Predicting Restaurant Closure using Yelp Dataset

Sheenam Gupta
UC Riverside
sgupt068@ucr.edu

Ponmanikandan Velmurugan
UC Riverside
pvelm001@ucr.edu

Krishna Kabi
UC Riverside
kkabi004@ucr.edu

ABSTRACT

Restaurant business is heavily dependent on the location, neighborhood competition and their past reviews. Much of which is available through online datasets. The user interaction has helped ease making investment choices towards the future of the restaurant business. This project predicts the likelihood of a restaurant closure within next three years. We use publicly available Yelp dataset for year 2016-2019 to extract insights about restaurant's success or failure. We applied decision tree and neural network and got an accuracy of 61% and 74% respectively. The project highlights what features, and tunable parameters aided in good model performance.

KEYWORDS

Text mining, Data preprocessing, Machine learning, Natural language processing

1 INTRODUCTION

With the ever-increasing reliance and abundance of big data, data mining has become one of the most widely used methods to extract information from large datasets. Often, data mining is referred to as knowledge discovery in databases (KDD). This consists of the extraction of non-trivial and previously unknown patterns of knowledge from large amounts of data. The KDD pipeline includes the processing of raw data, including data integration, cleaning, feature selection, and dimensionality reduction. Following this is the analysis, which often includes pattern discovery (and association and correlation), classification, clustering, and outlier analysis. Post-processing concludes the pipeline, consisting of evaluation, interpretation, and visualization. There are various techniques of data mining. Choosing a technique depends on the problem one is trying to solve. The models we have chosen to implement include Neural Networks, and Decision Tree. Using these methods, we intend to propose practical insights for customers or investors regarding a business closure.

2 RELATED WORKS

In previous years there have been a lot of studies on yelp datasets to predict restaurant success. Many papers have considered different parameters to predict the business closure. Some also tried to find which restaurant features have the most impact on the restaurant's success and predict the rating and success of the restaurant. For example, In the research completed by Wang, Zeng and Zhang [1], it aimed on predicting restaurant success. They did sentiment analysis on restaurant reviews and used multi-class classification algorithms to predict accuracy. It concluded that sentiment analysis helped in improving accuracy of model and the text is user reviews affect a restaurants rating. In the paper conducted by Lu, Qu, Jiang and Zhao [2], it aimed to predict whether a restaurant will be open or closed after a year. They considered many text-based and non-text features and concluded that text features do not help increase the accuracy of model whereas non text features have good impact on increasing the accuracy of the model.

3 DATA COLLECTION AND PREPROCESSING

This section explains how we collected, customized, and preprocessed the yelp dataset so that it can be used to train our models.

3.1 Yelp Data collection:- The data was downloaded from yelp website. The data was large enough to be handled on local resources, so we

took help of cloud services by uploading them on Kaggle and Google collab notebook. Yelp contains 6 files in its dataset, but only 2 files were used for this project – "business.json" and "reviews.json". Business file contains the details of all the business including "business address", "name", "stars", "categories", "attributes" and many more. And reviews file contains the reviews for all the business ids which are present in business file," username", "review_text", "star rating given by user" etc.

3.2 Feature generation:- Since, yelp business provide details of all businesses that are open and closed till current year, and the time period that we considered was 2016-2019, our initial effort was to extract features only for restaurant businesses out of all businesses between that period. The reason for choosing the time frame between 2016-2019 and not current year was because of COVID-19. The pandemic led to huge temporary and permanent shut down of businesses and can be considered as an anomaly in comparison to other years. We then considered restaurant's only for Texas state which had second highest count of restaurants in entire dataset. Finalizing if a restaurant is closed by 2019 was challenging since, no raw feature gave that information. To create the target variable all reviews dated between 2016-2019 were considered and a threshold for minimum reviews in a year was set as ten. Any restaurant with reviews less than threshold in 2017 were marked as tentative for further processing. All restaurants having reviews above threshold were marked as open for that year. Further, any business id whose review were not present in the following year was marked as closed. Consecutive years till 2019 were processed similarly. Figure 1 shows the counts of opened and closed restaurants for each year from 2016 to 2019 after performing all the above steps. The business ids which were marked as tentative were processed as follows. For each business if the last 5 reviews which had the phrase ''closed, out of service, shut down, not opened'' and its equivalent were filtered out.

Figure 1: Counts for opened and closed restaurants from 2016 to 2019

We initially thought if we find any such record, we would mark them as closed. On further analysis we figured out that such reviews many a times meant that the restaurant was closed just for that day or that time. Manual intervention was required to ultimately mark the business id as closed or opened.

There were some restaurants which were closed in 2016 itself or were not having any reviews after 2016, so these business ids were removed from the closed dataset because we wanted to consider the closed restaurants with a minimum year of operation as 1 meaning restaurants that were remained open after 2016. Finally, after merging the final closed and opened dataset, there were total 2883 restaurants in Texas from 2016 to 2019. Figure 2 shows the total opened and total closed count after the pre-processing.

