Sentiment Classification for Restaurant Reviews

Amr Mohamed
Exchange student from CY Tech
France to GU CSE Department
amrabdelraheem9@gmail.com

Anh Thu DOAN
Exchange student from CY Tech
France to GU CSE Department
thudoann45@gmail.com

Abstract

In this paper, we will go through the process of implementing a Machine Learning classifier that aims to classify whether online-given restaurant reviews are positive or negative. Moreover, we will see the data preprocessing, cleaning, and vectorization techniques that were applied. In addition, we will go through the process of model selection, hyperparameter tuning, evaluation, and interpretation.

1 Introduction

Online customer feedback has been recognised as an economic value for consumers and restaurants. With the expansion of the internet and social networks, customer reviews significantly influence business growth and gain new consumers. Therefore, the classification between negative and positive reviews contributes a massive benefit for the business to provide better quality services and minimise the harmful effects of bad reviews.
This project aims to use Supervised Learning Text Classification to predict whether given feedback(s) is(are) positive or negative.

2 Methods

2.1 Data description

The data was gathered by many contributors, who contributed to collecting thousands of reviews about restaurants and their annotation (positive or negative reviews), as shown in Table 1 below. The dataset consisted of two main parts, a training dataset and a testing dataset; where the training dataset consisted of 2 columns, annotation and review, and 7018 instances. Then the training set annotation column is composed of the annotations given to a study (1 for positive review and 0 for negative review) by two different annotators and put together in one column separated by a '/'. The other column is the text of the study.

	annotation	**review**
0	1/1	The restaurant was amazing.
1	0/0	It was a bad experience.

Table 1: A sample of the training dataset

The testing set consisted of 2 columns, annotation and review, and 1751 instances while having the same structure as the training set. Still, the annotation of each study is just a single value, which means that its value was given by just one annotator, as shown in Table 2 below.

	annotation	**review**
0	1	It was a great experience!
1	0	I won't be back here again.

Table 2: A sample of the testing dataset

2.2 Data processing and cleaning

We began by processing the training data to start the data cleaning procedure. Firstly, we split the annotation column into two columns, one for the first annotation and another for the second annotation, to gain information about the percentage of similarity/difference between the two annotations. We found that the annotators of the training set reviews disagreed in the annotation task 5.6% of the time in total; 68% of those annotations had a genuine disagreement between the two annotators (1/0 and 0/1). In comparison, the rest, 32%, was a disagreement out of a typo from one of the annotators.
To deal with the reviews that had disagreements, we performed sentiment analysis of these rows and updated the values with the new annotation of the sentiment analysis. Finally, we had a final annotation column consisting of 2 classes, 0 and

1, 0 represents the negative reviews, 1 represents the positive reviews, 3617 instances of the training data belonged to the positive class, and 3324 instances belonged to the negative class.
To finalise the data cleaning process, all reviews' text was set to lower case to reduce the size and not have more than one version of the same word in the vocabulary set. Moreover, we have tried other approaches like removing stopwords, but it tended to decrease the performance of the models as some of the stop words were highly contributive to the models' classification performance.

2.3 Feature pre-processing

To pre-process the text data we have from the reviews, we used a TF-IDF vectorizer, an algorithm that measures the relevance of words in a document among a set of documents.

$$tf - idf(t,d) = tf(t,d) * idf(t) \quad (1)$$

where

$$idf(t) = log[n/df(t)] + 1 \quad (2)$$

In the equations (1), *tf* represents the term (word) *t* frequency in document *d* across the rest of the documents in a given corpus, while *idf* is the inverse document frequency that measures how frequent is the term among all the documents in the corpus. In the case of our study, the documents are the texts of the reviews.
The TF-IDF vectorizer vectorised the reviews' text to 11260 features (words) to be compatible with each of the to-be implemented classifiers as numerical values to be trained and evaluated.

3 Results

3.1 Learning algorithm selection

Moving forward to the classification task, we started it by choosing a trivial baseline classifier which predicts whether a review is positive or negative based on the class of the **most-frequent** class in the training dataset. The baseline line model achieved an accuracy of approximately 50% when tested on the unseen data of the testing set.
To move forward, we implemented several classifiers, from which the highest performers were the Linear Support Vector Classifier (LSVC) and Support Vector Machine Classifier (SVC) by achieving 96.685% accuracy for each of them on the testing set, as shown in Table 3 below, showing an improvement of 46.85% in the classification performance compared to the baseline classifier. Therefore, we chose Linear SVC and SVC to move forward and fine-tune to obtain higher accuracy.

