CSC 180-01 Intelligent Systems

Mini-Project 1: Yelp Business Rating Prediction using Tensorflow

Due at 2:00 pm, Friday, October 2, 2020

Tran Ngoc Bao Huynh (Student ID: 219763298)

Ong Thao (Student ID: 219467431)

Problem Statement

Our goal is to predict a business’s star rating based on all of Yelp’s review text using Tensorflow to determine the best neural network implementation for evaluating a business’s star rating.

Methodology

For this project, we first started with preprocessing Yelp’s business and review datasets by filtering out businesses that had less than 20 reviews and those that were not open. We then went about creating three different dataframes: businesses with all information, businesses with no review count column, and businesses related to coffee. After creating the three business dataframes, we then moved onto preprocessing the review datasets and also creating a dataframe to store all of the reviews. We then performed data aggregation to group all reviews for a specific business together, and merged the business and aggregated review dataframes into one dataframe on “business_id”. After obtaining our final dataframe, we saved it into a CSV file, so that next time instead of running preprocessing again for both business and review json datasets, we can just load the CSV file that contains our final, preprocessed data to save time.

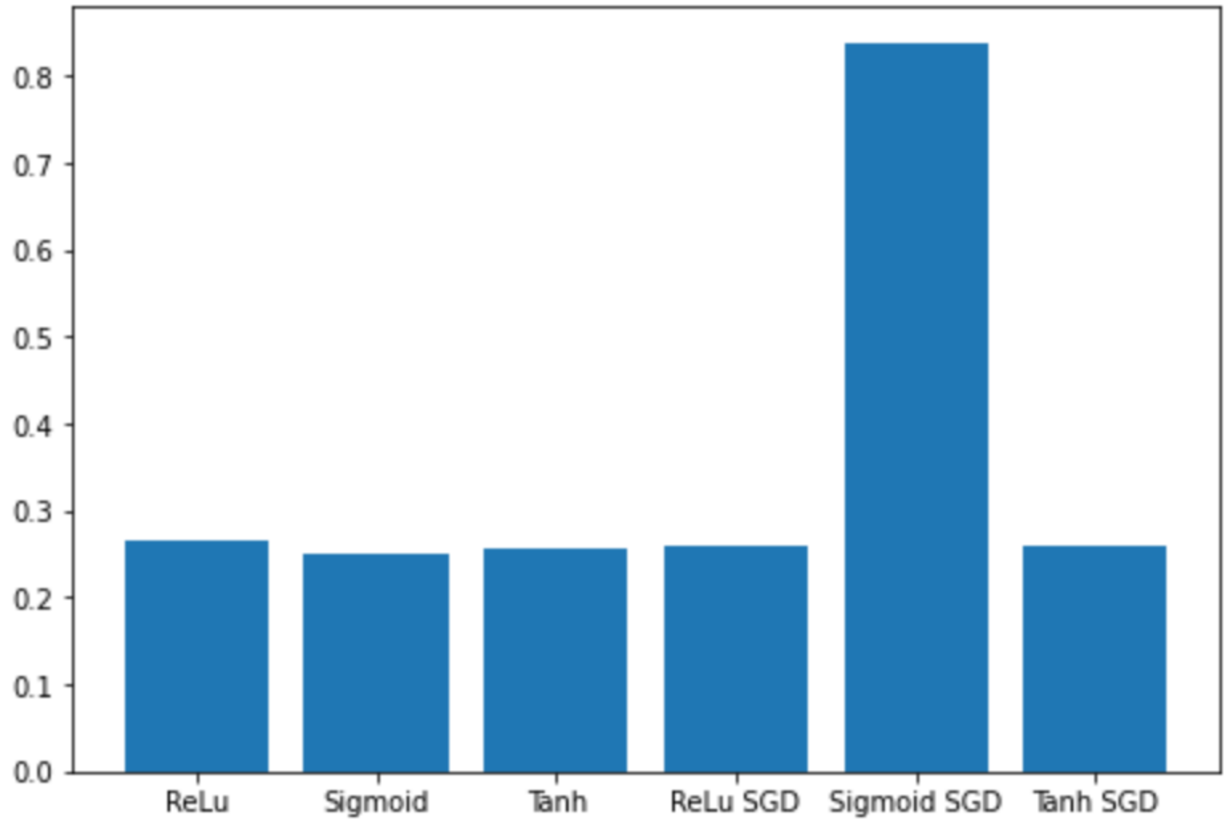
After preprocessing, we then used tfidVectorizer to obtain the TFIDF representation of each business and converted the ouput into numpy arrays. After this, we then converted the numpy arrays into dataframes so we can split the data into testing and training data using sklearn. After splitting the data into testing and training data, we moved onto testing the six neural network implementations using 5000 neurons on the input layer and six layers (4 hidden layers with 40, 20, 15 and 5 neurons on each layer; respectively).