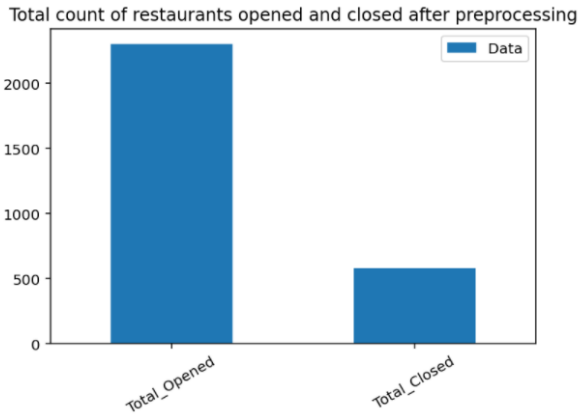

Figure 2: Total Counts of opened and closed restaurants in Texas

Below mentioned are the features generated and how they were generated :-

- **is_open and age :-** Yelp business dataset provides its own "is_open" feature but as mentioned earlier it had restaurants that are closed as of 2021 and we wanted restaurants that were closed by 2019. As described above the feature was generated and categorized as 0 for open and 1 for closed.
  For age/years of operation if a restaurant is in tentative dataset that did not have 'closed' phrase mentioned in last 5 reviews we could not simply tag them as open/close for that year. We agreed to use the last quarter as the determining factor for attaching the year of closure. For instance, if the last review was before October of that year, then the year of closure is the current year else following year. So if a restaurant have year of closure 2017, its age was 1 as starting date of all restaurants was considered to be 2016, same way restaurant closing in 2018 have age 2 and closing in 2019 have age 3 and is its open then 4.

- **Categories:-** Categories column has multiple features in string format. For our project we considered the cuisine of the restaurant and split it in 16 different columns using one hot encoding to analyze if cuisine of restaurant is one of the reasons for restaurant closure. Above figure 3 shows the one hot encoding for all the 16 cuisines. Some businesses were having multiple cuisine types, so for them those particular cuisine types are set to 1 and rest to 0 whereas for some business there was no cuisine type specified, they are kept in "Other" category.

- **Attributes:-** Attributes column had multiple features that described ambience, price range, 'has_delivery' and other additional feature can impact overall ratings, hence closure. Imputation of missing values and all categorical features were one-hot encoded which resulted in 15 columns.

- **Open_hours :-** To check if a restaurant has its open hours displayed on website or not. Three additional columns suggesting, if a restaurant is open for breakfast/ lunch/ dinner were also created.

- **Review Count :-** Although 'review_count' is available in business file. That gives total review count from the date the restaurant is open till 2021. Since, our time frame is between 2016-2019, we had to get equivalent count of reviews from user review data. The no. of reviews between 2016-2019 was fetched and then normalized with the restaurant's age.

- **Positive, Negative and Neutral Reviews: -** Earlier we thought using sentimental analysis we would categorize reviews as positive, negative or neutral received between 2016-2019. But that was modified as we see user ratings already gives that data explicitly. And we would achieve no additional value doing sentimental analysis. So, user ratings received between 2016-19 for a business_id that are > 3 was considered as positive. Similarly, ratings = 3 as neutral and < 3 as negative ratings.

- **Text based feature:-** User review can have a bigger impact on overall performance of the restaurant. The reviews per particular 'business_id' was grouped. It was then cleaned by removing stop words, punctuation, performing lemmatization. Since, unigram features do not provide much context, bigram features were created. From top 30 bigram features, few phrases were considered which would give idea about user's view on food and location. These were used to generate frequency of such words in the reviews. For instance, occurrence of 'good food ', 'great food' etc was put in a new column 'bi_good_food'. Similarly, 'nice place', 'great service', 'great place' etc was put in another column as 'bi_good_loc'. Figure 3 shows the Top 30 Bi-gram features from which phrases were considered to create the above columns.

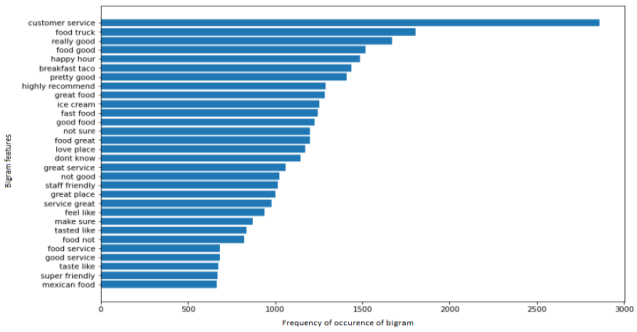

Figure 3: Top 30 Bigram features (reviews between 2016-2019)

