Emily Chiao - emchiao
Kayson Hansen - kayson

Tripadvisor Las Vegas Hotel Reviews - Neural Network Sentiment Analysis

Abstract:

With the recent improvement of large language models, such as GPT-3 and the newly released GPT-4, natural language processing has been receiving an increasing amount of attention. One approach to natural language processing is sentiment analysis, which identifies the emotional tone behind a body of text. Sentiment analysis is a common way for companies, such as Tripadvisor, to determine and categorize opinions about a product, service, or experience. However, the specific language used in a given review varies drastically from user to user, and it is often difficult to determine the overall sentiment of a user's complex review. In order to better understand the type of language people use in reviews, we have developed a neural network model to classify Tripadvisor hotel reviews to investigate whether the overall sentiment of a user's review can be accurately predicted. Initial experimental results demonstrate that neural network models can predict, with extremely high accuracy (~90%), whether reviews contain positive or negative sentiment. They can predict, with fairly high accuracy (~60%), the number of stars reviews were given on a 5 star rating system.

Introduction:

Language contains a vast number of intricacies and complexities that can be misunderstood by both humans and artificial intelligence alike. Our motivation was to better understand the type of language people use when describing their opinions on experiences and investigate whether reviews for casinos are generally positive or negative. We built both a binary classification neural network model and a multi-class classification neural network model to perform sentiment analysis on TripAdvisor hotel reviews in Las Vegas. The inputs to our algorithm were paragraph embedding vectors generated from the hotel reviews using doc2vec. The outputs are either a positive (1) or negative (0) sentiment for the binary classification model or a prediction of how many stars (1-5) for the multi-class classification model. We were able to achieve 90.28% test accuracy for the neural network binary output and 60.47% test accuracy for the neural network multi-class output.

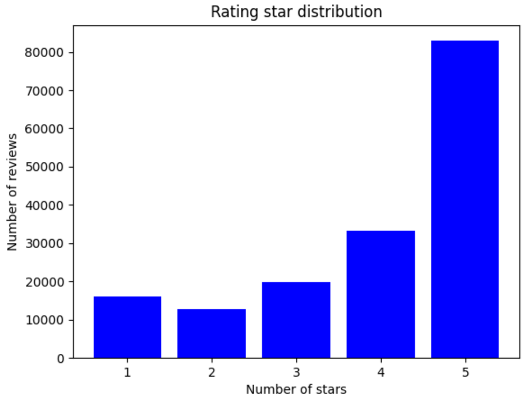
Related Work:

We chose three papers that each used different machine learning techniques for sentiment analysis. The first paper, authored by Bhumika Jadav and Vimalkumar Vaghela, discussed sentiment analysis on movie reviews and tweets using support vector machines (SVMs). Jadav and Vaghela found that using SVM models, they could achieve an accuracy of about 74.5% on movie review sentiment analysis and an accuracy of about 78% on tweet sentiment analysis. The second paper, authored by Lopamudra Dey, et al., discussed sentiment analysis on movie reviews and hotel reviews using Naive Bayes and K-nearest neighbors (KNN) models. Dey et al. found that the sentiment of movie reviews could be predicted with an accuracy of approximately 81% using Naive Bayes models and an accuracy of 71% using KNN models. They also found that the sentiment of hotel reviews could be predicted with an accuracy of approximately 55% using Naive Bayes models and 52% using KNN models. Finally, the third paper, authored by Ilham Tiffani, discusses the optimization of Naive Bayes models for sentiment analysis using unigram, bigram, and trigram features. Tiffani found that using unigram features, an accuracy of 81.30% for sentiment analysis on hotel reviews could be achieved, compared to 71.60% accuracy using bigram features and 71.90% accuracy using trigram features.

