PREDICT USER RATINGS BASED ON REVIEW TEXTS

NIHARIKA BANDLA
nb5427@rit.edu

ABSTRACT
Many of the e-commerce websites are mostly adapting to the day-to-day new changes occurring in the market. One of many such recently emerged trend is to be able to get feedback from the users or buyers depending on the type of business. This helps the vendors not only to get to interact with the buyers and know about their opinions on the product but also to get an idea of how the product is doing relatively to a different product line/competitors. It primarily is used as an online survey which helps in the prediction of how well a product from the same line might be received by the buyers. This serves a general platform for market analysis as well. Hence, most businesses that took to online business, are following the trends of accepting ratings and reviews from the buyers. Here comes a problem, along with the benefits. Some users give reviews and corresponding mismatched ratings for whatever reason. Due to this, some of the vendors business/reputation may get tampered as well. Therefore, a feasible model is proposed where it indeed predicts the ratings based on the review texts given by the users.

1. INTRODUCTION
As most of the businesses took to online marketing, the current trend imposes a lot of interest in the reviews and ratings given by the users who already purchased a particular product. In fact, the particular product’s sales are to most part dependent on the previous user’s rating/review. Some of the most popular websites that follow this kind of a pattern are Yelp, Amazon, eBay. All of these websites provide a way for the users to be able to give feedback on a product purchased within the website.

The current project makes use of such review texts in order to perform an inference-based analysis and thereby, train the model to be able to predict the ratings based on these texts. For this purpose, Yelp text reviews data-set has been chosen to be used for the current project because it has got a very rich crowd-sourced data which can be very useful for training purposes. The data-set consists of all the information about a particular business, tips, check-ins, reviews texts, star ratings, votes etc.

The main idea behind the project broadly lies in predicting the class of an unknown instance. Specifically, it means that a given review text will be needed to be classified as three different being either “bad”, “average” or “good. In a much broader perspective, it is to determine the contextual polarity of a given text sentence.

A comparative analysis of the current informative data-set would be able to help build an efficient and feasible predictive model. Also, using a rich data-set such as Yelp can be helpful to understand the business needs and the product standing in the market better and can also get a relative standing of the product which can be greatly helpful to the business vendors.

2. PROJECT GOAL
The objective of the project is to implement a model that predicts user ratings based on the review texts.

3. MOTIVATION
In the recent times, several online shopping websites have come up where buyers can give out reviews to the products they buy and also interact with the sellers directly. eBay and Amazon are some of the websites that provide with such a facility. Every review is associated with a star rating starting from one star to 5 starred ratings. There are cases where the rating is very disproportionate to the review. This certainly is a great impact on the business. A model which can assign the right star rating based on the review text could be very useful in such scenarios which would in a way lessen the number of such disproportions.

4. DESIGN
The obtained data-set is in the JSON format containing 1 million records. Some of the key statistics of Yelp data-set[1] are as follows:

- Businesses- 42,153
- Check-ins- 31,617
- Users- 212,898
- Tips- 403,317
- Check-ins- 31,617
Reviews- 1,125,458

Also, as a next step I have chosen to set up the required environment for the project. Hence, I made use of a no-SQL database called MongoDB in order to store the data since the available data is in the form of JSON objects. As part of initial analysis, I chose to split the data-set into equal parts. This is because the entire data-set has taken up to 3.9GB of storage which is really huge. Therefore, I have chosen the data-set with a compatible size which is of 50,145 instances. Now, as a next step, I have taken some statistics of the star ratings as well which are present in the reviews data-set.

- one-star- 110,772
- two-star- 1,02,737
- three-star- 1,91,001
- four-star- 252,666
- five-star- 3,15,887

A lot of literature review [4][12][3][2][6] has been involved in order to get a better light certain concepts from subjects like Sentiment analysis, Machine learning and Natural language processing. One of the categories that my project could fall under is definitely the supervised learning. I have come to this conclusion based on this research. The main reason for this is that a prior knowledge of the target variable is already known, that needs to be predicted at the end of my project. I will be predicting the rating of the review text in my case. Hence, it very well can come under supervised learning category. Post this initial analysis on the topic, I have spent the most time in data-set preparation. The approach that has been chosen to be undertaken in this specific project can be categorized as the Bag of Words approach[9]. Hence, the steps involved can be called as feature extraction [7] procedure. The steps involved in data-set preparation are:

- Feature reduction
- Stemming
- Bi-gram technique

In the Feature extraction step, I have reduced my number of classes from 5 to 3 in the following manner. I have taken an equal number of samples from all the 5 classes from the entire data-set of 10,000 each. Here, I have programmatically changed the labels of reviews whose ratings were "2" to "1" and similarly the ones that were rated as "4" to "5". This is because it is most likely that the degree of a sentiment of a 4 starred rating is closer to the ones that are rated as "5". This part is classified as the class reduction process for my project. I currently have it simplified to three classes where 1 means bad, 2 meaning average and 3 being good.