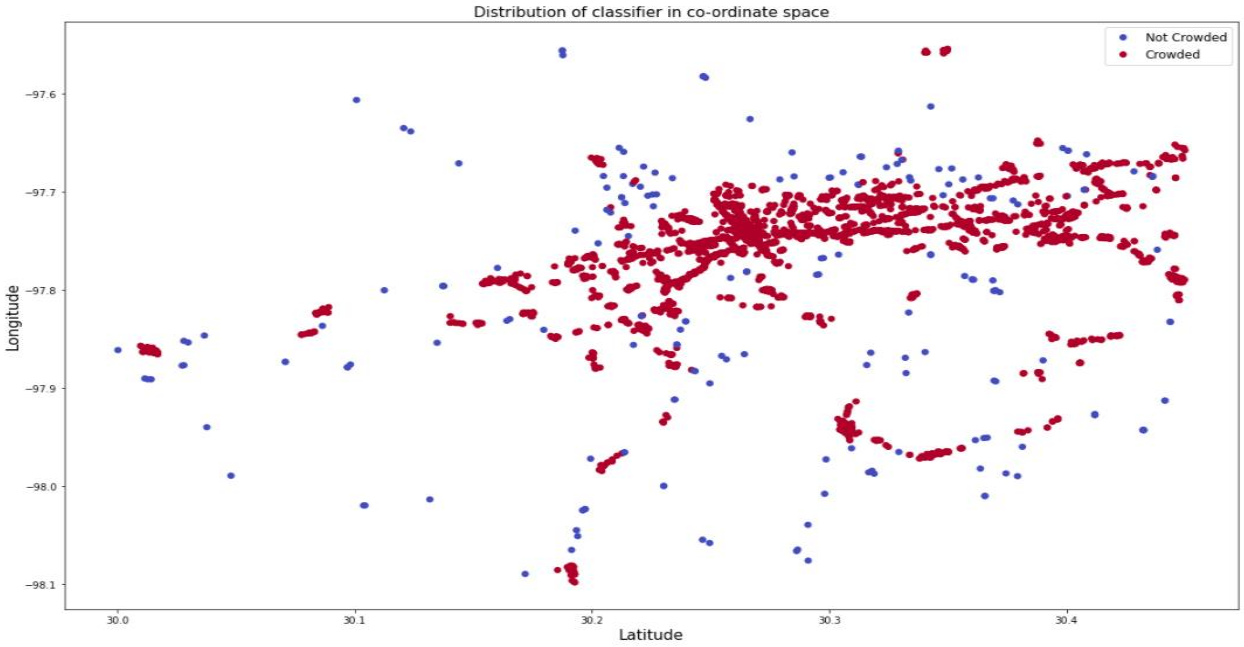

Figure 4: Crowded/Not Crowded Restaurants in Texas

- **Crowded Restaurants: -** This feature has the potential to determine the restaurant closure and had to be extracted from the raw business dataset. Initially, we calculated the haversine distance which is one of the metrics that defines the distance between two geographical latitude and longitude points. Using this metric, we then did the grouping by making using of the DBSCAN algorithm. But due to the sensitivity of the DBSCAN towards the hyperparameters we could not find any optimal values for our case. In search of a different algorithm, we assumed that the earth at close distances can be considered as a plane instead of a sphere. By using this we can calculate the distance between

two spatial points by converting them to cartesian points and using the Euclidean distance. In order to optimize this further, we used the KD-Tree data structure to stores the spatially near points together. Combining the assumption and the data structure, we used the ball tree algorithm to label the crowded and non-crowded restaurants. We used the crowd_factor as 5 and radius as 0.5 meaning if a restaurant has 5 nearby restaurants (including itself) within a radius of 0.5 miles, then it is classified as crowded. Figure 4 shows the result of our findings.

- **Chain Restaurants: -** This feature can be useful to compare the success and failure of independent restaurants with popular franchises in the same locality. This feature was also extracted from the raw business file. Initially, we went with regex modules. But soon we found that to be tedious due to raw number of unique cases in restaurant names. We assumed that restaurant with almost same names (without considering latent meaning) are considered to be part of a chain. With this notion, we created a Lucene indexer and searcher to parse the data and find almost similar restaurant names. We then used this to classify the restaurants as chain or independent. We used the TF-IDF model for the ranking algorithm. The extraction process involves creation of a search engine (using Lucene) on which each restaurant name is fed to find the near match names and search scores. This search score is compared with a hyperparameter threshold to achieve best results. The search algorithm follows the TF-IDF model to compute the search scores. Figure 5 shows the chain restaurants in Texas.