Model	Accuracy
Linear SVC	96.85%
SVC	96.85%
Multilayer perceptron	96.6%
Logistic Regression	96.2%
Multinomial Naive Bayes	94.7%
Random Forest	93.7%
Gradient Boosting	91.7%

Table 3: Classifiers' accuracies on the testing set

3.2 Learning algorithm hyper-parameters tuning

. Moving forward to the hyperparameter tuning of the models, we started by tuning the linear SVC model using a grid search where it didn't show much of an improvement from the accuracy acquired through the default parameters of the model. By taking a step back, we tuned the TF-IDF vectorizer and the Linear SVC model to maximise the model's accuracy by searching the different parameters of both the vectorizer and the model. We found the TF-IDF hyperparameter **max document frequency (max_df)** which sets a threshold of a ratio of the total number of reviews that if a word occurs in a ratio of reviews higher than the threshold, the vectorizer ignores it. By tuning the max document frequency parameter.

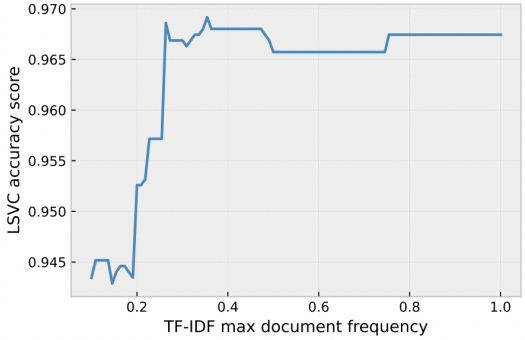

Figure 1: LSVC accuracies over the max document frequency parameter values of TF-IDF

From Figure 1, we can see that the accuracy of the Linear SVC model is maximised to an accuracy of 96.92% when the max_df parameter is equal to

approximately 0.354.

Afterwards, we performed the same steps as the steps followed for Linear SVC hyperparameter tuning for the SVC model.

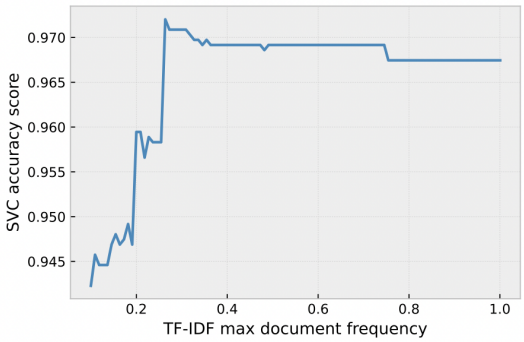

Figure 2: SVC accuracies over the max document frequency parameter values of TF-IDF

From Figure 2, we can see that the accuracy of the SVC model is maximised to an accuracy of 97.2% when the max_df parameter is equal to approximately 0.26.

3.3 Model evaluation

Several types of classifier models were used to classify the data. The primary evaluation score we used is the accuracy score to define which model's performance is better than the others. As a result, we continue comparing the two highest model's accuracy, Linear Support Vector Classifier (LSVC) and Support Vector Machine Classifier (SVC), using a confusion matrix.

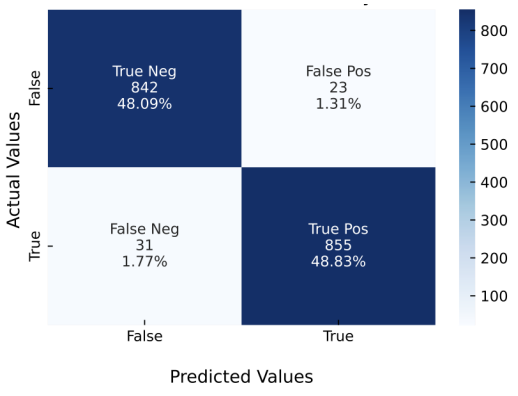

Figure 3: LSVC confusion matrix

In Figure 3, we represent the confusion matrix using the Linear Support Vector Classifier (LSVC) with the hyperparameter max_df approximately equal to 0.35. The first cell represents 842 negative reviews correctly classified by the model over 865 total negative reviews (True negatives). The correctly categorised negative reviews are 48.09% over the whole data set. In the next cell, the False positives instances were 1.31% of the data, which are the negative reviews were predicted as positive ones. Similarly, in the model's classification process of the positive reviews, the percentage of the false negatives was 1.77%, meaning that this proportion of data was positive reviews. The model predicted them as negative ones.

Reviews	Precision	Recall
Negative	0.96	0.97
Positive	0.97	0.97

Table 4: Precision and recall table of Linear SVC

A shown in Table 4, the LSVC model had a precision on the negative reviews of 0.96 which means that for the reviews that were classified negatively by the LSVC model, 96% of which were negative reviews, while the precision of the positive class is 0.97 which means that for the reviews that were classified positively by the LSVC model, 97% of which were positive reviews. Moreover, the recall represents how many classes were predicted correctly from the positive classes. In both cases, we got the recall with 0.97. For each of both classes, the model managed to classify 97% of the reviews; for example, for the positive reviews class, the model classified 97% of the instances with positive annotation as positive.