As a next step, I have made use of the porter’s stemming algorithm[11] to continue with the process of stemming the words. This process helped me to get to the root form of every word which simplifies the classification process as the stemmer reduces a particular word to its dictionary form and can, therefore, represent most words in the data-set and avoid redundancy. This also helps many words and its derivatives to be in the same base form.

As a next step, I have made use of the porter’s stemming algorithm[11] to continue with the process of stemming the words. This process helped me to get to the root form of every word which simplifies the classification process as the stemmer reduces a particular word to its dictionary form and can, therefore, represent most words in the data-set and avoid redundancy. This also helps many words and its derivatives to be in the same base form.

In the final step, I decided to pair the words in order to conserve the ultimate contextual meaning of the sentence.

For this purpose, I have first tried to go ahead with a uni-gram approach, but again it killed the importance of a sentence like “this is not good”. In order to better the case, I wanted to use a bi-gram approach that holds “not” and “good” together as a single word as “not good” and conserves the meaning.

### 4.1 Dataset division

[Figure 1: Data-set division]

Once the data-preprocessing step has been completed, I have split the data-set into two parts based on a 75-25 percentage split. This resulted in a total of 37,500 training instances and 12500 test instances.

### 4.2 Solution design

I have created an interesting solution design that describes my model and approach diagrammatically, so it would be easy to understand the entire process at a glance. The design emphasizes the importance of supervised learning model, feature extraction methodology and the role of machine learning classifier algorithm.

[Figure 2: Solution design]

Following the solution design, I worked on preparing the
5. TECHNOLOGIES USED

- Python for programming
- Pymongo for interaction of python with mongodb
- Pycharm development environment
- scipy and numpy for scientific computations
- matplotlib for plotting graphs and confusion matrices

6. IMPLEMENTATION

The current problem implementation includes the identification of feature vectors and the machine learning algorithms implementation in order to develop the final predictive model as mentioned in the project goal.

6.1 Feature Vectors

Feature vectors can be considered to be the most critical components to the present solution design as they are mostly responsible for the way the machine learning algorithm responds to the training of the data-set and thereby its accuracy. Therefore, some of the feature vectors have been identified which would fit in the scope of the solution.

The first of the feature vectors includes the class reduction process. Initially, the data-set consisted of 5 classes consisting of 10,000 instances of each. Every class belongs to each star rating. Like 1-starred ratings, 2-starred ratings and so on. Now, 1-starred ratings and 2-starred ratings are both combined as one class which is categorized as "bad". 3-starred ratings have been left to be as the "average" class and 4-starred ratings, 5-starred ratings have both been combined to be as the "good" class. This way the model will be trained to predict the unknown class ratings into only three different categories.

The second feature amongst the feature vectors is the Bag of words model. In this model two main important steps have been implemented. They are counting and tokenization. As part of the counting process, the entire vocabulary in the data-set is counted and the obtained unique number of words in the data-set are 40,360. Coming to the tokenization part, an integer value is assigned as an id to every possible token which is obtained right after the sentence splitting phase.

The third feature vector is to eliminate stop words. Sometimes the words frequency becomes extremely high with the repetition of each word in every document. This can also mean that the word may be quite common and make not much difference contextually as well. Some of such examples are “a, an, the”. Such words are eliminated to get only important words in a particular sentence in a document.

The final feature is the implementation of the n-gram technique. Based on the data-set size, I have chosen to implement a bi-gram technique which would pair up the words in the sentence based on the number of grams specified. In a bi-gram technique, the number of grams is 2. Hence, every two words in a sentence are paired in order to maintain the contextual significance of a particular review texts.

6.2 Algorithms

According to the solution design of the problem, the scope of the problem very much lies under the supervised learning strategy within the machine learning space. Therefore, in order to predict an unknown class from known attributes based on the derived inferences from the training data-set, I have chosen two useful algorithms for that space. They are the KNN algorithm and linear SVC algorithm.

As part of the KNN algorithm implementation, a certain number of steps have been followed in order to accomplish the classification problem. They are, to find the distances from each of the given instances with that of the unknown instance using Euclidean distance measure, then ranking the distances in the ascending order. Then picking the neighbors based on the given k-value and then finally do a voting and assign the unknown to a class that has the majority votes.