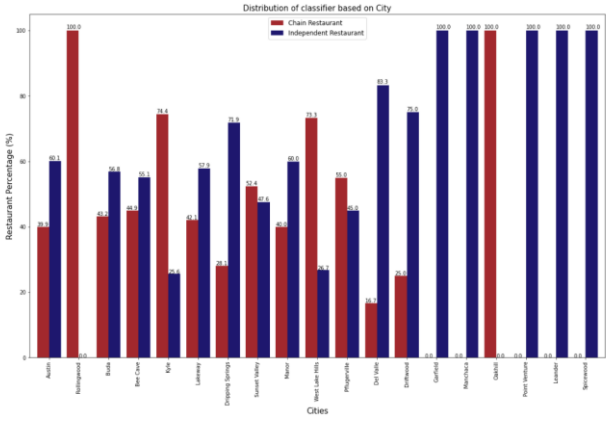

Figure 5: Chain Restaurants

4. EVALUATION METRICS:

Since we are dealing with restaurants business closure the dataset clearly indicated an imbalanced dataset. So below two approaches were tried:

1. Use the imbalanced data and perform evaluations to find the efficient F-Score and compare it with respect to the other model.
2. Under sample the majority class using the near miss strategy and compare accuracy metric with respect to the other model.

5. PROPOSED METHODS

This section explains about the methods implemented in this paper. Before model training, we dropped two features (age and years_of_operation) as it was leading to data leakage. Then the normalized columns created taking total count received between 2016-2019, “review_normalized“, pos_norm”, “neg_norm” and “neutral_norm” features were also removed as we later realized the age taken into consideration for creating the feature was not entirely justified. Similarly, few redundant or correlated features were dropped by plotting a correlation plot of all features as shown in figure 6. The dataset was split into training and test. And the training data was used for 5-fold cross validation. All the numerical features were scaled.

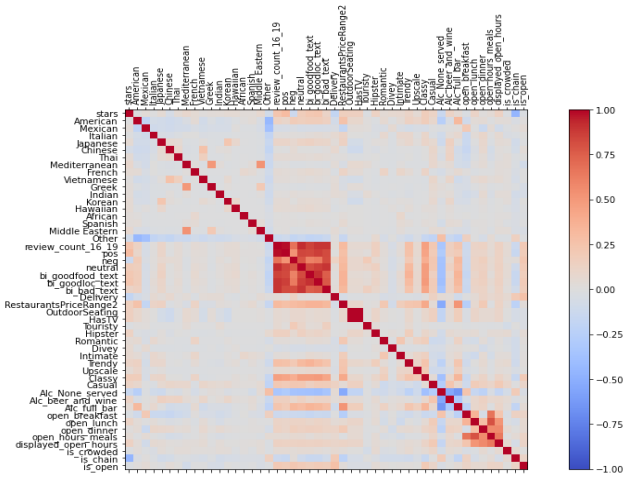

Figure 6: Correlation Matrix

5.1 Decision Tree:- Our first method is to implement the Decision Tree classifier to predict the restaurant closures. Different features were chosen and compared with respect to the validation set performance to ensure good performance of the model with respect to unseen. Data was split into training set and testing set in the ratio 80:20 respectively. This model was implemented from scratch. We used the entropy as the splitting criterion over the gini index because our research showed entropy being a complex technique is good for accurate selection of features. First the best split at each level of the tree is calculated based on Information gain. Information gain is calculated based on entropy which is a metric to measure the impurity of a split. With this notion, we try to maximize the information gain at each level by minimizing the entropy (or) split impurity (or) maximizing the probability of a particular class label. However, this posed difficulties to features having a greater number of unique values. This is a common problem with the ID3 algorithm, and we switched to the C4.5 which was a successor to the previous algorithm. Therefore, for the new C4.5 algorithm we used the information gain and normalized it with the split ratio. This ensures that feature having a greater number of unique values were offered less gain ratios than features have less unique values. The tree construction was done dictionary data structure and Gain Ratio as the splitting criterion to ensure that best split was chosen at every level. We then removed the used categorical features so that that does not appear in the same subtree again. However, we did not remove the continuous features as this allowed fine grain splitting with respect to continuous values. Recursive call was made to the C4.5 algorithm to construct the entrie tree until any one of the stopping criteria's were met:-

- All samples for a given node belong to same class (an ideal pure split).
- There are no more attributes for further splits.
- There are no samples left on the dataset.
- If a tunable height is reached.

Dynamic height was passed to the tree to know where the tree gives the reasonable performance with respect to the validation set and to avoid overfitting or underfitting with respect to validation set. Best features were chosen for the good performance rate by running the model on different features and testing with them. The model gives the accuracy of 61%.

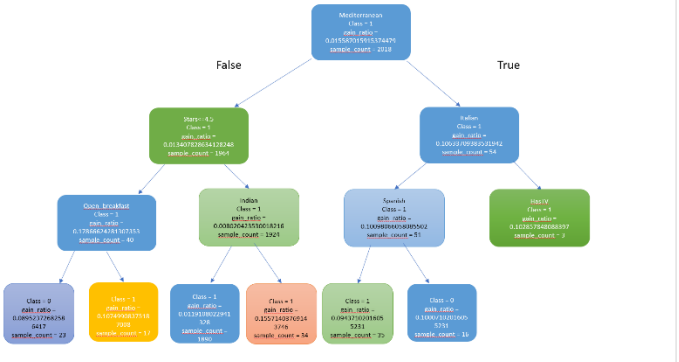

Figure 7: Visualizing DTrees (height = 3).

