HIGH SCHOOL EDITION
Journal of Student Research
Volume 12 Issue 1 (2023)

Sentiment analysis on Amazon Product Reviews

Aman Saridena

ABSTRACT

Sentiment Analysis refers to analyzing text in order to determine the sentiment or opinion that the text is supposed to convey. In this article, the text that is being analyzed are Amazon product reviews taken from Kaggle's Amazon Reviews Dataset, and the predicted sentiment is whether or not the reviewer liked or disliked the product.

Introduction

Supervised learning is a field of computer science which uses the data provided to it to create computational models that can be used to make predictions or help in decision making [1, 2, 3]. This article deals with a subtype of supervised learning, sentimental analysis. The computational model's input are called features, while the output produced by the computational model are called labels. An example consists of a feature and its corresponding labels. In this article, the reviews are the features, and the sentiments are the labels.

Review	Sentiment
The toy broke immediately!	Negative
The paper towel cleaned spills very well.	Positive
The speaker's sound quality was excellent.	Positive
The light blew up.	Negative
The flashlight didn't turn on.	Negative

The dataset that the model uses is a list of examples with features and labels. The dataset is split up into training and validation sets. The split between the two is at a ratio of 4:1. The training set examples are the data that is used to create neural networks, while the validation set are used to compute the error on each neural network and find the model that gives the best fit.

The neural network is formed by graph nodes that are connected by edges. Each edge is weighted with an activation function, which is applied to the input before returning it as the output. The graph's final output is the end result that happens when the input traverses through the entire graph.

Binary Classification Problems

Binary classification problems are a subset of supervised problems where there are only two outcomes for the label, either 0 or 1. In this case, 0 represents a negative review sentiment and 1 represents a positive review sentiment. The output of the neural network is denoted using the symbol $\hat{y}$ ε (0,1) and is a function of the features. The model predicts

that the example falls under the category 1 if $\hat{y} > 0.5$ and under the category 0 if $\hat{y} < 0.5$. Usually in binary classification problems a sigmoid or "softmax" activation function is used, since this function's output will always lie in the range of 0 and 1.

Representing Categorical Variables as Numerical Data

When either the label or features have categorical variables, these variables need to be represented as numerical data. This can be done through one-hot encoding [5], a process where each option for a categorical variable is turned into an individual binary classification feature with only two outcomes either a 0 or 1.

This means that in one-hot encoding, the number of components, k, is set to the number of categorical variable options, which in this case is the number of words. For each word, the index of the vector at that word's corresponding component is set to 1, while all of the other indices are set to 0.

While one-hot encoding allows categorical data to be converted into numerical data, the large number of potential features makes it not viable in many cases, leading to the use of embedding, another method of representing categorical variables.

An example of a categorical variable is a favorite sport. For this example, the options will be baseball, basketball, and football. Sport originally has 3 categorical options, but by using one hot encoding, we can turn each variable into a separate feature.

Example	Sport	Baseball	Basketball	Football
1	Baseball	1	0	0
2	Basketball	0	1	0
3	Football	0	0	1

Binary Cross Entropy Error

Due to the use of a sigmoid activation function in the neural network model, the model's fit was determined through the binary cross entropy error. The formula used to compute the BCE is $BCE = BCE(y, \hat{y}) = y(ln(ln\,\hat{y})) + (1 - y)ln(ln(1 - \hat{y}))$. N is the size of the output vector. Y_i is the i-th value of the output vector and the $\hat{y}_i$ is the predicted output of the model. The loss's output is between 0 and 1. With the loss being closer to 0 the smaller the difference between y and $\hat{y}$ is and 0 if y and $\hat{y}$ are the same.

Tokenizing and Filtering Reviews

The sentence form of the review needs to be converted into numerical data before being used in the neural network. The first part of this is to tokenize the review. Tokenize means to split up the words, while removing punctuation and stopwords, words that don't affect the actual sentiment of the review, since they are found in both negative and positive reviews [6]. For example, a review, such as "This is a great product, and I would buy it again!" would be converted to ["great","product","buy","again"], since "This", "is", "a", "and", "I", "would", "it" are all stopwords.

Converting a Word into an Embedding Vector

After the reviews are tokenized, each word vector will have a different length; however, the neural network needs all of the inputs to have the same length. First, we choose a target length. All of the word vectors with a size greater than this length are trimmed to this length, and the ones with a lower length are padded. When the reviews are padded, blank spaces (“ “) are added to the review until the target length is met.

Now that all of the reviews have been made the same length, each word is turned into a numerical vector. One way to do this is through the use of one-hot encoding as described in the multi classification section. A problem of utilizing this one-hot encoding approach to represent categorical data is that each vectorial representation has too many components to be utilized successfully when dealing with many possible choices. In order to lower the number of components, while minimizing the error of the model, a process called embedding is utilized.

In embedding, the number of components that the vector representation that each word has, k, is chosen beforehand. For each word, a random vector with k components is created. From here, the neural network optimizes the vector for each word to minimize the model’s error.

After converting each word to a vector-based representation, each individual review becomes represented as a 1d array storing 1d vectors or a 2d array containing numerical data. The shape of this array is (s,k), where s is the number of unique words found in the sample review set and k is the number of components.

