86
RF_uni+bi	0.84	0.80	0.93	0.86
NB_uni	0.84	0.81	0.90	0.85
RF_uni	0.78	0.75	0.88	0.80
LR_L1_uni+bi	0.78	0.79	0.76	0.78
LR_L1_uni	0.77	0.79	0.74	0.76
GB_uni+bi	0.76	0.77	0.74	0.76
GB_uni	0.75	0.76	0.73	0.74
DT_uni+bi_alpha=1e-02	0.69	0.67	0.75	0.71
DT_uni+bi_alpha=0e+00	0.69	0.69	0.69	0.69
DT_uni_alpha=1e-02	0.64	0.63	0.65	0.64
DT_uni_alpha=1e-03	0.64	0.65	0.61	0.63

The results indicate that model selection strongly influences classification performance.
Among all approaches, **Naive Bayes models**, especially those using both unigrams and bigrams, achieved the highest accuracy and F1-scores, **showing their strength in handling sparse and frequency-based text data**.
Random Forest models performed comparably, demonstrating that ensemble methods provide **robust generalization and effective handling of linguistic variation**.

Logistic Regression (L1) achieved **balanced results**, indicating **good performance on linear relationships but limited ability to capture complex patterns**.

Gradient Boosting models performed consistently but slightly lower, suggesting **sensitivity to parameter tuning**.

In contrast, **Decision Tree models** showed the **weakest performance**, reflecting **overfitting and limited generalization**.

Overall, probabilistic and ensemble-based models outperformed single-tree approaches, confirming their suitability for text classification tasks in empirical data mining.

The pairwise heatmap visualizations provide insight into how similarly or differently the models perform across various evaluation metrics. Each heatmap shows the absolute difference between models for a given metric (Accuracy, Precision, Recall, and F1-score), allowing for a direct comparison of their relative consistency. **From these plots, it can be inferred that models belonging to the**