5.2 Neural Networks:-

Usually, neural networks works better than traditional machine learning algorithm to capture the non-linear behavior. Here, we tried to leverage the functionality of Keras package and implemented simple neural network. A single

dense layer was used with sigmoid 'tanh' activation function. The output layer had sigmoid activation function to output the probabilities of sample belonging to class 1. The loss function used is 'binary_cross entropy' which would return high values for bad predictions and low values for good predictions. It calculates a score that summarizes the average difference between the actual and predicted probability distributions for predicting class 1. A a perfect cross-entropy value is zero.

$$H_{p(q)} = -\frac{1}{N}\sum y_i \cdot \log(p(y_i)) + (1 - y_i) \cdot \log(1 - p(y_i))$$

The hyper-parameters considered for the model was:
Batch size: 16
Layer: 1
Neurons: 200
Epochs: 300
Optimizer: Adam
Loss function: binary_cross entropy
Learning rate: 0.00001

All the hyper-parameters were tuned using grid search. Ideally, adding more layers usually help to improve model performance but we did not see that improvement which could be explained due to two reasons. Firstly, deep neural networks are more equipped to handle complex data such as text, images or audio and adding more layers for this dataset was just overfitting the training data. Second of call, we have less data to work with since, we focused on a single state in US. This could also lead to poor performance.

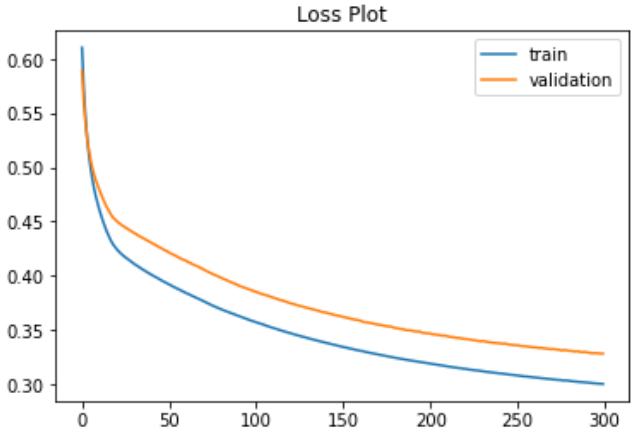

Figure 8: Loss plot

6 EXPERIMENTAL EVALUATION:

We initially ran two experiments to see which method improves model performance. As mentioned earlier in Section 3, we wanted to see how under sampling the majority class affects the model performance with respect to validation set. So, we tried four methods, near miss, condensed nearest neighbor, Tomek links and neighborhood cleaning rule.

Near Miss V3: This method selects majority class samples that are closest to each example in minority class. This under sampling strategy gives equal distribution between majority and minority class.

Condensed nearest neighbor (CNN): It initially starts with one sample and then scans the dataset for all samples. It adds them to the "*store*" only if they cannot be classified correctly by the current contents of the store. It uses k-NN algorithm to determine to do so.

Tomek Links: This method finds pairs of examples, one from each class, which have the smallest Euclidean distance to each other. This means that in a binary classification problem with classes 0 and 1, a pair would have an example from each class and would be closest neighbors across the dataset [10]. In words, instances a and b define a Tomek Link if: (i) instance a's nearest neighbor is b, (ii) instance b's nearest neighbor is a, and (iii) instances a and b belong to different classes [10].

Neighborhood cleaning rule: This under sampling strategy combines the advantages of Edited nearest neighbors and K-NN. Intuitively, the strategy removes the noisy points with respect to the majority class. Further, it also removes the repeated points in the majority class.

We plotted the error bars for all four cases. The Near miss and CNN method produced nearly balanced set and Tomek links and NCR produced still gave imbalanced distribution. So, we compared accuracy for former two and F1 score for latter two to see which method performed better as shown in Figure 9.

Since, 'Near Miss' gave better validation accuracy, and gave better balanced data, we used 'Near miss' to train our model instead of CNN for the balanced case.

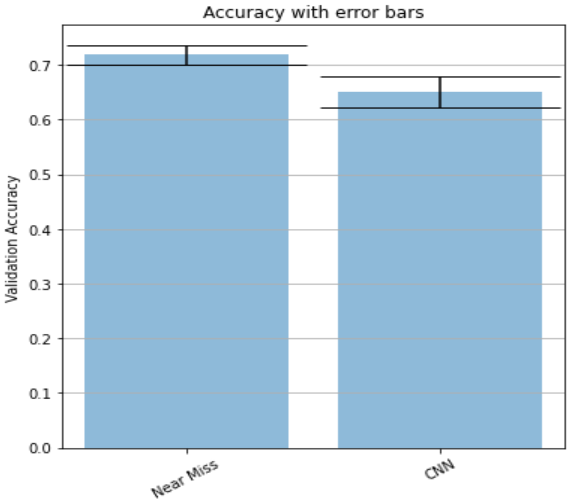

Figure 8: Comparing Near Miss with CNN

For the unbalanced case, we found the Neighborhood Cleaning Rule to be efficient than Tomek links with our experiments and used it to perform our model construction.

