Report on Sentiment Analysis Project

Narasimhan Kovalai

December 2022

1 Analysis

The sentiment analysis project was done successfully using the concept of Weighted Logistic Regression and feature extraction.

When we explore the dataset given, we find that 80 percent of the feedbacks were negative and 20 percent were positive. This is a classic example of an unbalanced dataset. Also, the number of feedbacks is 11,675 which is a relatively smaller dataset for a NLP classification task.

Thus the two major challenges were:

- **Small training set**
- **An unbalanced dataset**

The problem of small training set was solved by choosing the simplest among all models-Logistic regression. More complex data such as **Naive Bayes** fit the traning data well but are not able to predict test cases. Thus, to address the problem of **overfitting** (high variance, low bias), logistic regression was used.

The problem of imbalanced data was solved using a slight variance of logistic regression. Weights were assigned to both the classes in roughly inverse ratio of their size while calculating the loss function. In our case, 11 : 55 weight ratio was assigned to negative and positive classes respectively.

A lot of thinking went into deciding the optimised weight ratio, by tweaking it continuously. An attempt to simplify this optimisation process was done in the project using **Grid Search Cross Validation Algorithm** where a 100 fold cross validation was used. This fastens the process of selecting an optimal weight ratio from a given list of weight ratios. The challenge is to however, provide a very thoughtful list of weight ratio for the algorithm to select from.

Porter Stemmer algorithm was used to performing stemming on the feedbacks and NLTK(Natural Language Processing Toolkit) package was used to tokenize the feedbacks and parse them using stopwords as delimiters. We obtained the following results on a test set which was 20 percent of the original data:

- **Accuracy Score**: 0.8016457340840191
- **Confusion Matrix**: [[1469 367] [91 382]]
- **Area Under Curve**: 0.8038599630596895
- **Recall score**: 0.8076109936575053
- **Precision score**: 0.5100133511348465
- **f1 score**: 0.6252045826513911

A thumbrule to note is that in case of unbalanced datasets is as follows:

- **Use precision and recall to focus on small positive class** — When the positive class is smaller and the ability to detect correctly positive samples is our main focus (correct detection of negatives examples is less important to the problem) we should use precision and recall.
- **Use ROC when both classes detection is equally important** — When we want to give equal weight to both classes prediction ability we should look at the ROC curve.
- **Use ROC when the positives are the majority** or switch the labels and use precision and recall-When the positive class is larger we should probably use the ROC metrics because the precision and recall would reflect mostly the ability of prediction of the positive class and not the negative class which will naturally be harder to detect due to the smaller number of samples. If the negative class (the majority in this case) is more important, we can switch the labels and use precision and recall.

  We chose a 11:55 ratio such that all parameters are well optimised. We can choose another weight ratio using GridSearchCV by changing the parameter in it which is required to be optimised.

The confusion matrix for the above weighted logistic regression is:

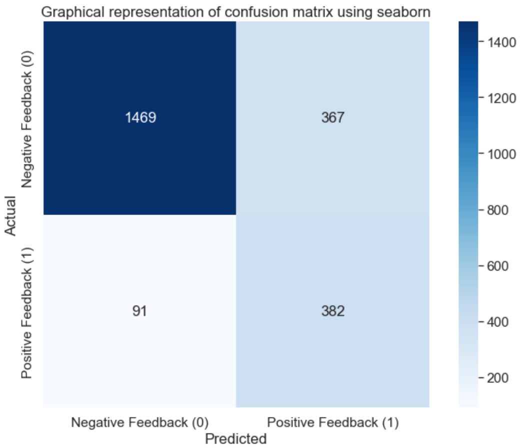

Figure 1: Seaborn confusion matrix plot 11:55 weight ratio.

For the case of unweighted Logistic Regression (1:1) weight ratio, we get the following results and the confusion matrix plot is visualised via seaborn:

- **Accuracy Score**: 0.8549155478562148
- **Confusion Matrix**: [[1778 58] [277 196]]
- **Area Under Curve**: 0.6913929537048552
- **Recall score**: 0.4143763213530655
- **Precision score**: 0.7716535433070866
- **f1 score**: 0.5392022008253095

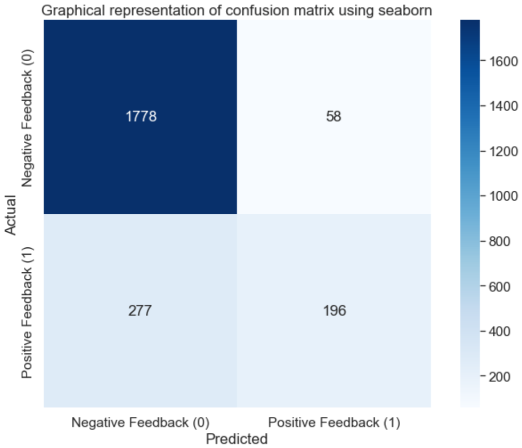

Figure 2: Seaborn confusion matrix plot 1:1 weight ratio.

Clearly, we can see the unweighted classifier is only able to classify the majority class(negative feedback) correctly. But it misclassifies most of the minority class(positive feedack). Also all scores except accuracy score and precision score perform poorly, hence this model cannot be used for our project.Further improvements can be made by using the method of data augmentation if original dataset is free to be modified. A popular algorithm called **SMOTE(Synthetic Minority Over Sampling Technique)** can be used to balance the data, given we are permitted to tweak the original training dataset. Finally, undersampling of majority class or oversampling of minority class can also be done to create a balanced dataset. This project model was built ensuring that original training data is never changed. **THANKYOU**