For our case, because we were dealing with three different dataframes, we tested all of the neural network models on the dataframe with all the reviews first. After running each model implementation three times (with EarlyStopping incorporated in each run to stop the training when the model stops improving), we saved the model that had the best weights. This is so that we can use it later to train our other two dataframes and determine how more focused data and less columns affects the training model's RSME score and prediction. We also compared the prediction and RSME scores of all models using a bar plot and lift chart.

Experimental Results and Analysis

We tuned different numbers of layers (4-8 layers with 2-6 hidden layers), and neurons N1(50-25-15-10), N2(40-20-15-5), N3(150-75-45-30), N4[25,10], N5[50,25], etc.. 4 hiddens layers with N2 neurons set returned the best result so we will use that to implement the models. (Because of limited ram issues, we could not have all of the experience run at the same time.)

From the bar plot below, we can see that Sigmoid Adam was our best model while Sigmoid SGD was the worse one for our case. Additionally, the other model's RSMEs were fairly similar.



We use the best model (Sigmoid - Adam) to predict the rating and then choose randomly 5 arbitrary businesses from the test data to compare the result:

	name	categories	star_rating	Predictior
2	b'Impressions Dental'	Cosmetic Dentists, Dentists, Oral Surgeons, He...	5.0	4.807671

10	b'Ted Wiens Tire & Auto'	Oil Change Stations, Tires, Automotive, Auto R...	4.5	4.398614
27	b'London Pub Company'	Nightlife, Bars, Pubs	4.0	3.926776
235	b'Sonic Drive-In'	Ice Cream & Frozen Yogurt, Food, Fast Food, Bu...	3.0	3.023425
1100	b'Nothing Bundt Cakes'	Bakeries, Food, Cupcakes, Desserts	3.5	3.517546

In the extra part, we tried adding more layers and neurons (we aimed for ⅔ of the total input + output size). Although it took longer to run, the RMSE score is pretty good (about 0.256)

Task Division and Project Reflection

Our group implemented all phases of the project together and the work was evenly divided between the members. Our biggest challenge in this project was not having enough RAM on our machines to load the Yelp datasets and do data preprocessing. Because of our computer limitations, we were forced to use Google Colab and hoped that it provided us with enough resources to run our Python code snippets. Additionally, another challenge that we faced was the fact that each cell code took super long to execute. This elongated the development process and hindered us from moving onto the next step as we had no way of knowing if our data is formatted or running correctly. However, our group did take full advantage of the long execution times by researching tensorflow, reviewing labs, and generating our reports. We read about and plan to try to implement cross validation and ensemble methods to see if it could improve the accuracy and performance. However, we could not make it this time.

Despite the setbacks, we did learn lots from this project as a team. One thing being that we should start on future projects earlier, so we can have more time to prepare our report and presentation. Also, if we started earlier, we would have had more time to fine-tune our models and produce better results. However, regardless, we were able to learn lots about neural networks and the different types of models you can develop using Tensorflow.

Additional Comments

Both Tran and I each had our own project implementation then we met up with each other over Zoom to go over our individual implementations and combined our work together. We had similar implementations, but our results at the end differed a lot (especially the SGD output for each activation function). After some discussion, we saw that Tran's output for the neural networks is actually more accurate, so please reference her regression graphs and visuals.