Capstone Project 2: Report

Questions:

Define the problem:

- Sentiment analysis of the Yelp customer review
- Prediction of the star rating based on the review
- Recurring Themes
- Try to Tag/Entities with sentiment

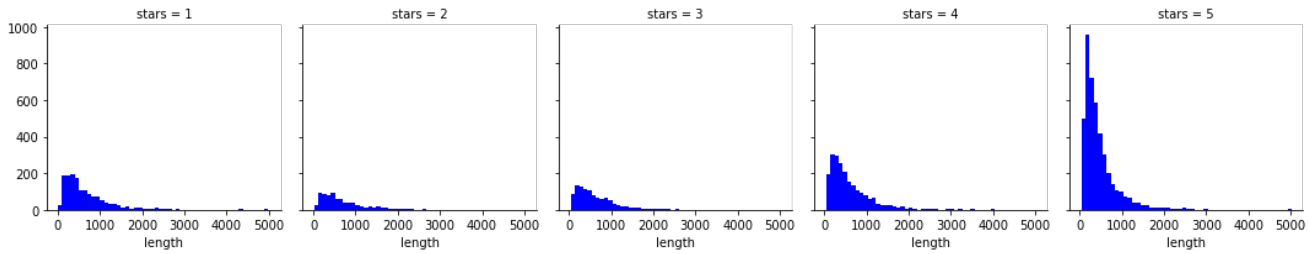
Identify your client:
Client could be all the e-commerce companies starting with Yelp

Describe your data set, and how you cleaned/wrangled it:

- The dataset was directly downloaded from https://www.yelp.com/dataset
- Several columns were renamed to make them readable
- column “review_id” was made the index
- Missing data was searched but their were no missing data
- A separate new column ‘length’ was made which counts and stores the length of the reviews to be used for further analysis

Initial findings:

- Graph was plotted with number of review vs length for all the 5 stars rating separately.



As per the above graph we can see that the length of the reviews were more for the 1,4 and 5 stars. We can also deduce that for stronger feeling the length was more.

- We also observed that there is negative correlation between:

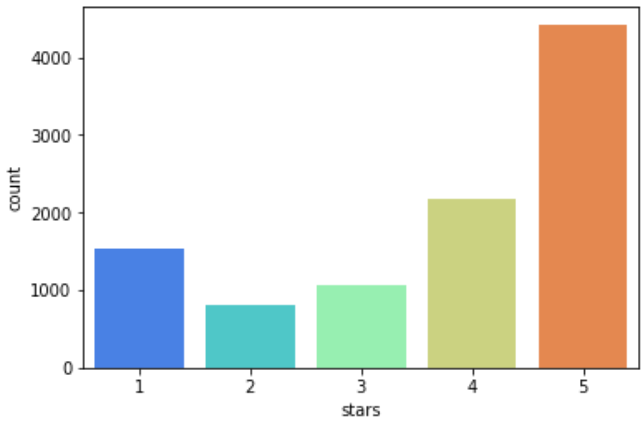
Cool and Useful
Cool and Funny
Cool and Length

- Thus, we can say that the reviews marked cool tend to be curt, not very useful to others and short. Whereas, there is a positive correlation between:

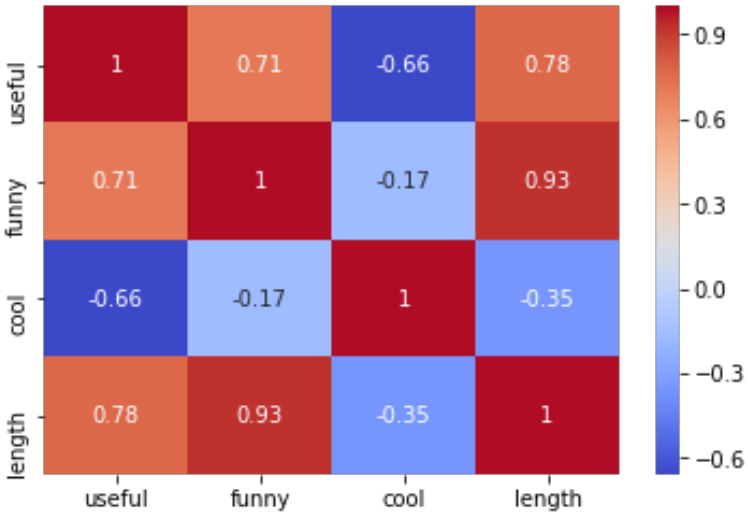
Funny and Useful
Funny and Length
Useful and Length

- Thus, we can say that longer reviews tend to be funny and more useful.

