Amazon Review Helpfulness

Vatsal Patel

Objective

The objective of this project is to build a model which assesses whether a review associated with a product from Amazon is helpful to a customer or not.

Approach

The approach is outlined by having a strong understanding of the user experience when shopping online. Users generally have a decent understanding and expectation associated with the product and want to use the review as a tool to aid them in their decision to purchase (or not) the product. If they end up purchasing or not purchasing a product based on the review, then they can conclude that the review was helpful. Things that I generally keep in mind when identifying whether the product is worth purchasing and the usefulness of reviews are :

- Price
- Overall rating
- Reviews
- Product description, title and summary
- Other items different buyers have purchased
- Other recommended items
- Product rank across it's category
- How helpful other users have found this review

Once these features are identified, we can essentially narrow this down to a binary classification problem. A label of 1 would be given if the review is helpful to the buyer, 0 otherwise. Given these features it is easy to train and assess the accuracy of various models to predict the helpfulness of reviews.

Amazon Dataset & Data Preprocessing

Due to limitations of resources, and the quantity of data available (~50GB - total of review data and metadata associated with each product in 2018 from Amazon), I've chosen to work with a subset of the initial data. The subset which I've chosen encompases video game related products and reviews from Amazon. A thorough description of each of the available columns associated with the data upon download can be found in the README.md file associated with the project. Here I will discuss the new columns added to the dataset which will be additional features for the model.

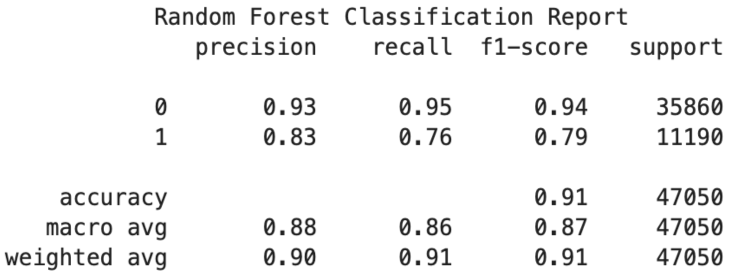
Feature	**Description**
time_since_review	Amount of years since the review was posted
count_also_bought	Count of products also purchased by consumers
image_present	1 if there was an image present associated to the product, 0 otherwise
best_rank	Best rank (minimum) associated to the product across all of its ranks
recommended_item_counts	Count of other products being recommended
similar_item_present	Count of other similar products
description_count	Count of words associated to the description after removing stop words

summary_wc	Count of words associated to the summary after removing stop words
title_wc	Count of words associated to the title after removing stop words
review_wc	Count of words associated to the review after removing stop words

Modelling

The two models I decided to use to solve this problem are logistic regression and random forests. Logistic regression is a strong choice for this task because it converges to any decision boundary which can divide the training sets into positive and negative classes. Random forest, a bagging, ensemble algorithm which is known for its high performance through the use of multiple decision trees to generate a more accurate and stable prediction. They operate on a majority win basis to maximize information gain.

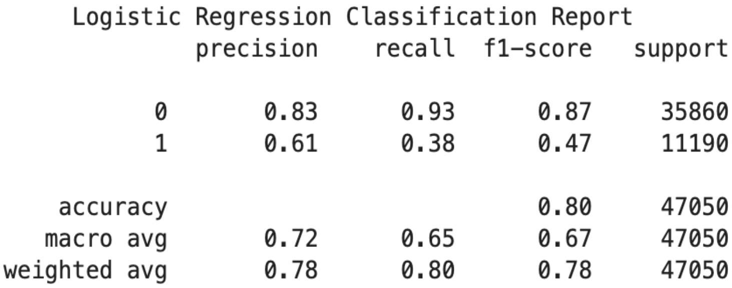
The accuracy of each model was tested across the test and validation sets. The distinction between these two types of datasets is that the validation data is a sample of data which is held back from training your model. It is unbiased and new data which the model has never before seen and helps mimic the performance of the model in a production based environment. The metrics used to assess the accuracy was a classification report which indicates the precision, recall and f1 score. These statistics are useful in assessing the model performance by identifying the ratio of true positives and true negatives with the ratio of false positives and false negatives.



```
 Random Forest Classification Report
    precision    recall  f1-score   support

 0       0.93      0.95      0.94     35860
 1       0.83      0.76      0.79     11190

    accuracy                         0.91     47050
   macro avg       0.88      0.86      0.87     47050
weighted avg       0.90      0.91      0.91     47050
```



```
 Logistic Regression Classification Report
              precision    recall  f1-score   support

           0       0.83      0.93      0.87     35860
           1       0.61      0.38      0.47     11190

    accuracy                           0.80     47050
   macro avg       0.72      0.65      0.67     47050
weighted avg       0.78      0.80      0.78     47050
```

Based on the statistics shown above, it is evident that the random forest model during training substantially out performs the logistic regression model. We can see the model accuracy on the validation set by identifying the true positive predictions (correctly labelled 1 or 0) and dividing it by the sample size. We see that the accuracy associated with the random forest is 90.324 % while the accuracy associated with the logistic regression is 79.772 %. Overall, it is evident that the random forest is the best choice of model for this task.

Discussion

In retrospect, given more time and resources to complete this project there are several areas I would explore and continue to develop. Firstly would be to set up a hadoop / spark server which can handle the data size associated for this project, given this, I can perform analysis and train models on the entire dataset rather than a fraction of it. I would also explore implementation and assessment of various other more complicated models with a neural network framework. Given the size of the data, a deep learning approach to this problem will yield results even better than that given by the random forest. In terms of deployment of the model, I would explore deploying via Flask / Django through AWS Sagemaker with real time predictions. Using Sagemaker will allow deployment at scale to be more systematic and simple, while giving the user the best experience through real time predictions associated with each review.