Sentiment Analysis on Movie Reviews

Fotimakhon Gulomova
Bachelor of Science in Applied Artificial Intelligence
IU International University of Applied Sciences, Germany
fatimakhongulomova@gmail.com
fatima.gulamova@iu-study.org

Abstract
I worked on sentiment analysis for movie reviews. The main purpose of this report is to categorise reviews into two groups: positive and negative. I used two machine-learning models, Logistic Regression and Random Forest to identify the most effective approach for sentiment classification on the popular IMDB Dataset of 50K Movie Reviews dataset.

---

1. Introduction

In this work, I explored sentiment analysis on movie reviews. The main goal of this project was to find the most effective machine-learning model for sentiment classification. The IMDB Dataset of 50K Movie Reviews dataset was used.
I used two machine-learning models: Logistic Regression and Random Forest. Every technique was selected for its unique strengths, for example, Logistic Regression for its simplicity and effectiveness and Random Forest for its robustness and to reduce overfitting (GeeksforGeeks, 2023).
The experimental setup consists of preprocessing the dataset through tokenisation and vectorisation for the machine-learning models, training each model and evaluating these models using standard metrics such as accuracy, precision, recall and F1 score.

2. Method

I used two machine-learning models to classify the dataset in this work: Logistic regression and Random forest.

1. Logistic Regression

- **Description:** Logistic Regression is a linear model for binary classification tasks and is easy to apply in machine learning. It analyses the relationship between independent variables and classifies data into discrete classes (Kanade, 2024).

$$y = \frac{e^{b0+b1X}}{1 + e^{b0+b1X}}$$

- **Implementation:** I used the TF-IDF vectors to train the model.

```
from sklearn.linear_model import LogisticRegression

logreg = LogisticRegression(random_state=42, solver="saga")
logreg.fit(X_train, y_train)
y_pred_lr = logreg.predict(X_test)
```

2. Random Forest
   - **Description:** Random Forest is a machine learning technique which combines the output of multiple decision trees to produce a single result. It is easy to use and flexible because it can be used for both classification and regression problems (*What Is Random Forest? | IBM*, n.d.).
   - **Implementation:** I used TF-IDF vectors and multiple decision trees to improve classification performance for training the model.

```
from sklearn.ensemble import RandomForestClassifier

model = RandomForestClassifier(random_state=42)
model.fit(X_train, y_train)
y_pred_rf = model.predict(X_test)
```

3. Experimental Setup

Data

As mentioned, I used the IMDB Dataset of 50K Movie Reviews dataset from Kaggle. This dataset consists of 99,582 movie reviews labelled as positive or negative. I took the sample of this dataset only 30% of the dataset, which consisted of 30,000 movie reviews.

#	Column	Non-Null Count	Dtype
0	review	15000 non-null	object

1	sentiment	15000 non-null	object

Then, to process the dataset, I followed these steps:

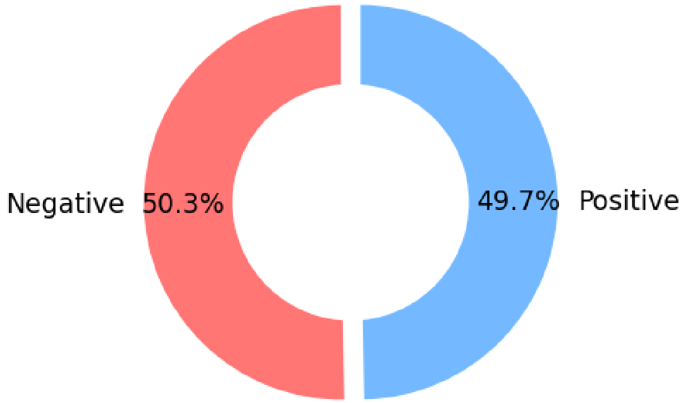
1. **Text Cleaning:** To remove HTML tags, punctuation and non-alphabetic characters.
2. **Tokenisation:** To split the text into separate words (Jain, 2024).
3. **Stop Words Removal:** To drop common words as they do not have a significant meaning (Jain, 2024).
4. **Lemmatisation:** To reduce words to their base or dictionary form (lemma) (Jain, 2024).

```
def clean_review(text, wl=WordNetLemmatizer(),
               stop_words=set(stopwords.words('english'))):
    # Remove HTML tags
    cleaned_text = remove_html_tags(text)

    # Converts to lowercase and splits up the words
    words = word_tokenize(cleaned_text.lower())

    filtered_words = []

    for word in words:
        # Remove the stop words and punctuation
        if word not in stop_words and word.isalpha():
            filtered_words.append(wl.lemmatize(word))

    filtered_words = ' '.join(filtered_words)

    return filtered_words
```