Similarly, when we ran the corrrelation plot, three features, 'pos', 'neg' and 'neutral' showed higher correlation with total review count and we had to drop them. But these features seemed more important in determining the business closure since, if the number of positive reviews for a restaurant is higher than negative reviews then its usually less likely to close in future. So, we ran two experiments:

- Using positive, negative, neutral count of reviews and dropping total_review count as that was highly corelated with former three.
- Using total_review count as feature and dropping pos, neg and neutral features.

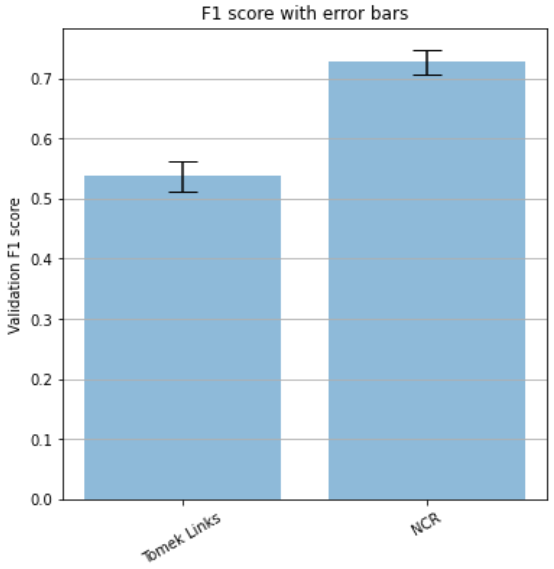

Figure 9: Comparing Tomek links with NCRule

An additional experiment was run, to see if numerical columns such as star rating, total review count and price range indeed had any impact on model performance. The error bars for all three cases were plotted as shown in Figure 10.

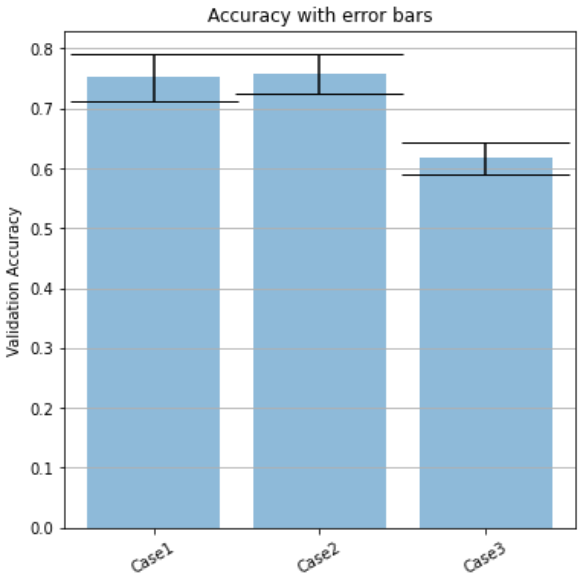

Figure 10: Comparing feature selection

From the study we concluded that case 1 having restaurant features such as review count, category, chain information and crowded information to be more successful in predicting the restaurant closure than case 3 which is just using the categorical features. However, based on the study we could not find any statistical significance between case 1 and case 2 because of the overlapping.

6 Decision Tree and Neural Networks Results:-

We implemented the two models as stated in above sections and analyzed the performance metrics of the models to understand their efficiency and advantages.

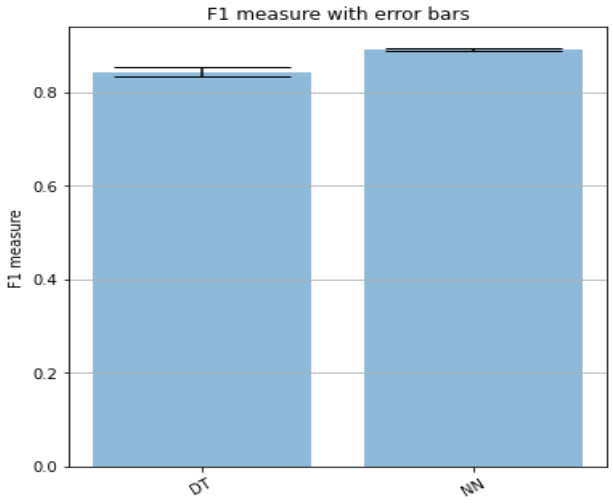

Figure 11: Comparing Imbalanced Models

The figure 11 depicts the f-measure comparison of the Decision tree with Neural Networks in case of imbalanced data. We found the NN to be performing better. Additionally, we found NN was capable to capture the continuous feature more efficiently than the Decision trees. We were not able to test all the continuous feature in the Decision trees as it took considerable time to run a DT construction with continuous variable as it need to check all the possible value split for a continuous variable. Therefore, the Decision tree was constructed only based on the categorical and limited continuous features.