Each of the aforementioned papers differed from our work in the machine learning models they built—surprisingly, no other researchers chose to use neural networks or logistic regression to classify text sentiment. Instead, they either used support vector machines, naive Bayes, K-nearest neighbors, or naive Bayes with unigram, bigram, or trigram features. Almost all the other researchers performed similar data preprocessing as we did: they removed stop words, removed unnecessary punctuation, performed stemming, etc. Ultimately, our models were more accurate than any of the other researchers (~90% accuracy), which seems to suggest that either neural networks are more effective at sentiment analysis than support vector machines, naive Bayes, or K-nearest neighbors models, or that our dataset was more comprehensive than the datasets used by the other researchers.

Dataset:

First, we scraped Tripadvisor in order to obtain our datasets. We hypothesized that Las Vegas hotels and casinos would have a high volume of reviews for each hotel, so we chose 15 Las Vegas hotels to scrape for reviews. The number of reviews for these hotels ranged from approximately 10,000 to 30,000. In total, we had approximately 150,000 reviews. We conducted stratified sampling with our data to obtain a training set with 80% of the data, a cross-validation set with 10%, and a test set with 10%, so we had around 120,000 training examples, 15,000 cross-validation examples, and 15,000 test set examples. Each set included reviews with various ratings from various hotels, so we would not have, for example, all of the test set reviews being a 5-star rating from the same hotel. We did not perform data augmentation because we scraped a sufficient number of reviews. We performed analysis on our data by finding the total number of reviews of each star, and we computed the mean and standard deviation of the ratings. There were 16,109 1-star reviews, 12,720 2-star reviews, 19,906 3-star reviews, 33,156 4-star reviews, and 82,908 5-star reviews. The average review was 3.93 stars. The standard deviation was 1.34 stars, indicating that the reviews were relatively closely distributed.



Given that a large proportion of the ratings are 4 or 5 stars and thus classified as positive sentiment, it should be noted that a classifier that classified *every* review as having positive sentiment would have an accuracy of 80.1%. This gives us a good baseline to compare our model's performance to.

The features for our data were the entries in the 50-dimensional vectors generated by doc2vec. doc2vec takes in documents as inputs (in our case, the Tripadvisor hotel reviews) and outputs vectors that attempt to capture the theme or overall meaning of the documents. We did not perform data normalization because doc2vec effectively performed data normalization by generating embedding vectors. These features were appropriate for our task because they allowed us to encapsulate the sentiment of our reviews in a numerical format that we could then use to train our models.

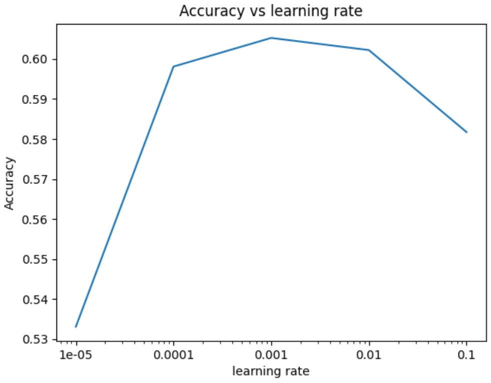
Methods:

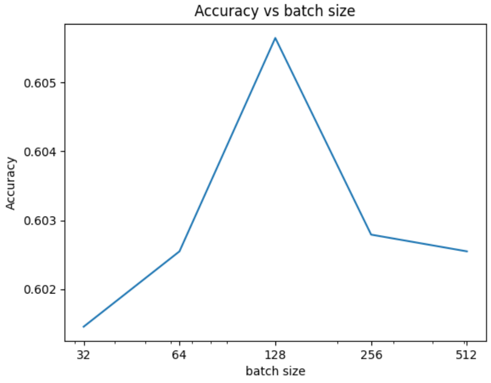
We built binary and multi-class classification neural network models for sentiment analysis. In both cases, we used a 4-layer neural network with the Relu activation function, which is $max(0, X)$, for the 3 hidden layers. The first hidden layer had 48 units, the second had 20 units, and the third had 10 units. The output layer used a sigmoid activation function to generate probabilities for classifying the sentiment of the reviews for the binary output, and a softmax activation function for the multi-class output. The sigmoid activation function is $f_{w,b}(x^{(i)}) = \frac{1}{1+e^{-z}}$ where $z = w * x^{(i)} + b$, and $\sigma(z)_i = e^{z_i} / \sum_{j=1}^{K} e^{z_j}$ is the softmax activation function. The neural network learned the optimal weights and biases in both cases using the cross entropy loss function, given by $J =- \sum_{i=1}^{m} log(\hat{y}_i)$. The neural network algorithm works by first taking in a vector of inputs, then taking the dot product and adding a bias to obtain a z value. We then apply the Relu activation function to the output, and pass that in as the input to the next layer. We repeat this process until the output layer, where we apply either the sigmoid activation function to the z value to obtain a probability between 0 and 1, or we apply the softmax activation function to the z value to obtain a probability between 0 and 1 for each class. To learn the optimal weights and biases for each layer, we apply mini-batch gradient descent, taking the gradient of the loss function (cross-entropy loss), then using back-propagation to find the gradient of the error with respect to each weight and bias (we repeatedly apply the chain rule to find these gradients). We then move our weights and biases in the opposite direction as the gradient at each step.

Additionally, we built logistic regression models to compare our neural network models to, in order to establish a baseline level of accuracy. We built both a binary and multi-class classification logistic regression model, using the sigmoid and softmax activation functions, respectively. The logistic regression algorithm works similarly to the neural network algorithm, just without the hidden layers. We first take in a vector of inputs, then take the dot product and add a bias to obtain a z value. We then apply the sigmoid activation function to the z value to obtain a probability between 0 and 1, or we apply the softmax activation function to the z value to obtain a probability between 0 and 1 for each class. To learn the optimal weights and biases, we apply mini-batch gradient descent, taking the gradient of the loss function (cross-entropy loss) and moving our weights and biases in the opposite direction as the gradient at each step.

Experiments/Results/Discussion:

We optimized our hyperparameters by training our neural network models repeatedly with various learning rates and batch sizes. We didn't include the regularization constant, lambda, as a hyperparameter, because our cross-validation error was never significantly higher than our train set error. The figures below show the accuracy of our multi-class neural network model using various learning rates and batch sizes, and from the graphs, it's apparent that the optimal learning rate was 0.001, and the optimal batch size was 128. In both cases, we used our cross-validation set to fine-tune the hyperparameters.





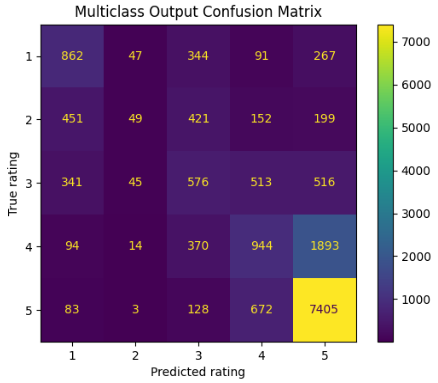
We performed error analysis by manually looking through examples in the cv set and categorizing them by common traits. We looked through all of the misclassified reviews in the first 200 reviews. One thing we noticed was that some negative reviews that used very positive words, like “great” and “prestige,” were misclassified due to the model not understanding sarcasm. Additionally, some misclassifications were not necessarily errors on behalf of our model – some reviews included a mixed bag of positive and negative sentiments, and there were cases where users marked an arbitrary star value that did not align with their written review.