- We plotted graph between number of review v/s stars and observed that our dataset is dominated by 5 star reviews.

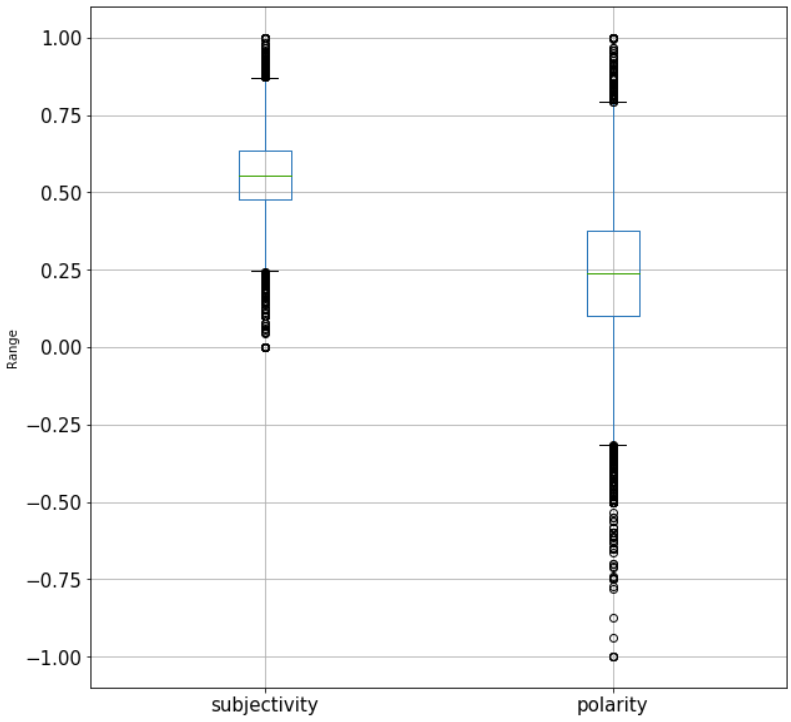


- We plotted heat map between the steal, and saw the following observation:



- Their are total of 4618 unique businesses reviewed in the above dataset of 10,000.

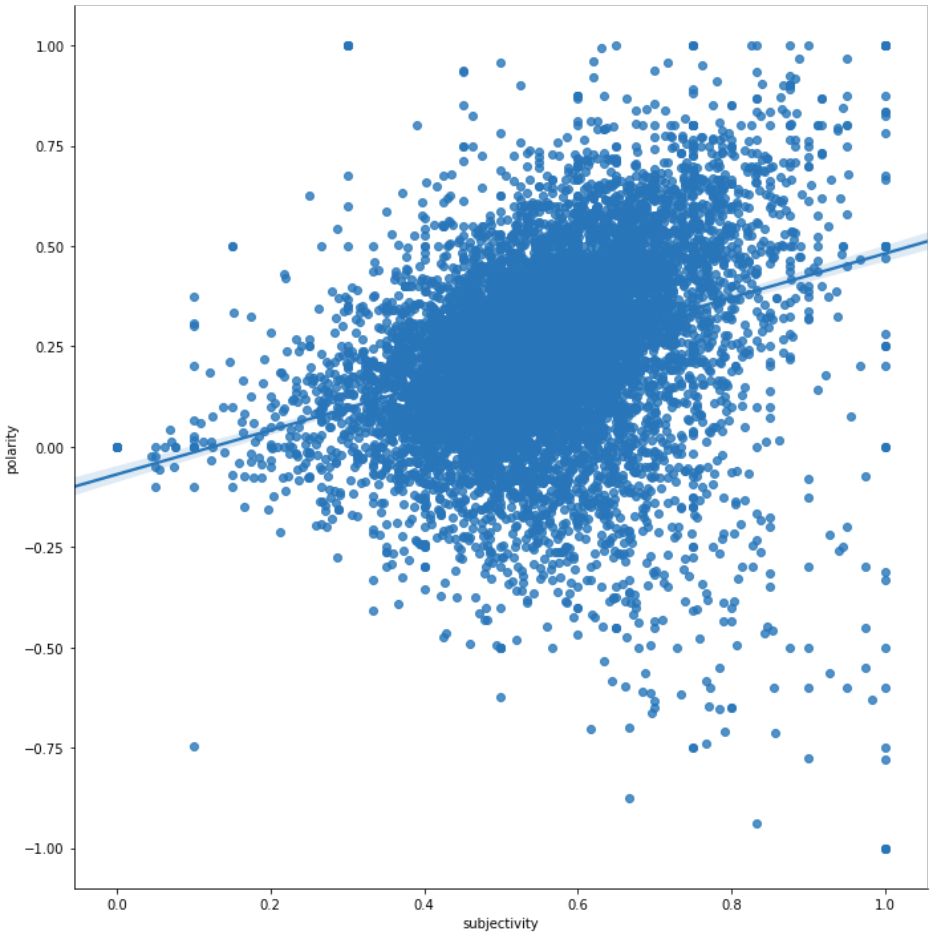
- We performed first type of Sentiment analysis using **textblob**. We checked for the polarity and subjectivity of the text reviews. Polarity — It simply means emotions expressed in a sentence, across a range of negative, to positive.
- Subjectivity — Subjective sentence expresses some personal feelings, views, or beliefs.
- So my program has confirmed to me that all the 10000 records are there and gave me a mean polarity of 0.24, which is good that means as an average, most people are in between neutral to positive with the services. And as you can see the 50% Value which means the median is above zero i.e., 0.24.