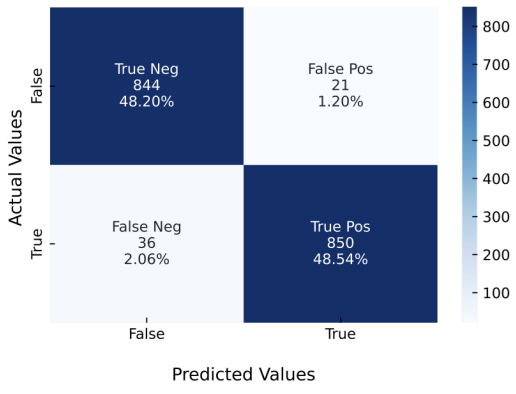

Figure 4: SVC confusion matrix

As demonstrated in the confusion matrix of the tuned SVC model (Figure 4), the true negative

was 48.20%, which is higher than the true negative when we used the Linear SVC model 0.11%. However, the percentage of the true positive was 48.54%, and it was lower than the LSVC model, which was 48.83%. This model's false positive and false negative are 1.20% and 2.06%, respectively; the difference between them is 0.86%. On the other hand, the difference between those in LSVC is lower, which is 0.46%.

Reviews	Precision	Recall
Negative	0.96	0.98
Positive	0.98	0.98

Table 5: Precision and recall table of SVC

From the data given in Table 4 we can infer that we got a higher recall for both classes, 98%, than the LSVC model. The precision was also higher for the positive class, and we got the same precision for the negative class with 96% compared with the previous model.

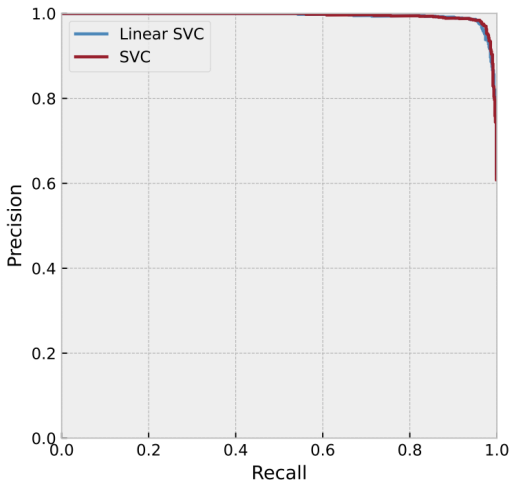

Figure 5: Precision/Recall curve of Linear SVC and SVC

As shown in Figure 5, the two curves of both models had the same trend as both curves were near the top right corner. Both precisions only decreased to 0.6 when the recall gradually increased to 1.

Finally, since the trade-off between recall and precision of Linear SVC is less than that of SVC. The difference in accuracy is very slight (0.25%), and putting into consideration that the Linear SVC model is more interpretable than SVC since the input features have weights that directly affect the model's classification performance, we choose the Linear SVC model as the final model to use and interpret.

3.4 Feature Importance and Model Interpretability

To interpret the selected Linear SVC model, we inspect the features (words) that contribute to the model's classification performance.

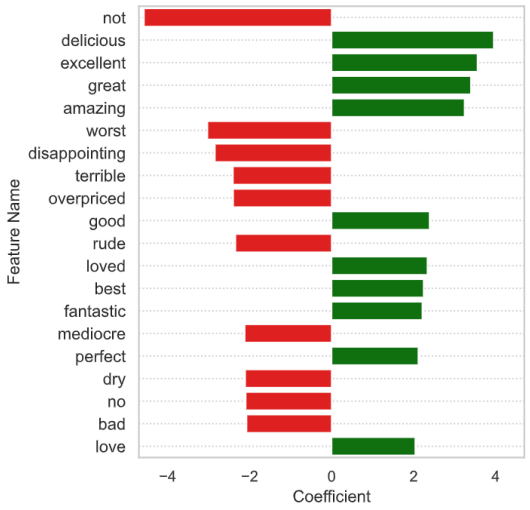

Figure 6: Top 20 features contributing to Linear SVC classification performance

In Figure 6, we can see the highest 20 words in coefficients, which means that those words were the highest contributor to the model in identifying whether a review was a negative review or a positive review, so we can see that the word *not* the most contributing to the model in identifying negative reviews with a coefficient of approximately -4.5, as well as the terms *worst, disappointing, terrible, and overpriced etc.*. were highly contributing in the same task. On the other hand, the word delicious was found to be the highest contributor to the model in the task of positive reviews identification with a coefficient of approximately 4, as well as the words *excellent, outstanding, unique, good etc...*

4 Conclusion

To conclude our study, we have developed a Linear Support Vector Classifier model that classifies customer reviews of restaurants with an accuracy approximately equal to 97% on an unseen dataset. The model classification performance was then

evaluated through a confusion matrix and the calculation of the precision and recall, which showed no trade-off between them. Moreover, We have tackled the aspect of model interpretability. We were able to retrieve the weights of the model's coefficients that represent how the different words affect the model's output when seen in the input text.

References

TF-IDF :
https://en.wikipedia.org/wiki/Tf\%E2\%80\%93idf

Yin-Wen Chang, Chih-Jen Lin (2008). "Feature Ranking Using Linear SVM". :
http://proceedings.mlr.press/v3/chang08a.html