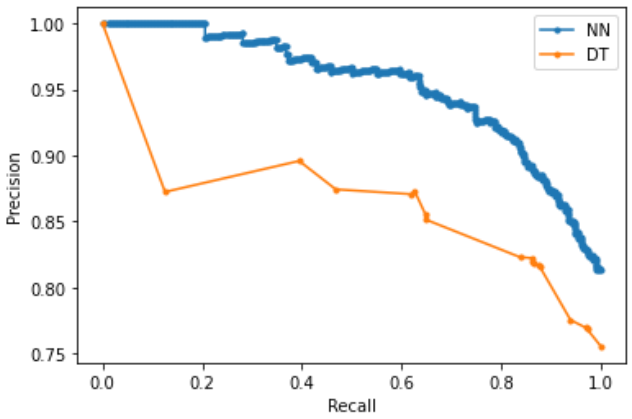

Figure 12: Comparing Precision Recall curves

The figure 12 compares the precision and recall curves of bot the models. Both the models were performing almost similar but Neural networks was able to achieve more Area under curve than Decision Trees.

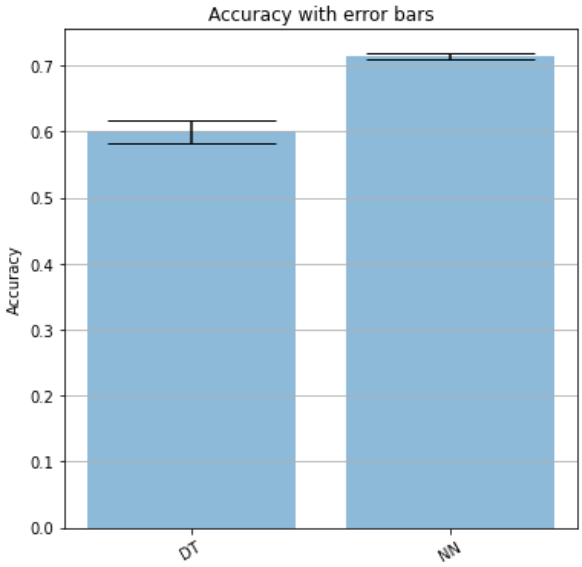

Figure 13: Comparing Balanced Models

The figure 13 depicts the comparison of DTrees with NNets with respect to accuracy metric. The Near Miss under sampling gave an equal representation of both the class giving the way to use accuracy as the comparing metric. Due to same limitations stated before on the imbalanced case, DTrees was unable to capture the insights of continuous features in reasonable time. But considering the time taken aside running the

DTrees on the continuous feature was giving almost similar results as NNets.

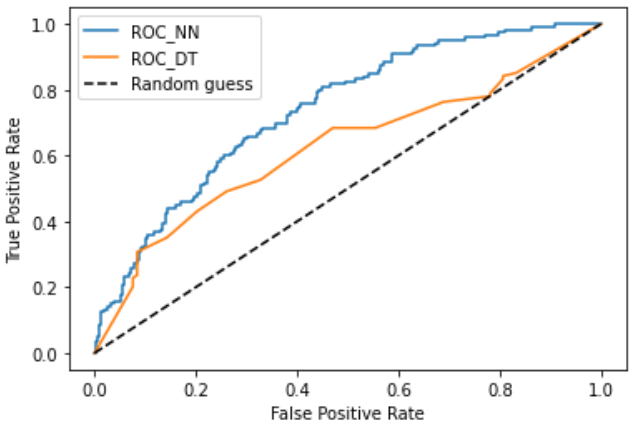

Figure 14: Comparing ROC curves

The figure 14 depicts the comparison of ROC curves of both the models. Both the models were performing better than the general case and NN performing better than Decision trees.

