Group 29 Guilherme Almeida Patrão – 99230 | Raquel Filipa Marques Cardoso – 99314

Mini-Project Nº 2 (MP2) Report

Truthful vs. deceptive hotel reviews

Models

Throughout the course of this project, we worked with three distinct machine learning models: Logistic Regression, Support Vector Classifier (SVC), and Naïve Bayes. Prior to employing these models, the data underwent a text pre-processing pipeline, which included lowercasing, the removal of non-alphabet characters, handling contractions, removing punctuation and stop-words, and subsequently lemmatizing each word. Following pre-processing, we applied feature extraction using the TF-IDF model. The was consistently set at 0.125 for all models, since the database consisted of 1400 lines in **`train.txt`**, the training set, and 200 lines in **`text_just_reviews.txt`**, the testing set.

We commenced with a Logistic Regression model, which is well suited for scenarios where a linear relationship exists between the features and the target variable. Fine-tuning involved adjusting parameters like regularization strength **`(C)`** and regularization type **`(penalty)`** to optimize performance.

Subsequently, we progressed to a SVC model, known to be effective in handling both linear and non-linear data. To fine-tune this model, we utilized Grid Search to explore a range of hyperparameters. We examined different values for **`'C'`**(the regularization parameter) and the consideration of **`'linear'`** and **`'rbf'`** as **`'kernel'`**. To ensure the model's generalization capabilities, we employed a 3-fold cross-validation.

Afterwards, we implemented a Multinomial Naïve-Bayes model, which assumes conditional independence between features, and is well-suited for text classification tasks, making it an ideal choice for this classification problem. To enhance its performance, we integrated GridSearch into a Pipeline configuration. Within it, we combined the model with a CountVectorizer and a TF-IDF transformer, allowing us to convert text data into numerical features and transform these features into a numerical representation that considers the importance of words in the entire dataset. The model underwent a 5-fold cross-validation process to determine the most suitable settings for each component.

Ultimately, Naïve-Bayes was our final choice, since it showcased the highest accuracy score among the models explored.

Experimental Setup and Results

In the experimental setup for this project, our primary evaluation measure was accuracy, serving as the metric for assessing the performance of our text classification model. We compared our models' accuracy against various baselines:

1. The weakest baseline, based on the Jaccard metric, achieved an accuracy of 58.5%;
2. A stronger baseline of 80.0% accuracy rate;
3. And a bonus baseline, to demonstrate our model's potential, of 85.0% accuracy rate.

Below you can find a confusion matrix of our chosen model and a tabulated summary of all model accuracies when evaluated against our own test set, derived from the train.txt file:

Labels/Labels	Deceptive Negative	Deceptive Positive	Truthful Negative	Truthful Positive
Deceptive Negative	44	0	5	0
Deceptive Positive	0	39	0	2
Truthful Negative	6	1	33	1
Truthful Positive	1	2	1	40

Table 1 – Confusion matrix of the Multinomial Naïve-Bayes model.

Labels/Models	Logistic Regression with TF-IDF Vectorizer	Support Vector Classifier (SVC) with Grid Search and TF-IDF Vectorizer	Multinomial Naïve-Bayes with CountVectorizer, TF-IDF Transformer and Grid-Search
Deceptive Negative	85.71%	83.67%	89.80%
Deceptive Positive	90.24%	92.68%	95.12%
Truthful Negative	85.37%	87.80%	80.49%
Truthful Positive	88.63%	88.64%	90.91%
Global Accuracy	87.43%	88.00%	89.14%

Table 2 – All models accuracies in detail.

We selected the Naïve-Bayes model as the final choice due to its outstanding performance, achieving the highest accuracy across all labels, except for **'Truthful Positive'**, where the Support Vector Classifier (SVC) outperformed it with a 7.31% higher accuracy. Nonetheless, the Naïve-Bayes model outperformed both the Logistic Regression and Support Vector Classifier models in all other labels and achieved the best overall accuracy.

We also explored a Logistic Regression model using a TF-IDF Vectorizer, which exhibited noteworthy performance, especially in the **'Truthful Positive'** category, surpassing our chosen Naïve-Bayes model by 4.88%. However, when evaluating accuracy globally and across different labels, the Logistic Regression model fell slightly short in comparison to both the Support Vector Classifier (-0.57%) and the Naïve-Bayes model (-1.71%).