- The covariance between the two variables is 0.0287693. We can see that it is positive, suggesting the variables change in the same direction as we expect.

- We can see that the two variables(polarity and subjectivity) are positively correlated and that the correlation is 0.69351. This suggests a high level of correlation, e.g. a value above 0.5 and close to 1.0.

- The following plot shows a positive correlation between Subjectivity and Polarity. Meaning, as subjectivity increase, the polarity in the response increase too, Or in other words, the more strong feelings are expressed, the more the overall comment is subjective.



- The scatter diagram is used to find the covariance and correlation between two variables. This diagram helps you determine how closely the two variables are related. After determining the correlation between the variables, you can then predict the behavior of the dependent variable based on the measure of the independent variable.

- The covariance between the two variables is 0.0287693. We can see that it is positive, suggesting the variables change in the same direction as we expect.

- We can see that the two variables are positively correlated and that the correlation is 0.69351. This suggests a high level of correlation, e.g. a value above 0.5 and close to 1.0.

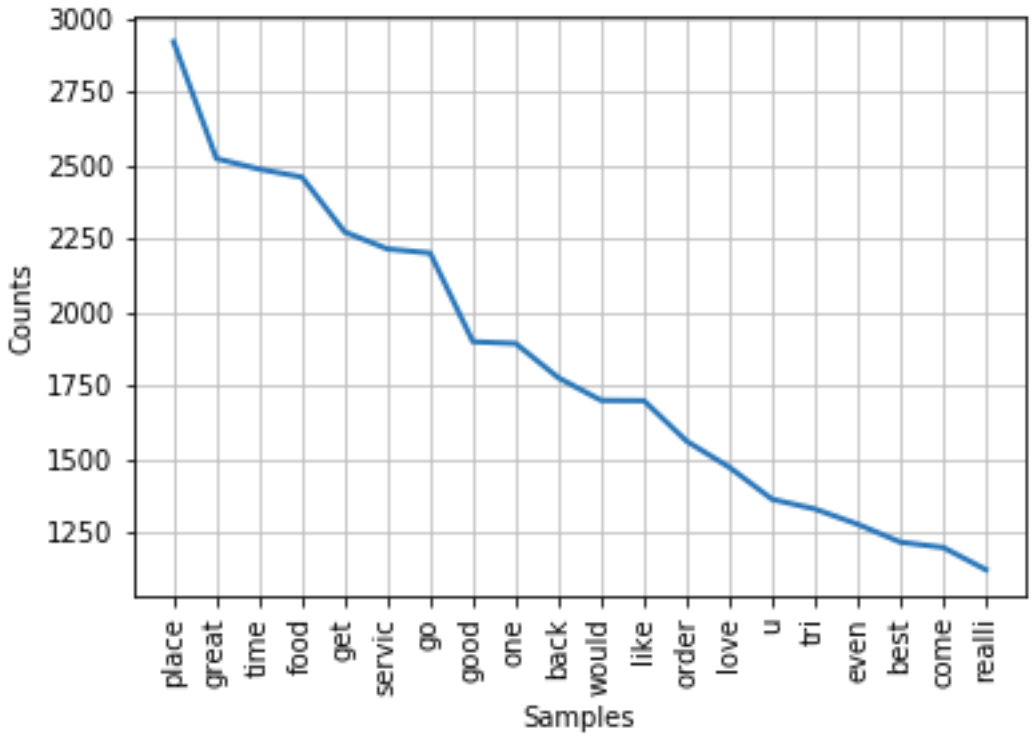
Wordcloud:

- From the above wordcloud we can see that most common theme was Service, Timings, Table service, Drinks more than food,location and in the food section it was pizza and burger.

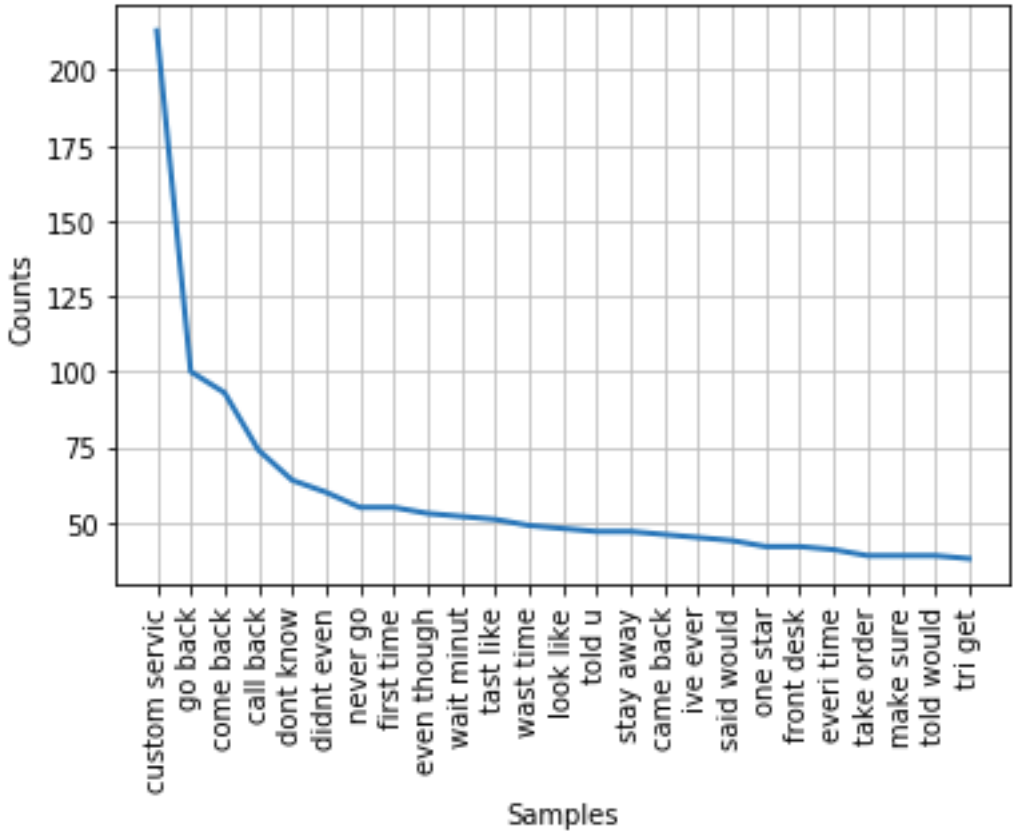
Data Preprocessing:

Removed punctuations, stopwords and then tokenized the words to perform stemming and lametting.

Unigram and Bigram frequency distributions: Frequency distribution of the most occurring words was plotted and we observed that place, time and service were the most commonly mentioned words.



Negative bigrams:
Frequency distribution of bigrams in the negative reviews were plotted and we saw that customer service was the most occurred bigram in the negative reviews.



CountVectorizer for text classification

In order to train our model we need to assign numbers to our words which are present in the reviews.Their are many ways to do this:

Bag of words: but that would be inefficient, all words have same importance, no semantic information is preserved.

Countervector: It can be used here as it counts the number of words used and the number of features depends on it. This converts our text features into numbers which helps in the model preparation.

Tfidf: Uncommon words are given more importance than the common words.