Exploratory Data Analysis (EDA)

Then, I conducted an Exploratory Data Analysis (EDA) to understand the distribution and characteristics of the data.

The dataset is roughly balanced, with a nearly equal number of positive and negative reviews this means that the models do not deal with a bias towards one class. Next, I identified the most common words in positive and negative reviews.



Feature Extraction

I converted the text data into numerical features using the TF-IDF vectorization method. Term frequency-inverse document frequency (TF-IDF) is a technique that can be divided into two parts: TF (term frequency), which measures how often a term appears in a document, and IDF (inverse document frequency), which determines how important a term is in the entire corpus. TF-IDF is the product of TF and IDF (Anirudha Simha, 2021).

$$tf - idf(t, d) = tf(t, d) \times idf(t)$$

```
count_vect = CountVectorizer(ngram_range=(1, 2))
transformer = TfidfTransformer(norm='l2', sublinear_tf=True)
```

```
counts = count_vect.fit_transform(df_prep['Preprocessed Review'])

X = transformer.fit_transform(counts)
y = np.array(df_prep['Target'].values, dtype='float64')
```

Data Division

I split the dataset into training and validation sets, with 80% of the data used for training and 20% for validation. This means that a main part of the data was used to train the model and a sufficient amount was used to evaluate model performance.

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)
X_train.shape, X_test.shape, y_train.shape, y_test.shape

