BUDT758B Final Project

Sentiment Analysis of Amazon Reviews

Wanyun Yang, Mengyuan Ji, Ruoning Che

Introduction

The E-commerce industry has grown tremendously in recent decades as we entered the era of Web 2.0. Major companies in this field, such as Amazon, have significantly increased their size and profit. In this project, the problem we will be investigating is constructing a model to predict the sentiments of Amazon customers' reviews of beauty products. It is a general model that can be used on reviews of any other categories. The model will give us an expected customer sentiment, either positive or negative, after analyzing the assigned customers' reviews.

In the modern world, we face the data overload problem every day in any aspect. Companies are collecting too much customer feedback to handle. It costs too much labor and time to analyze the text content with a low error rate. Therefore, sentiment analysis is needed. It is an exciting field full of challenges.

We use deep learning algorithms to evaluate customers' attitude over the text of product reviews into actionable insights so that it is meaningful for Amazon alike companies to use. We believe building models on a merchandise category will help us better understand customer preferences on this specific topic, such as beauty products. We find out why some customers love the product from positive comments and focus on avoiding negative ones. It will help enhance the customer experience.

The problem with the current method, Random Forest from Machine Learning, is that it only helps us build a mediocre predictive model based on the dataset. Random Forest is the ensemble method of the decision tree. In the ensemble, each decision tree processes the sample and predicts the output label (in the classification). Most times it is not as accurate as deep learning

algorithms like Recurrent Neural Network (RNN), especially when the dataset is big. In our scenario, we want our model to be predictive and do not care much about the features we choose to include in the modeling, so RNN is a better choice.

We will use the RNN, more specifically, the Long Short-Term Memory(LSTM) model, to analyze the Amazon review data. We choose the RNN model to perform the analysis since the RNN model will take on the existing memory information and consider it part of the next cycle's input. The memory can assure that we capture the context and time-series information in the reviews we analyzed. However, there are particular challenges when using RNN models. The major challenge we are facing is to train the model for a satisfactory overall accuracy. RNN based networks are not easy to learn. The total loss would not always be decreasing as more epochs are done. LSTM models can help the vanishing gradient issue in RNN.

Existing Methods

The existing method we chose is one of the traditional machine learning methods - Random Forest. Random forest is indeed an efficient method in the classification of both categorical variables and continuous variables. It helps reduce overfitting and in decision trees, and it also helps with automating missing values. However, the Random forest model also has some significant drawbacks when working with big data sentiment analysis. Due to the ensemble of the decision trees, compared with other machine learning methods, it lacks interpretability on features and could also fail to determine the significance of each variable. Comparably, the RNN model will interpret each variable to the point where it is no longer meaningful for overall performance.

More importantly, LSTM models can capture context information within the sentence by passing on the memory parameters, which is good to have for text-based sentiment analysis.

Random forest models as a supervised machine learning algorithm, consist of a large number of individual decision trees to work as an ensemble. First, we assigned number labels to the reviews corresponding to their sentiment. The positive review is assigned to be 1, and the negative review is assigned to be a number of 0. Then, we dropped any NaN values in the data frame and used the TF/IDF method based on unigram to extract 30 desired features from the text sentiment files and transformed them into a NumPy array format. After feature selection, the actual random forest model with 100 decision trees and “entropy” function as the criterion to measure the quality of splits. The depth of decision trees is set to be 2. Lastly, the 10-fold cross-validation is used to show the prediction accuracy of the model.

Our Methods

Our project intends to build a model that can infer the reviewer's attitude towards the products based on the text. In the reviews, both meanings and the order of words will affect the semantics, so we decided to use Recurrent Neural Network, in particular Long Short-Term Memory, to achieve this goal.

The data set includes each review's text and the overall rating, whose scale is one to five in an integer format. According to rating values, we labeled 1, 2, and 3 as 'negative,' 4 and 5 as 'positive,' the last ten years. Then, we filtered out the useful columns like overall ratings and text reviews. To process the reviews’ text content, we wrote each content cell to a .txt file saved to

positive and negative folders. Then we used loops to concatenate them to a corpus. After we got the corpus, we converted all the letters to lowercase, removed the punctuations, and stopped words first. Had an overview of the length of the reviews so that we can determine an appropriate vector length. Build a word-number mapping dictionary using TF/IDF values. Then we divided the data into training and validation sets by an 8:2 ratio. TF/IDF feature extraction method is used to capture essential features from the text data.

The RNN model has two hidden layers. The first hidden layer contains 256 neurons, and the second layer includes 32 neurons. The output layer has two neurons with softmax as the activation function. For those hyperparameters, we have defined the batch size to be 20. We used the cross-entropy loss function and a total of 10 epochs to check and improve the accuracy of the predictive model.

Data

We got our raw dataset of Amazon reviews in a zipped JSON file from Github. The large dataset's last updated time was 2018, and it offered 233.1 million Amazon reviews from 1996 to 2018. We decided to use a small subset of beauty products because of the limited RAM provided by Google Colab.

Our "All Beauty" dataset contains 5269 records of reviews. In this subset of the data, all users and items have at least 5 reviews, which helps us avoid some potential outliers. The dataset includes the customer rating (from 1 to 5 integers only), the timestamp, ASIN, review ID, product type, and the reviews' text contents.

Result

The model results of random forest exhibit an average validation accuracy of 94% with a standard deviation of 1.2%. The random forest as a black box model performed also quite well as the number record in our dataset is relatively small.

The validation results of the LSTM model achieved an accuracy highest up to 90%, which has outweighed our initial objective, achieving a model accuracy of at least 80%. However, the validation accuracy of each batch doesn't necessarily increase as each batch is performed. The reason is that the error surface of the LSTM model is either very flat or very steep so that the learning rate is difficult to be controlled to reach an exact optimal performance. The model accuracy of 90% means that if we took an unlabelled amazon text review from the beauty department into the LSTM model, 90% of the time, the model would give out a correct sentiment label with either the review being a positive or negative comment, without any additional information provided.

Comparably, using our dataset which has 5261 records of text reviews in total. The baseline methods performed as well as the RNN model. One of the major reasons is that RNN would actually perform better on significantly large datasets. Due to the nature of our hardware limitations, it is relatively difficult to establish an obvious difference to exhibit strong computational power and the advantages of using RNN on sentiment analysis for big data.