Neural Network Model

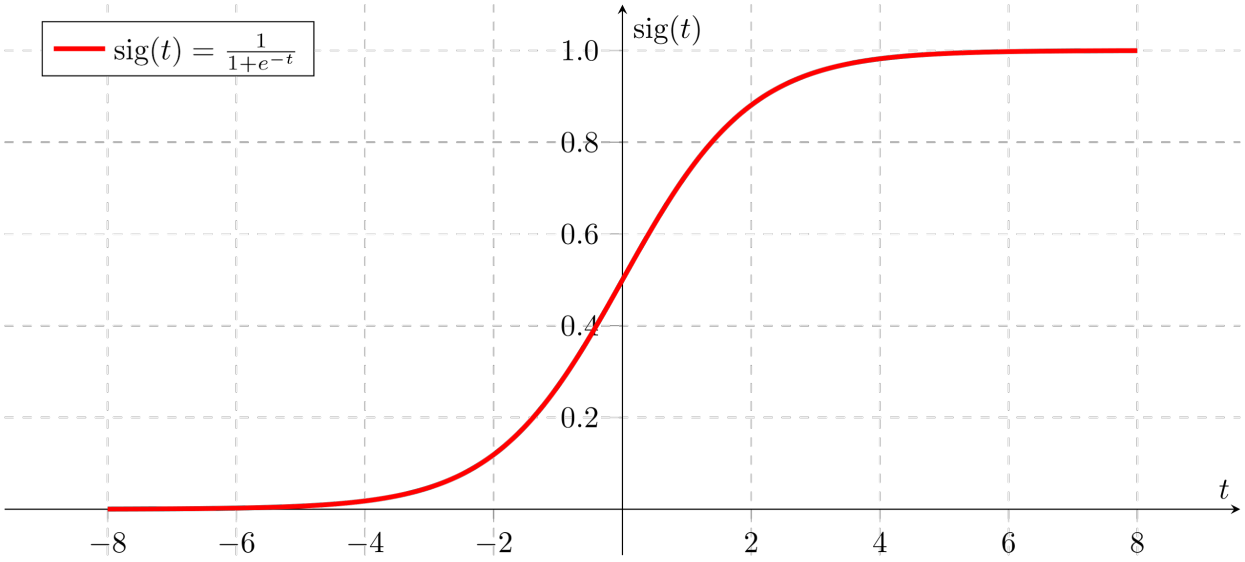
This neural network is made up of multiple node layers. Each layer consists of a group of nodes that all have activation functions, which rely on hyper parameters. A hyperparameter is a parameter in the neural network that affects the model that is trained. In this problem, one hyperparameter is the size of the embedding matrix that is created by the embedding layer. The larger the hyperparameter is, the more accurate the model is on the training set. However, once the hyperparameter is too large, there is a risk of overfitting, where the model becomes accustomed to the specific data points. This will lead to it incorporating the noise or variation in the specific dataset and producing larger binary cross entropy errors on other features. Different hyperparameters are chosen in this article to determine what optimizes the binary cross entropy error.

The first layer in this neural network is the embedding layer, which is the layer that converts each review into a 2d embedding matrix where each word is stored as a 1-dimensional vector of numerical vectors.

The second layer is the dropout layer. The dropout layer takes in a parameter called the rate. This rate is between 0 and 1. This layer randomly chooses input units and sets them to 0, while keeping the total sum of all of the layers in the data the same by incrementing each non-zero element by 1/(1-rate). The dropout layer reduces overfitting that occurs in the data.

The second layer is a recurrent layer, or a function that takes in as input the embedding matrix of shape (s,k) before returning a 1 dimensional tensor as the output of the function.

The third layer that is used is a dense layer which uses a sigmoid activation function: $\sigma(x) = \frac{1}{1+e^{-x}}$.



Sigmoid activation functions return a number that lies in the range between 0 and 1. In binary classification problems, this final output can be utilized to find the prediction by rounding the number to the nearest integer, producing a final output, $\hat{y}$, that is either 0 or 1.

To find out the hyperparameters that allow the model to give the most accurate predictions, we create multiple models before fitting them to the training set. For each fitted model, we choose the model that minimizes the binary cross entropy error on the validation set. The reason that we split the examples between the training and validation sets, is that if the same examples that are used to train the model are used to validate it, that will result in overfitting.

Results

In this neural network, a hyperparameter, k, is the number of components in the embedding matrix output by the embedding layer. In this article, the values that I tried for k were 30, 50, 100 and 200.

When k is set to 30, the model on the training set had a loss of 0.5222 and an accuracy of 0.7493. When the model made predictions on the validation set, there was a loss of 0.2956 and had an accuracy of 0.8812.

With k having a value of 50, the model that is created has a loss of .4689 and an accuracy of .7493 on the training set. The results of the model on the validation set were a loss of 0.2908 and an accuracy of 0.8818.

When k is set to 100, the results of the model that was created had a loss of .4689 and an accuracy of .7644 on the training set examples. The model's results on the validation set were a loss of 0.2753 and an accuracy of 0.8866.

For a k value of 200, the results of the model on the training set were a loss of .4539 and an accuracy of .7747. The results on the validation set were a loss of 0.2739 and an accuracy of 0.8866.

References

[1] Andriy Burkov. The hundred-page machine learning book, volume 1. Andriy Burkov Canada, 2019.

[2] Sebastian Raschka. Python machine learning. Packt publishing ltd, 2015.

[3] Cha Zhang and Yunqian Ma. Ensemble machine learning: methods and applications. Springer, 2012.

[4] Team, Keras. “Keras Documentation: Keras API Reference.” *Keras*, https://keras.io/api/.

[5] “ML: One Hot Encoding to Treat Categorical Data Parameters.” *GeeksforGeeks*, 21 June 2022, https://www.geeksforgeeks.org/ml-one-hot-encoding-of-datasets-in-python/.

[6] Perry, Tal. “What Is Tokenization in Natural Language Processing (NLP)?” *Machine Learning Plus*, 16 May 2022, https://www.machinelearningplus.com/nlp/what-is-tokenization-in-natural-language-processing/.