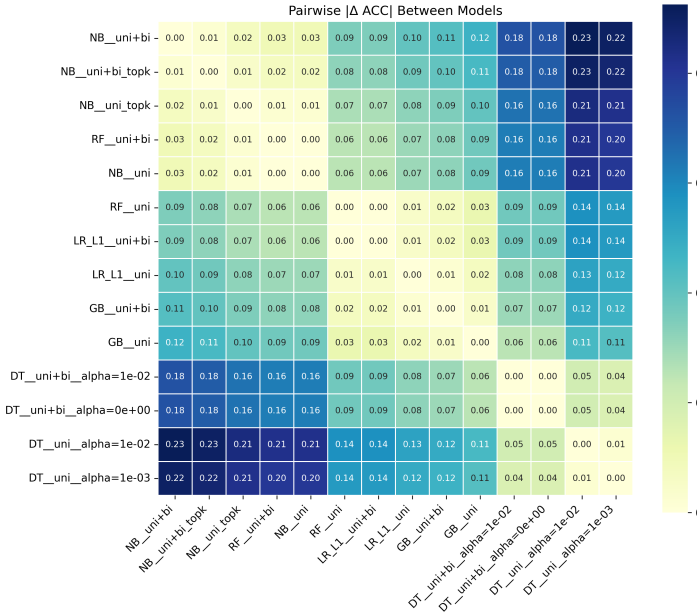

Figure 1: Pairwise Correlation of Model Accuracy

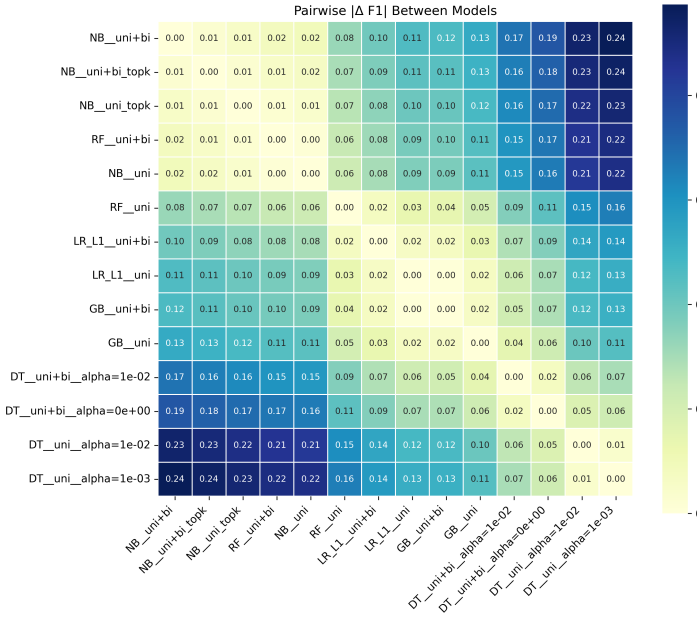

Figure 2: Pairwise Correlation of Model f1 values

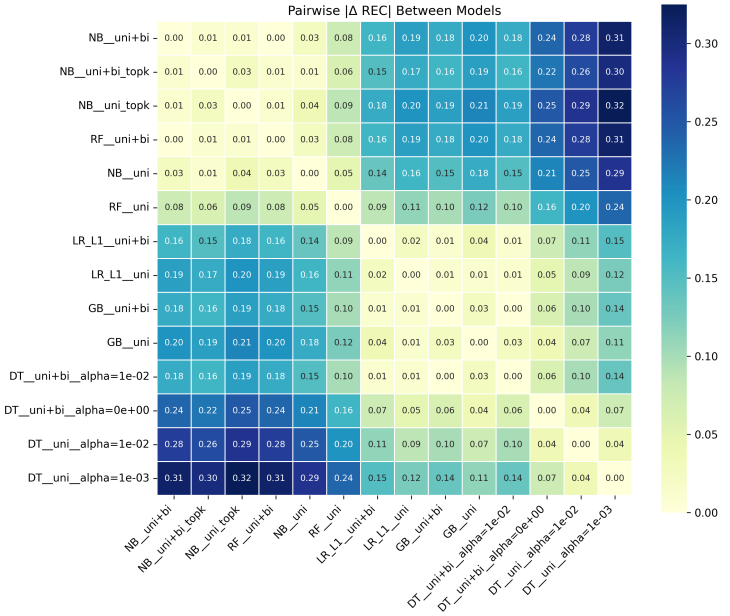

Figure 3: Pairwise Correlation of Model recall

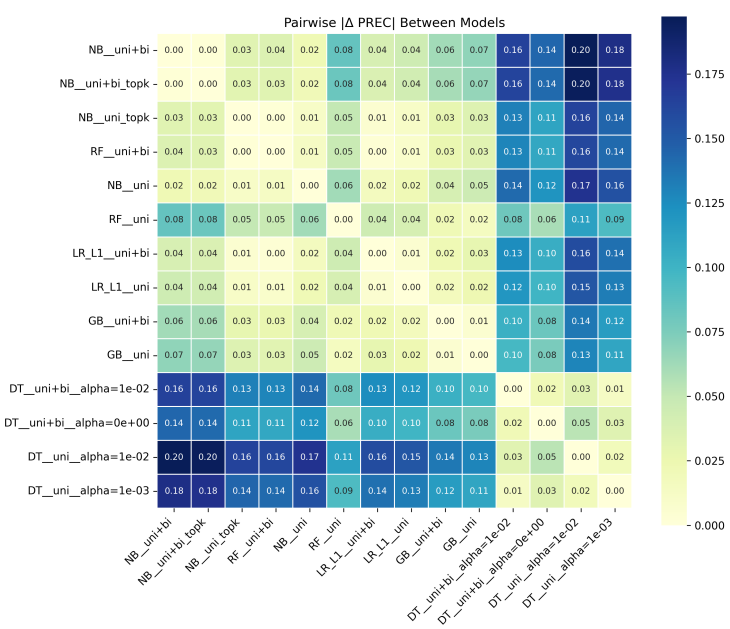

Figure 4: Pairwise Correlation of Model precision

same family—particularly the Naive Bayes and Random Forest variants—exhibit smaller pairwise differences, indicating stable and consistent behavior across configurations. In contrast, **the Decision Tree models display larger variation in pairwise distances, reflecting higher sensitivity to hyperparameter changes such as pruning (ccp-alpha)**.

The overall trend suggests that ensemble and probabilistic models maintain more uniform performance patterns, while single-tree models vary widely depending on parameter tuning. These findings reinforce that model robustness and stability are better achieved

through ensemble or regularized approaches in text classification tasks

5 CONCLUSIONS

Model Performance Analysis

Across all experiments, the **Multinomial Naive Bayes classifier** achieved the highest performance, with an **accuracy of 0.87 and an F1-score of 0.88**. This demonstrates that probabilistic models are highly effective for this dataset, as they efficiently handle sparse and high-dimensional text features.

The **Random Forest** model produced similarly strong results, particularly with an F1-score of 0.86, highlighting the robustness of ensemble tree-based methods in capturing non-linear patterns and reducing overfitting.

The **Logistic Regression model with L1 regularization** achieved balanced yet slightly lower performance, reflecting its ability to manage feature sparsity while maintaining interpretability.

Gradient Boosting models showed consistent but modest results, likely due to their sensitivity to hyperparameter tuning and the limited dataset size.

In contrast, **Decision Tree** models performed the weakest overall, indicating signs of overfitting and limited generalization ability.