# Output: ((11965, 1102714), (2992, 1102714), (11965,), (2992,))
```

Evaluation Method

I used the following metrics to evaluate the model performances:

1. **Accuracy:** This metric calculates how accurate the model is by dividing the correct predictions by the total predictions. However, this metric may not be suitable for classifying imbalanced datasets. (MarkovML, 2023).

$$Accuracy = \frac{TP + TN}{N}$$

2. **Precision:** This metric measures the ratio of true positives to the total number of positive predictions (MarkovML, 2023).

$$Precision = \frac{TP}{TP + FP}$$

3. **Recall:** Recall calculates the proportion of true positives to the sum of true positives and false negatives (MarkovML, 2023).

$$Recall = \frac{TP}{TP + FN}$$

4. **F1 Score:** The harmonic mean of precision and recall, providing a balanced measure of the two (MarkovML, 2023).

$$F1Score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

4. Results and discussion

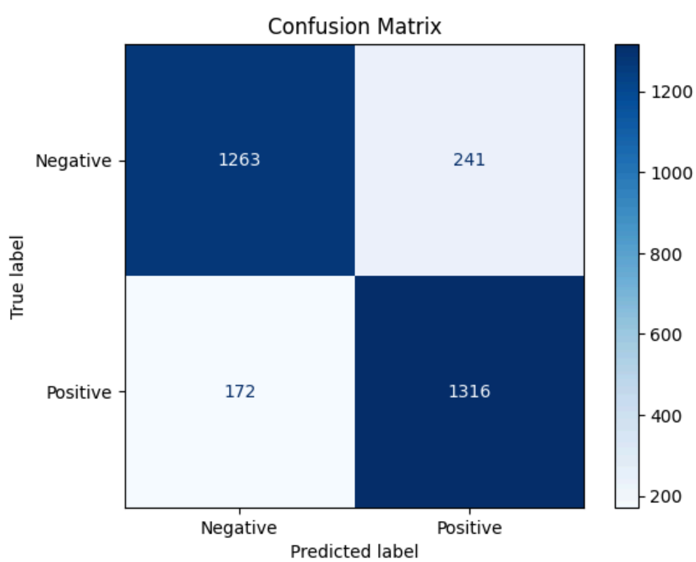
Model Performance

The models' performance was evaluated using accuracy, precision, recall, and F1 score metrics and they were trained and validated on the IMDB Dataset of 50K Movie Reviews.

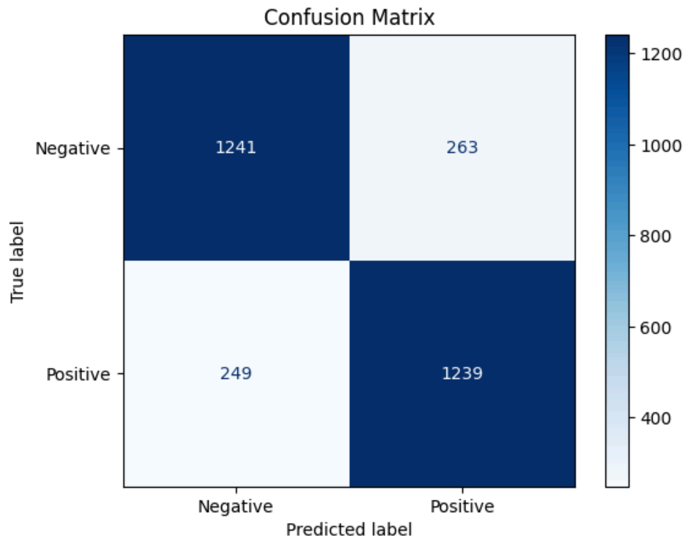
Model	Accuracy	Precision	Recall	F1 Score
Logistic Regression	0.86	0.88	0.86	0.86
Random Forest	0.83	0.84	0.83	0.83

While the Logistic Regression model presented robust performance, achieving a high accuracy of 86% and the balance between precision and recall indicated that the model effectively classified both positive and negative reviews, the Random Forest model achieved a slightly lower accuracy of 83%.

Logistic Regression



Random Forest



5. Conclusion

In this study, I investigated the effectiveness of sentiment analysis models in classifying movie reviews as positive or negative using a publicly available dataset from Kaggle called IMDB Dataset of 50K Movie Reviews and two machine learning models: Logistic Regression and Random Forest were implemented and evaluated using metrics.

Kaggle: https://www.kaggle.com/fotimakhongulomova/sentiment-analysis-on-movie-reviews
GitHub: https://github.com/fatimagulomova/iu-projects/tree/main/DLBAIPNLP01

Possibilities for future work

To gain a better understanding of sentiment in movie reviews in my future work, I plan to explore more advanced models, such as deep learning techniques like Long Short-Term Memory (LSTM), and to enhance the generalizability of the model I will use a more diverse dataset which were drawn from different sources and languages.

References

Anirudha Simha. (2021, October 6). *Understanding TF-IDF for Machine Learning | Capital One*. Capital One. https://www.capitalone.com/tech/machine-learning/understanding-tf-idf/

GeeksforGeeks. (2023, June 8). *Logistic regression vs random forest classifier*. GeeksforGeeks. https://www.geeksforgeeks.org/logistic-regression-vs-random-forest-classifier/

Jain, A. (2024, February 2). All about Tokenization, Stop words, Stemming and Lemmatization in NLP. *Medium*. https://medium.com/@abhishekjainindore24/all-about-tokenization-stop-words-stemming-and-lemmatization-in-nlp-1620ffaf0f87

Kanade, V. (2024, May 13). *Everything you need to know about logistic regression*. Spiceworks Inc. https://www.spiceworks.com/tech/artificial-intelligence/articles/what-is-logistic-regression/#:~:text=Logistic%20regression%20is%20a%20supervised%20machine%20learning%20algorithm%20that%20accomplishes,1%2C%20or%20true%2Ffalse.

MarkovML. (2023, December 19). *Model Evaluation Metrics: Methods & Approaches*. https://www.markovml.com/blog/model-evaluation-metrics

What is Random Forest? | IBM. (n.d.). https://www.ibm.com/topics/random-forest#:~:text=Random%20forest%20is%20a%20commonly,both%20classification%20and%20regression%20problems.