Text Mining and Natural Language Processing

2022-2023

RICCARDO RICCIO 506483

REVIEW SENTIMENT ANALYSIS

Introduction

The aim of the project is to develop a sentiment analysis model using the Steam reviews dataset to predict a score given the review text. Sentiment analysis is a technique in natural language processing used to classify the emotion that are expressed in a text (positive, negative, neutral).

Data

The Dataset used in this project is the Steam Reviews dataset present on kaggle at:

(link: https://www.kaggle.com/datasets/andrewmvd/steam-reviews)

The Dataset contain reviews and respective sentiment score relative to games on the Steam platform.

Dataset Specifications:

-Columns: There are 5 columns on the original datasets (app_id, app_name, review_text, review_score, review_votes) but only 2 are used:

review_text = Text of the review made by user of the game

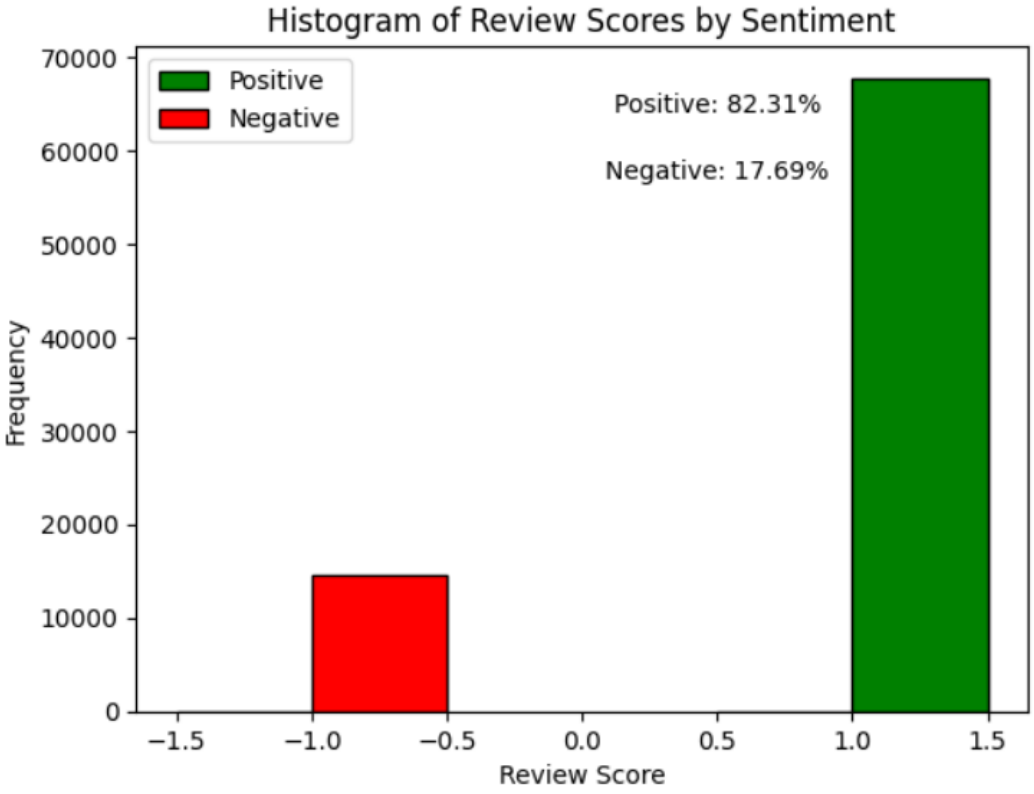
review_score = Sentiment label relative to the review (-1 = game not recommended, 1= game recommended)

-Rows: More than 6 millions sample rows. Only 10.000 were used to train different models and 100.000 used on the final best model.

-Some rows contained missing values or duplicate that I removed since not useful for training the models.

-Number of words: In the sample dataset used for training (8.000 rows) the reviews contained about 25.000 different rows after the preprocessing step

-Imbalance: The dataset contain imbalance classes with 84% of positive label and 16% of negative once.