The process of choosing k is extremely critical in the entire process. I have followed to methods for this purpose. One is by the hit and trial method and the other is that $k = \sqrt{n}$ method. Here, by computing the square root of 37500 training instances, I ended up at 194. Therefore, I chose to
test the algorithm for a range of k-values, between 100 and 200 and take the maximum accuracy to be the optimum k. The graph showing k-values is given below:

![K value plot](image)

Figure 4: Optimum k-value plot

The conclusion from the above figure clearly is that the highest accuracy is at $k=180$ which is an accuracy of **59.12**. Later, an implementation of a tri-gram technique has improved the accuracy to **59.68**.

The second kind of algorithm which I have chosen to use to get a better accuracy is the **Linear SVC algorithm**. This is categorized to come under the Eager learning algorithms which builds its own training model in order to train and yield better results. Part of the algorithm is to be able to find an equidistant hyperplane that separates two classes exactly into two halves. In the case of my problem where there are three classes, the algorithm provides three different models which are the bad-average model, average-good model, and the good-bad model and then comes up with the classification process. However, better results along with better accuracy have been obtained only by the linear SVC algorithm. The accuracy obtained is **76.4**.

7. RESULTS

The algorithms usage on the given data set yields the following output based on each algorithm. Before, presenting the results, it is important to know that the algorithms have both been tested for accuracy with and without feature vectors. The difference with the feature vectors being present or not has impacted the results in a certain way. However, for heavier data-sets with considerably higher amounts of data instances present, the feature vectors can then play a vital role. Part of testing for this accuracy with and without the presence of feature vectors, the implementation of a tri-gram model has gone into consideration. The accuracy for KNN has increased from 59.12 to 59.68 which is not quite significant in this case. But in the case of linear SVC, a slightly more significant difference has been recorded. The accuracy went from being 70.31 to 76.4. This can definitely be increased in case of larger data-sets. The accuracy predictions of each algorithm on the model is visually represented by the confusion matrices.

7.1 KNN algorithm before implementing feature vectors

![Before Feature Vectors implementation](image)

Figure 5: Before Feature Vectors implementation

The classifier algorithm has been run to test for the accuracy without the implementation of the feature vectors. The results are visualized in the following format. The obtained accuracy at this step is 51.12.

7.2 KNN algorithm after implementing feature vectors

![After Feature Vectors implementation](image)

Figure 6: After Feature Vectors implementation

In this step, the classifier has been run on the training data set after the implementation of feature vectors such as the bag of words model, stop words elimination and a bi-gram model. The accuracy obtained at this point is 59.12. A slight improvement over this has been obtained when a trigram model is obtained. The accuracy at this point is 59.68. Some of the varied feature vectors also like taking into account of the TF-idf weights of words along with significantly
larger data-sets could have certainly improved its accuracy to a greater extent.

7.3 Linear SVC before implementing feature vectors

At this stage of the project, I chose to find an algorithm which would certainly give better results compared to KNN. I had a lot of learning at this stage which included a study about Eager learner classifiers vs lazy learners[8]. This detailed study has given me a path to follow in terms of the second algorithm selection. Upon the study of the paper, it was clear that KNN is a lazy learning algorithm where it does not build its training model unlike in cases of eager learning classifiers, but it stores the training data-set and works on the same. Therefore, the accuracy might tend to drop down a bit. Hence, I have chosen the linear SVC classifier algorithm for this purpose which is categorized as an eager learning classifier. The results obtained have been better when tested on this particular data-set when compared to the KNN classifier. The accuracy obtained at this stage is 70.31.

7.4 Linear SVC after implementing feature vectors

In this step, the same process of removing the feature vectors has been repeated to test for comparison purposes and the results did show up impressively. The removal of feature vectors includes the removal of stop words, n-gram model, and class reduction techniques. The accuracy obtained at this step is 76.4. The accuracy obtained by the linear SVC algorithm can be regarded as an output not based on more of inference analysis but also a strategy that could be useful both in classification and clustering techniques.

7.5 Output

As a part of the testing process, the predictive model that has been developed through the project was checked against some of the random Yelp reviews from the websites. The model has predicted in some cases the exact star rating associated with the review rating online. As the goal of the project is to predict user ratings based on review texts, the prediction includes the review class identification to fall into three broad categories as mentioned earlier. They are, **bad** (1 star), **average** (3 stars) and **good** (5 stars). Some of the samples of are shown below:

In this step, the same process of removing the feature vectors has been repeated to test for comparison purposes and the results did show up impressively. The removal of feature vectors includes the removal of stop words, n-gram model, and class reduction techniques. The accuracy obtained at this step is 76.4. The accuracy obtained by the linear SVC algorithm can be regarded as an output not based on more of inference analysis but also a strategy that could be useful both in classification and clustering techniques.

