* What is the problem?

Star Ratings do not always match the sentiment of the reviews. This creates noise in understanding a business you may be interested in. To put it analogously -- People say it’s not what was said but the tone/sentiment that has the meaning.

Here the star rating is what was said, and the tone is the words in the review. This is a classification style problem. The goal is to see if machines can classify a star score better than a human can, based on human-generated text reviews.

* How is my solution valuable?

My solution will add value by creating a convenient representation of how people may feel about the business. This can be used as a comparative score to the average rating. And for some, they may find it as a better-tuned measurement for deciding about offering their money to a business.

* What is my data source, and how did I access it?

It is the yelp dataset found at - https://www.yelp.com/dataset/documentation/main.

It is large and offers a large variety of data.

The dataset is quite large, it covers 8.8 million reviews from 2 million users about 160,000 businesses. Available features are stars given, avg stars, review text, tip text, user/biz/review ids, business info, etc.

* What techniques from the program do I plan to use?

I will use multiple EDA techniques to clean and sample the data initially.

Next, I will create features to better mathematically map individual reviews to avg. star ratings. Then I will tokenize, lemmatize, and model reviews into different algorithms to get a sense of star prediction. To do this I will use NLTK tokenizer and lemmatizers. I will then use Bag of words and Tfidf Vectorization methods to preprocess text data.

I will then model with a Random Forest Classifier and a 2 input, one output neural network. The text input will go through an LSTM layer. And the numeric inputs will be passed through a sequential 3 dense layer input. The layers will be concatenated and driven to an output prediction.

* What is the biggest challenge I expect to face?

The dataset is humongous. I have limited computing resources and time. The goal is to find promising results on a smaller training set to then dive deeper in the future.

I will need to create features that will help the model decide but not leak info. As the dataset is resource-intensive, I need to sample and choose input features efficiently.

* What is the most important pain point I can answer?

There is so much data today that the old model of yelp can feel overwhelming. Finding more trustworthy ways to find a business you will like in a faster time is what I want to help solve with this project. Customers of yelp and untried benefits will benefit from more regularized star scoring as this program can potentially find anomaly scores and delete them from the average.

Another use case is the program can analyze text as a review is being written and recommend a well-fitted star score.