-Other specifications:

- Reviews contain emoji.

- Swear words are replaced with hearts. Example:

```
-----
♥♥♥♥ing ♥♥♥♥, don't pay $80 for this ♥♥♥♥
```

Methodology

The project is divided into different phases:

- 1) Exploratory Data Analysis
- 2) Text Preprocessing
- 3) Model Selection
    - Standard Machine Learning Algorithms
    - LSTM with GloVe embeddings
    - Pretrained Transformer Model
    - Fine tune pretrained model
- 4) Find best hyperparameter of best model and trained it on more data
- Results

1) Exploratory Data Analysis:

   In this phase I load and analyzed the dataset:

    - Take a sample of 10.000 rows from dataset
    - Check the shape to check if got dimension that I expected
    - shuffle the rows
    - delete duplicate or null rows
    - eliminate columns not useful for the task (maintain only review and score)
    - check structure of review (contain emoji and swear words replace with harts)
    - Visualized classes imbalanced
    - Check most common words relative to each sentiment with Word Cloud

2) Text Preprocessing:

In this phase I create the preprocessing function in order to transform the review text to better be used for training models. I tried different way to preprocessed the data (like not use stop words, use stemming etc.) but the last one use the following phases:

- Lowercase the text
- Replace the hearts with “swear”
- Remove special characters and punctuations
- Remove digits
- Remove emoji
- Remove multiple spaces
- Lemmatize the words

3) Model Selection:

After applying text preprocessing and splitting the data I tried various approach in order to find the best model:

-Machine learning Approach:

I initially determined the best value to use for the Vectorizer step (by training a logistic regression model with different values of Tfidf Vectorizer). Then I tried various algorithms like Logistic Regression, RandomForest, XGBoost and SVM with the Tfidf vectorizer using different sampling technique (in order to handle imbalance classes) and hyperparameter using Random Search.

-LSTM and Embeddings:

I load the GloVe pretrained embeddings and used with the LSTM architecture and trained it with and without SMOTE sampling technique

-Pretrained Model:

Load some pretrained transformer model like RoBERTa and BERT and used them directly to check the performance without training to check the best once that will be fine tuned inn the next phase.

-Fine Tune pretrained model:
Here I load the BERT model, freezed all the layers except the last 2 that I made trainable

in order to adapt better on our dataset. I got problem with the time for training such a model and used only one epochs

4) Find best hyperparameter of best model and trained it on more data:

I the checked the performance of each model during the model selection phases and chose the best one, the Logistic Regression, then I found the best hyperparameters and finally trained and test on a bigger dataset of 100.000 sample. I also tried to check if changing the threshold for the prediction can get better performance

Result

Results and performances of each models:

- Logistic Regression:

```
Model: Logistic Regression Accuracy: 0.853
Model: Logistic Regression F1 Score: 0.9163029037768078

Classification Report:
              precision    recall  f1-score   support

           0       0.75      0.43      0.55       343
           1       0.89      0.97      0.93      1657

    accuracy                           0.88      2000
   macro avg       0.82      0.70      0.74      2000
weighted avg       0.87      0.88      0.86      2000
```

- Random Forest

```
Model: Random Forest Accuracy: 0.8601666666666666
Model: Random Forest F1 Score: 0.9200265599037041

Classification Report:
              precision    recall  f1-score   support

           0       0.72      0.44      0.55       343
           1       0.89      0.96      0.93      1657

    accuracy                           0.88      2000
   macro avg       0.81      0.70      0.74      2000
weighted avg       0.86      0.88      0.86      2000
```

- XGBoost:

```
Model: XGBoost Accuracy: 0.8433333333333334
Model: XGBoost F1 Score: 0.9116051609369595

Classification Report:
              precision    recall  f1-score   support

           0       0.75      0.22      0.33       343
           1       0.86      0.98      0.92      1657

    accuracy                           0.85      2000
   macro avg       0.80      0.60      0.63      2000
weighted avg       0.84      0.85      0.82      2000
```