Impact of Bigram Features and Linguistic Insights

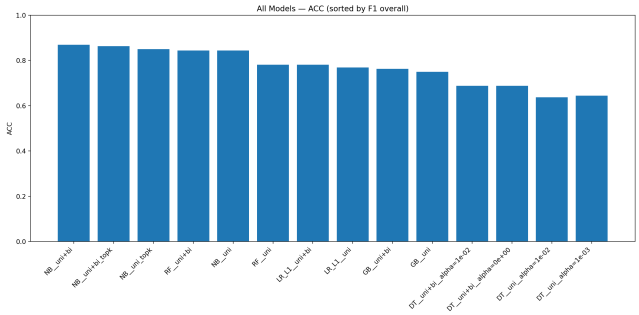

Figure 5: Bargraph comparison of Model Accuracy

The inclusion of **bigram features** led to noticeable improvements in recall for deceptive reviews across most models. This suggests that capturing local word combinations provides valuable contextual cues linked to deceptive phrasing patterns.

Linguistic analysis revealed that **deceptive reviews** tend to use expressive and persuasive language, emphasizing descriptive or emotionally charged terms, whereas **truthful reviews** focus on concrete experiences and specific details.

These observations are consistent with prior research in deception detection, supporting the notion that deceptive writing often relies on exaggerated and promotional wording, while truthful writing centers around factual, experience-based descriptions.

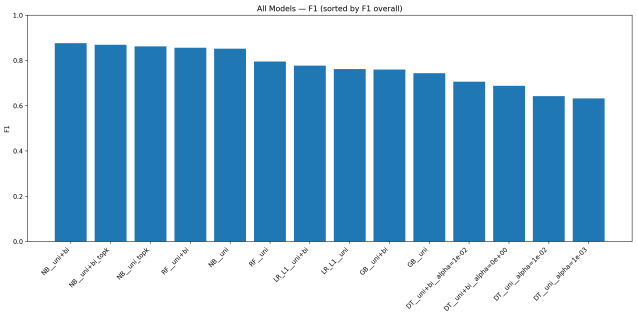

Figure 6: Bargraph comparison of Model f1

Research questions

(1) *How does the performance of the generative linear model compare to the discriminative linear model?*

The generative Multinomial Naive Bayes model outperformed the discriminative Logistic Regression model, indicating that probabilistic word frequency modeling better captures the structure of the deceptive texts in this dataset.

(2) *Do ensemble methods outperform the linear classifiers?*

Ensemble methods, particularly Random Forests, achieved performance comparable to Naive Bayes and generally higher than Logistic Regression, confirming that ensemble models can effectively handle complex feature interactions while maintaining robust generalization.

(3) *Does performance improve when bigram features are included?*

Yes. The inclusion of bigram features improved recall for deceptive reviews across almost all models, especially for ensemble approaches, demonstrating that capturing short contextual word patterns enhances model sensitivity to deceptive phrasing.

(4) *Which linguistic features most strongly indicate deceptive versus truthful reviews?*

Deceptive reviews were characterized by more persuasive and descriptive terms such as *location*, *luxury*, and *recently*, while truthful reviews more often contained factual and experience-based language such as *security*, *sofa*, and *priceline*. These linguistic differences suggest that deceptive texts emphasize impression management, whereas truthful ones focus on tangible experiences.

Overall conclusion.

In conclusion, the results show that lightweight models such as Naive Bayes and Random Forests are well-suited for detecting deceptive opinion spam, offering a good balance between performance, interpretability, and computational efficiency. Ensemble and probabilistic methods consistently outperformed single-tree and linear discriminative approaches, while bigram-based TF–IDF features enhanced the detection of subtle contextual cues. Future work could explore more advanced feature representations, such as contextual embeddings or sentiment-based features, to further improve the detection of deceptive language.

Link to github : https://github.com/Akshaj1017/Data-Mining-Assignment-Group-2/tree/main

6 REFLECTION (AI TOOLS)

We occasionally used generative AI tools to clarify methodological concepts and verify common parameter ranges for the models applied in this assignment. For instance, we asked for explanations about suitable hyperparameter values for algorithms such as Logistic Regression, and why these ranges are commonly effective. The information was used to cross-check our own understanding before performing systematic cross-validation.

REFERENCES

[1] Myle Ott, Yejin Choi, Claire Cardie, and Jeffrey T. Hancock. Finding deceptive opinion spam by any stretch of the imagination. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pages 309–319. Association for Computational Linguistics, 2011.

[2] 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. Association for Computational Linguistics, 2013.

[3] Trevor Hastie, Robert Tibshirani, and Martin Wainwright. *Statistical Learning with Sparsity: The Lasso and Generalizations*. MIT Press, Cambridge, MA, 2015.

[4] Thanh Nguyen and Son Tran. An overview of regularization techniques in machine learning. *Applied Sciences*, 12(15):7567, 2022. doi: 10.3390/app12157567.