As countervector gives a discrete value of number of counts so multinomialNB should perform better for this counter.

TfidfVectorizer for text classification

- Bernoulli NB is good for making prediction if the features are in a binary form. Gaussian NB is good if the features are normally distributed.
-
- As many of the common words like "are", "the" etc dominate our model hence we can use tfidf vectorizer which stands for Term frequency inverse document frequency.
-
- Tfidf= Term Frequency* Inverse Document Frequency where, term frequency= number of occurence of the word in the document/ number of words in the document inverse document frequency= log(no. of documents/no. of documents containing the words)
-
- After using the vectorizer we trained the model with MultinomialNB and the results obtained were: The accuracy score using MultinomialNB with tfidf: **0.78**

Word Embedding Technique:

Now, as before, we import the tokenizer and the pad sequences. We'll create an instance of tokenizer, giving it our vocab size and our desired out of vocabulary token. We'll now fit the tokenizer on our training set of data. Once we have our word index, we can now replace the strings containing the words with the token value we created for them. This will be the list called sequences. As before, the sentences will have variant length. So we'll pad and or truncate the sequenced sentences until they're all the same length, determined by the maxlength parameter. Then we'll do the same for the testing sequences. Note that the word index is words that are derived from the training set, so you should expect to see a lot more out of vocabulary tokens in the test exam.

Training the Neural Network: We trained the neural network with a layer of Embedding, Conv1D and Maxpooling and observed a valuation accuracy of 0.93 which was a good result. But the training accuracy went up to 1.0 which might be due to over fitting.

To reduce the over fitting we can try the DROP OUT technique.

Drop out :
Drop out is used to reduce the problem of over fitting in a model. This technique works by randomly selecting the neurons to 0 at each update of the training phase. There's some debate as to whether the dropout should be placed before or after the activation function. As a rule of thumb, place the dropout after the activate function for all activation functions other than relu. In passing 0.2, every hidden unit (neuron) is set to 0 with a probability of 0.2. In other words, there's a 20% chance that the output of a given neuron will be forced to 0.
Accuracy obtained using drop out comes out to be : 0.95 val and 0.99 training, which is very much similar with and without drop out. This case may be because of the limited number of samples.

Implementing Early stopping¶
Early stopping is a method that allows you to specify an arbitrary large number of training epochs and stop training once the model performance stops improving on a hold out validation dataset.

Callbacks
They provide a way to execute code and interact with the training model process automatically.Callbacks can be provided to the fit() function via the “callbacks” argument.

Early Stopping in Keras
Keras supports the early stopping of training via a callback called EarlyStopping.

This callback allows you to specify the performance measure to monitor, the trigger, and once triggered, it will stop the training process.

The EarlyStopping callback is configured when instantiated via arguments.

The “monitor” allows you to specify the performance measure to monitor in order to end training.

Checkpointing in Keras
The EarlyStopping callback will stop training once triggered, but the model at the end of training may not be the model with best performance on the validation dataset.

An additional callback is required that will save the best model observed during training for later use. This is the ModelCheckpoint callback.

The ModelCheckpoint callback is flexible in the way it can be used, but in this case we will use it only to save the best model observed during training as defined by a chosen performance measure on the validation dataset.

Result: we can see that the model stops at Epoch 2, giving us the least validation loss and an increased accuracy of 95.63. Earlier without using the early stopping our model gave an accuracy of 94.6.

Recommendation and final findings:
- Our final model was able to predict the star rating with 95% accuracy.
- Most of the commonly occurring words were : Customer Service, Time, First visit, wait time etc which tells us these are the major factor which a customer pays attention before giving the review.
- Most important part was the Customer service which all the businesses should focus on.
- The subjectivity and polarity made us understand that more strong feelings are expressed, the more the overall comment is subjective.

Future scope: We can also use RNN and LSTM as RNN and LSTM perform better with sequential data, sentiment analysis and time series data as compared to the CNN and feed forward neural network.

RNN captures information from sequences and time series data. Can take variable size input and output. They work on the following recursive formula:

Ht= function(H t-1, Xt) where Ht is the ouput at time step t Ht-1 is the ouput at time step t-1 Xt is the input provided at current time step t

We take input from the current state(X1) and output from the previous state(S0) and feed it to the RNN, which gives us the state 1 : S1 S1*Wy = Y1(output)