7.5 Output

As a part of the testing process, The predictive model that has been developed through the project was checked against some of the random Yelp reviews from the websites. The model has predicted in some cases the exact star rating associated with the review rating online. As the goal of the project is to predict user ratings based on review texts, the prediction includes the review class identification to fall into three broad categories as mentioned earlier. They are, **bad** (1 star), **average** (3 stars) and **good** (5 stars). Some of the samples of are shown below:

Figure 7: Before Feature Vectors implementation

At this stage of the project, I chose to find an algorithm which would certainly give better results compared to KNN. I had a lot of learning at this stage which included a study about Eager learner classifiers vs lazy learners[8]. This detailed study has given me a path to follow in terms of the second algorithm selection. Upon the study of the paper, it was clear that KNN is a lazy learning algorithm where it does not build its training model unlike in cases of eager learning classifiers, but it stores the training data-set and works on the same. Therefore, the accuracy might tend to drop down a bit. Hence, I have chosen the linear SVC classifier algorithm for this purpose which is categorized as an eager learning classifier. The results obtained have been better when tested on this particular data-set when compared to the KNN classifier. The accuracy obtained at this stage is 70.31.

Figure 8: After Feature Vectors implementation

Figure 9: Sample one-star review

The corresponding rating to this review, when pasted on to the program console, has predicted the same starred rating as well. The figure is shown below:
Another review belonging to the "average" class has been used for testing and the review that is used is the following:

Figure 11: Sample three-star review

The corresponding result to this review that the predictive model actually predicted the right rating when the same review has been pasted on the console and as an input to the predictive model.

Figure 12: Predicted three-star "average" rating

The final test to see the review- ratings that belong to "good" class have been identified. A random sample five-star rating review has been selected from the yelp reviews to see if the prediction still holds good. The sample review is shown in the following figure:

Figure 13: Sample five-star review

The model correctly predicts the right review for the model and identifies its class rightly to be that it belongs to the "good" class. The result has also rightly predicted as the 5-starred rating which implies it does belong to the good class. The following output is obtained when the review is pasted onto the console.

Figure 14: Predicted five-star "good" rating

All of the outputs show the right predictions and this tells us about the correctness of the model. During the testing process, I have also come across a few review texts for which the model did not predict as per the actual rating. For example, in few cases, an average rating with a 3 star rating has been predicted to be a 1-star rating which is a "bad" class. But there were also cases where a "average" review has been given a 5-star rating on the website. However, the model developed has given the right "average" 3 star rating. Such sort of predictions can help mitigate some of the spam in the websites as well.

8. COMPARATIVE ANALYSIS

As a Supervised learning approach has been the solution adaption for this problem and the data-set, KNN and linear SVC have been chosen to perform the classification on the data-set. A clear - cut comparison can be made between the two classifiers that I have chosen to use for this data-set.

- KNN, being a more basic algorithm, logic wise compared to the linear SVC algorithm, evidently linear SVC yielded better results compared to KNN classifier.
- KNN is an algorithm which uses a lazy learning based approach where as the linear SVC algorithm follows an eager learning based approach.
- KNN algorithm uses the euclidean distance measure to be the distance used to choose its neighbors whereas, the linear SVC algorithm uses a hyper-plane based distance separation.
- KNN algorithm, on the given input internally classifies based on plainly the neighbors where as the linear SVC algorithm internally divides the problem into three different models and compares each separately. Each model represents "bad-average", "average-good" and "good-bad" respectively.
- The implementation of KNN algorithm was easier when compared to that of the linear SVC algorithm.
- The accuracy results were higher in the case of linear SVC algorithm when compared to that of the KNN algorithm.
9. FUTURE WORK

The current model developed can be used to predict any value of an unknown class for any application in any domain. Other domains, especially financial domains part of companies where they try to predict a companies’ stocks could also make use of the model currently developed with slight domain adaptations. Apart from KNN[10] and Linear SVC algorithms[5], other classifying algorithms can also fit the solution scope such as the Naive Baye’s algorithm. As per the accuracy of results perspective, it can certainly be improved depending on the type of the data-set used and also using more feature vectors like the usage of the tf-idf weights. This can help determine the significance of words in each of the documents.

10. REFERENCES