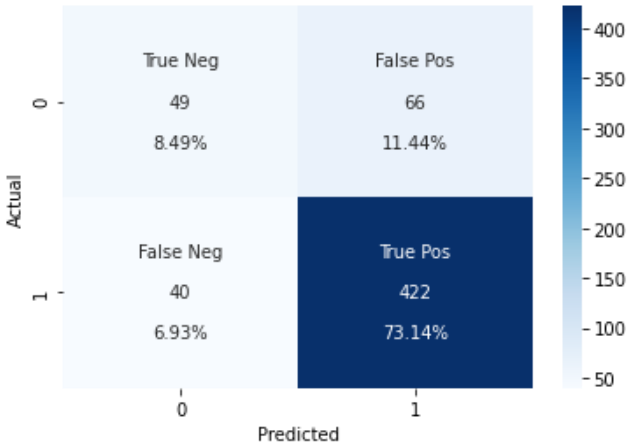

Figure 15: Confusion matrix for Imbalanced set by Neural networks

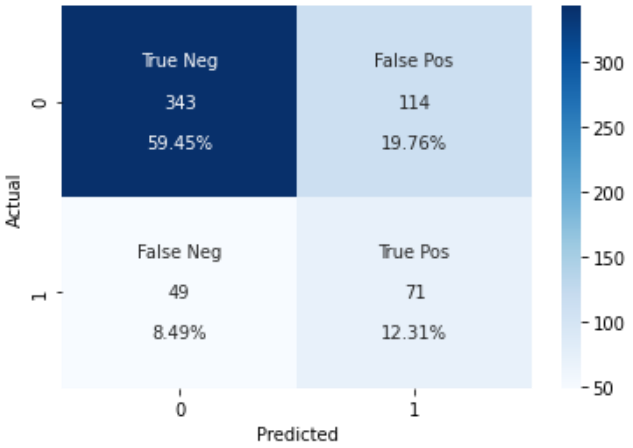

Figure 16: Confusion Matrix for Balanced set by Neural networks.

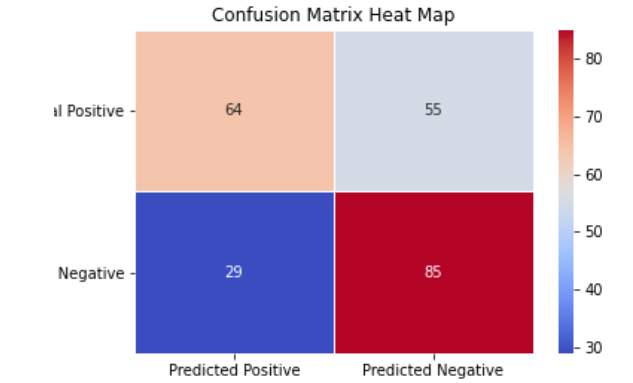

Figure 17: Confusion matrix for Imbalanced set by Neural networks

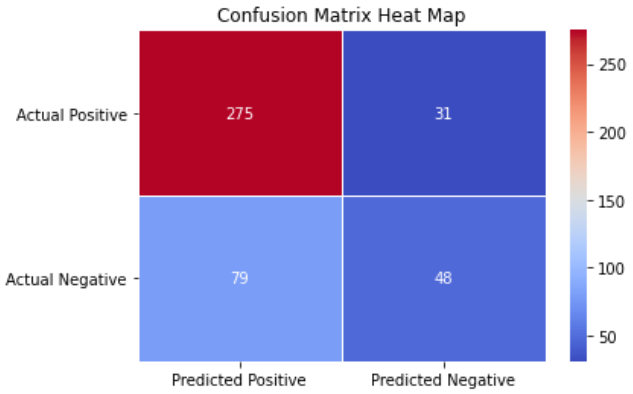

Figure 18: Confusion matrix for Imbalanced set by Neural networks

The figures 15-18 depicts the confusion matrix generated by the Neural networks and Decision Trees in case of imbalanced and balanced cases, respectively. We can infer that, in case of Balanced case, primary objective True positive was greater than False Negatives. This is important for our project as we do not want to wrongly classify the positive class.

7 DISCUSSION & CONCLUSION:

Our experiment confirms that neural network was better in predicting business closure when balanced set is considered. However, we would also like to highlight that the balanced set had very less data points and a higher number of samples could have helped in better training and hence, better test accuracy. Since NCR is better at under sampling as it eliminates the noisy samples, it does not provide a balanced data set. So, in future, we could try applying both under sampling and oversampling using Smote and then find evaluate the model using accuracy. We also, think the model lacks generalizability because of its construction with limited locational parameter and tuning it with respect to the validation set. However, further analysis in this places will result in a efficient prediction.

8 REFERENCES:-

[1] https://towardsdatascience.com/using-yelp-data-to-predict-restaurant-closure-8aafa4f72ad6

[2] https://thenewstack.io/can-yelp-data-predict-restaurant-closures/

[3] https://jiamingqu.com/files/Yelp%20Prediction.pdf

[4]https://www.kaggle.com/yelp-dataset/yelp-dataset

[5] https://towardsdatascience.com/feature-selection-correlation-and-p-value-da8921bfb3cf

[6] https://medium.com/@dtuk81/confusion-matrix-visualization-fc31e3f30fea

[7] https://thedatafrog.com/en/articles/text-preprocessing-machine-learning-yelp/

[8] http://cs229.stanford.edu/proj2017/final-reports/5244334.pdf

[9] https://towardsdatascience.com/text-mining-and-sentiment-analysis-for-yelp-reviews-of-a-burger-chain-6d3bcfcab17b

[10] https://machinelearningmastery.com/undersampling-algorithms-for-imbalanced-classification/