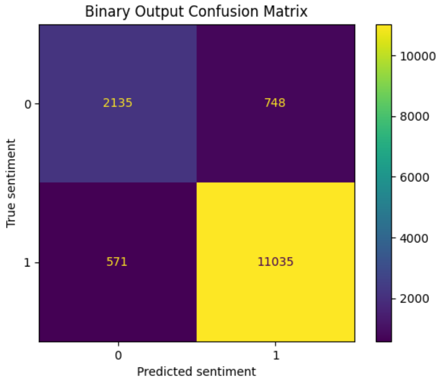
We conducted stratified sampling with our data to obtain a train set with 80% of the data, a cross-validation set with 10%, and a test set with 10%. The table below represents the error for each of our datasets on the models that we built. We found the cross-validation error to be similar to the train error, indicating that our models had low variance; therefore, we didn’t overfit to our dataset.

Model	Train Error	CV Error	Test Error
Neural network (multi-class)	39.21%	39.77%	39.53%
Logistic regression (multi-class)	40.80%	41.13%	40.55%
Neural network (binary)	8.87%	9.21%	9.72%
Logistic regression (binary)	11.07%	11.32%	10.96%

Overall, our results were closely in line with what we expected. Our primary metric to measure the performance of our models was accuracy, that is, the fraction of correctly predicted inputs over the total number of inputs. We achieved an error rate of only 9.72% when we classified the sentiment of reviews as positive or negative, and we achieved a larger, but still, reasonable, error rate of 39.53% when we classified reviews as a certain number of stars. In both cases, our neural network models outperformed the corresponding logistic regression models—which we used as baseline models to compare our performance to—with error rates around 1% lower.

The confusion matrices below show our model’s performance on each type of review. Our model with binary outputs had an error rate of 4.92% on positive sentiment reviews and an error rate of 25.95% on negative sentiment reviews. We attribute this disparity to there being many more positive reviews than negative reviews on Tripadvisor. For the model with multiclass outputs, the most obvious problems are misclassifying 57.10% of the 4-star reviews as 5-star and misclassifying 35.46% of the 2-star reviews as 1-star and 33.10%.





Conclusion/Future Work:

In conclusion, the neural network model for binary classification was the highest performing model (90.28% test accuracy), and the neural network model for multi-class classification was quite effective (60.47% test accuracy). In both cases, the neural network models outperformed the corresponding logistic regression models—which we used as baseline models to compare our performance to—with error rates around 1% lower. Our hypothesis for this disparity is that neural networks are more flexible, so they may be able to learn more complex patterns within the data.

If we had another 6 months to work on this, we would first get more data. We would scrape reviews from hotels in other locations besides Las Vegas and also look into scraping reviews in other categories on Tripadvisor besides hotels (e.g. restaurants). This would allow us to get a broader view of online reviews and how well the sentiment of reviews can be predicted based on text. Secondly, we would try training different types of machine learning models: decision trees, k-means clustering, etc. In our limited time for this project, we were only able to train neural network and logistic regression models, so it would be beneficial to investigate the effectiveness of other types of models.

References

B. Jadav and V. Vaghela. “Sentiment Analysis using Support Vector Machine based on Feature Selection and Semantic Analysis.” *International Journal of Computer Applications*, Volume 146, no. 13, July 2016.

“Doc2vec paragraph embeddings.” *Gensim*, https://radimrehurek.com/gensim/models/doc2vec.html.

I. Tiffani. “Optimization of Naïve Bayes Classifier By Implemented Unigram, Bigram, Trigram for Sentiment Analysis of Hotel Review.” *Journal of Soft Computing Exploration*, Vol. 1, no. 1, Sept. 2020.

L. Dey, S. Chakraborty, A. Biswas, B. Bose, and S. Tiwari. “Sentiment Analysis of Review Datasets using Naïve Bayes’ and K-NN Classifier.” *I.J. Information Engineering and Electronic Business*, 4, 54-62, 2016.

“The 10 Best Hotels in Las Vegas, NV for 2023.” *Tripadvisor*, https://www.tripadvisor.com/Hotels-g45963-Las_Vegas_Nevada-Hotels.html. [1]

[1] We generated our datasets ourselves through web scraping; the website cited was where we scraped.