- SVM

```
Model: SVM Accuracy: 0.859
Model: SVM F1 Score: 0.9190759171578574

Classification Report:
              precision    recall  f1-score   support

           0       0.72      0.44      0.55       343
           1       0.89      0.96      0.93      1657

    accuracy                           0.88      2000
   macro avg       0.81      0.70      0.74      2000
weighted avg       0.86      0.88      0.86      2000
```

- LSTM with GloVe

```
Accuracy: 0.831
F1 Score: 0.7542992900054615
Recall: 0.831
Precision: 0.6905609999999999
Classification Report:
              precision    recall  f1-score   support

           0       0.00      0.00      0.00       169
           1       0.83      1.00      0.91       831

    accuracy                           0.83      1000
   macro avg       0.42      0.50      0.45      1000
weighted avg       0.69      0.83      0.75      1000
```

- BERT:

```
BERT Model
Test Accuracy: 0.828
F1 Score: 0.9059080962800875
Precision: 0.828
Recall: 1.0
```

- BERT WITH ADJUSTED THRESHOLD:

```
BERT Model with Modified Threshold
Test Accuracy: 0.5718
F1 Score: 0.6947968638631503
Precision: 0.8476521739130435
Recall: 0.5886473429951691
Classification Report for BERT Model with Modified Threshold:
              precision    recall  f1-score   support

           0       0.20      0.49      0.28       860
           1       0.85      0.59      0.69      4140

    accuracy                           0.57      5000
   macro avg       0.52      0.54      0.49      5000
weighted avg       0.74      0.57      0.62      5000
```

- FINE TUNED BERT:

```
Test Accuracy: 0.179
F1 Score: 0.01770758554678153
Precision: 0.9487179487179487
Recall: 0.008937198067632851
Classification Report:
              precision    recall  f1-score   support

           0       0.17      1.00      0.29       860
           1       0.95      0.01      0.02      4140

    accuracy                           0.18      5000
   macro avg       0.56      0.50      0.16      5000
weighted avg       0.82      0.18      0.07      5000
```

- TUNED LOGISTIC REGRESSION WITH BIGGER DATA (WITH THRESHOLD 0.3 FOR LABEL 0)

```
Adjusted Classification Report for Logistic Regression:
              precision    recall  f1-score   support

           0       0.65      0.71      0.68      1463
           1       0.94      0.92      0.93      6766

    accuracy                           0.88      8229
   macro avg       0.79      0.81      0.80      8229
weighted avg       0.89      0.88      0.88      8229


Adjusted Accuracy: 0.8799368088467614
Accuracy: 0.8799368088467614
F1 Score: 0.9261805140466228
Precision: 0.9365367180417045
Recall: 0.9160508424475318
```

This data shows that the Machine learning approach got the best results while the LSTM and Transformer model got poor results basically always predicting only one label (1 in general) leading to bad performance in the other. This can be caused by different reasons such as:

1) Class Imbalance

2) Text Preprocessing

3) Need more training (epochs)

Etc. (need more data, need a more complex model …)

For the class imbalance, I tried different sampling approach but the results did not improve (same for the ML models). I also tried different preprocess strategies (training a logistic regression model with different preprocessing functions) but the performance where pretty much the same. For the training I got problems since it required a lot of time and power.

It might be useful (since they are the state of art for sentiment analysis) to check more in-depth the causes of this poor performance for the transformer model making more experiments on the 3 variables that I state before that might create this poor performance.

The final model that got the best performance is a Logistic Regression model (max_iter=500) with no sampling strategy

```
Adjusted Accuracy: 0.8799368088467614
Accuracy: 0.8799368088467614
F1 Score: 0.9261805140466228
Precision: 0.9365367180417045
Recall: 0.9160508424475318
```

Conclusion

In conclusion, I performed a sentiment analysis classification task by analyzing the Steam review dataset, finding insights on the text in order to apply the text preprocessing, and then I tried different approach such as Machine Learning algorithms and transformer model. While some models performed poorly and require further analysis the project's objective was accomplished with the identification of effective models that performed well on the dataset.