Before selecting the Naïve-Bayes model with Count Vectorizer and Grid-Search, we implemented the same approach on the SVC and Logistic Regression models. However, these variations yielded lower results compared to their original implementations, which led us to revert that implementation and ultimately select the Naïve -Bayes model.

Our final Naïve-Bayes model surpassed all baselines when tested against our own test set, demonstrating its potential to generalize and outperform the same baselines when evaluated with different test datasets, such as **`test_just_reviews.txt`**.

Discussion

It is essential to address the model's performance, particularly its response to challenging cases.

One challenging case was the following: *"went to chicago (...) sat & sun morning."* (line 409 on train.txt). This example showcases a review that our model classified as "deceptive negative" while the training data indicated it as "truthful positive". This discrepancy can be attributed to a combination of factors.
First, the review, after preprocessing including stop words removal, contains mixed sentiments, commencing with expressions of disappointment regarding a crowded hotel due to a convention, which could be perceived as negative. However, as the narrative unfolds, the reviewer mentions being upgraded to an executive level room with comfortable amenities, indicating a more positive experience. This mixed sentiment and nuanced language may have confused our model. Additionally, the presence of negative words such as "disappointed" and "unhelpful" could have led to negative sentiment misinterpretation errors, potentially influencing the model's classification. Furthermore, the model's inability to capture the overall context and consider the temporal flow of sentiments in the review may have played a role in this misclassification.

Another challenging case was the following: *"I must say (...) do in Chicago."* (line 1098 on train.txt). In this review, we encounter another instance of disparity between our model's classification, which is "truthful-positive", and the training data's classification, marked as "deceptive-positive".
The divergence in categorization for this particular review could be attributed to several factors. First and foremost, the review begins with highly positive language, with the reviewer praising the hotel's beautiful penthouse, fitness center, pool, and well-designed rooms, along with the great linens and pillows. This positive sentiment throughout the initial part of the review aligns with our model's classification as "truthful-positive".
However, as the review progresses, the phrase *"change pace boring hotel bathroom"* is introduced, which can be interpreted as subtly derogatory or ironic. The model may not fully capture the nuances of this language, potentially leading to a different classification. Moreover, the use of the term "deceptive-positive" in the training data may indicate a pattern in which reviewers present their opinions in a manner that appears overly positive but might contain subtle negative undertones.

In conclusion, we have identified two prevalent and common errors in our sentiment analysis: negative sentiment misinterpretation errors and ambiguity in hotel reviews. Negative sentiment misinterpretation errors are associated with the model's challenge in correctly handling words and phrases with negative sentiment connotations, potentially leading to misclassifications in mixed sentiment contexts. Ambiguity, as demonstrated in the second case, arises from the complex interplay of positive and negative language, often challenging the model's comprehension.
Furthermore, it's crucial to acknowledge the impact of stop words in context understanding. Removing certain stop words may disrupt the contextual flow, causing the model to misunderstand the intended sentiment and potentially leading to misclassifications. These observations shed light on the importance of context-aware and nuanced sentiment analysis, emphasizing the need for continual improvement in handling negative sentiment words and ambiguity, as well as the careful consideration of stop words removal.

Future Work

In future work, we aim to further enhance our text classification system by improving its contextual understanding and addressing the challenges associated with negative sentiment analysis. One of our key objectives is to develop a Stacking Classifier based on the three models we've created during this task. This approach would allow us to capture the potential complementarity between these models. For our meta classifier, we might consider employing models like the Gradient Boosting Classifier, which can effectively complement the base models. Furthermore, we plan to fine-tune the meta classifier's parameters through a Grid Search to ensure optimal generalization.
Additionally, we intend to investigate the impact of stop words on sentiment analysis. We will explore strategies for handling stop words more contextually, ensuring that their presence or removal aligns with the overall sentiment of the text. These combined efforts are expected to make a significant contribution to the system's accuracy and its capability to effectively handle complex sentiment expressions.

Bibliography

Otten, N. (2023, February 22). How To Implement Logistic Regression Text Classification In Python With Scikit-learn and PyTorch. Spot Intelligence. https://shorturl.at/vwKQT
Bedi, G. (2918, November 9). A guide to Text Classification(NLP) using SVM and Naive Bayes with Python. Medium. https://shorturl.at/duHLN
Chaudhary, M. (2020, September 4). SKlearn: Pipeline & GridSearchCV. Medium. https://shorturl.at/fjkV6