Yelp Data Challenge

PRANAB BHADANI[1], HARDIK CHAPANERA[1], RUCHI GUPTA NEEMA[1], AND SNEHAL VARTAK[1]

[1]*School of Informatics, Computing and Engineering, Bloomington, IN 47408, U.S.A.*
[*]*Corresponding authors: rneema@iu.edu, snehchem@iu.edu*

December 13, 2017

This project aims to recommend business to users and to predict number of reviews a business will receive in the future. The recommendation algorithm uses the review data to improve the recommendations. We also evaluate the performance of multiple recommendation algorithms.

Keywords: Recommendation System, Yelp Data, Aspect Mining

https://github.com/snehalvartak/Yelp-Data-Challenge

INTRODUCTION

The purpose of this project is to solve two interesting problems using the Yelp Data. Yelp is the one of most popular websites where users check for reviews before visiting a particular business. But Yelp does not provide recommendations at this point of time. Users have to go through the reviews of each business and decide whether to visit a particular business. The two questions that we attempt to tackle in this project are as follows:

1. Recommend business to users
2. Predict the number of reviews for a business

The approach, methodology and evaluation for each task is detailed in the upcoming sections. The data for this project is provided by Yelp as part of their Yelp Dataset Challenge.

TASK 1: RECOMMEND BUSINESS TO USER

Data Preparation

The Yelp data consists of 1.6M reviews by 366K users for 61K businesses. For the purpose of building recommendation system, we restrict ourselves to restaurants located in Las Vegas which consists of approximately 5K business and 4K users. From here, we went on to remove users who had reviewed less than 24 business and businesses with less than 20 reviews. This the final data set consisting of 4K unique businesses and 3K users that we use for task 1.

Methodology

We implemented two different algorithms for recommending business to users. The first one is based on aspect based opinion mining and the second one is based on collaborative filtering approach. Both the algorithms are discussed in details in the following section.

Recommendations Based on Aspect Opinion Mining

In this algorithm we tried to represent users and business by some feature vectors as shown in figure 1. Once we get feature vectors for different users and different business, we try to find the similarity between each user vector and business vectors, and we will recommend that particular business to user that has got maximum similarity with the given user vector.

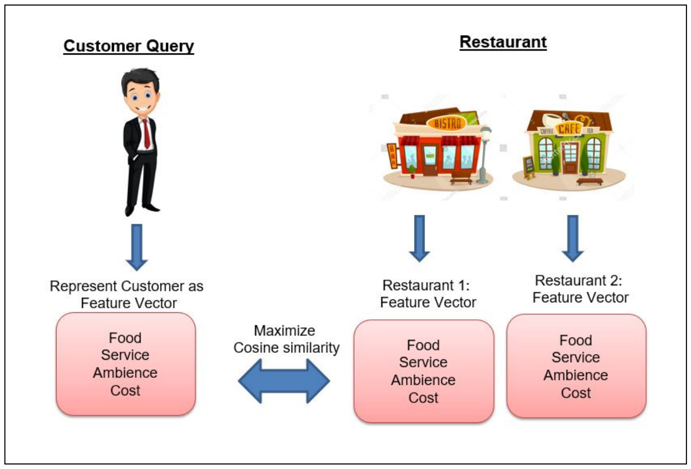
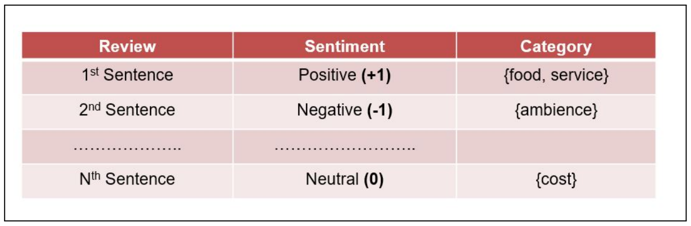

Customer Query
Restaurant
BISTRO
CAFE
Represent Customer as Feature Vector
Food
Service
Ambience
Cost
Maximize Cosine similarity
Restaurant 1: Feature Vector
Food
Service
Ambience
Cost
Restaurant 2: Feature Vector
Food
Service
Ambience
Cost

Fig. 1. Recommendation systems: Our Approach

Feature Extraction

The main task for this algorithm is to define the feature set. We have tried many different methods to extract features from the review data. For example, we tried to tag each of the word in review text with part of speech and find out top 15 most frequent used words with 'Noun' tag and found that people usually talks about food quality, service, ambience and sometimes about the price of food. We also tried to find out the top most named entity in review text and try to used that words as our feature set but we got best results with five features as: food, service, ambience, cost and misc. Therefore we represented each user and each business with these five dimensional feature vectors. The procedure to find out the vectors is explained in the following section in detail.



Review	Sentiment	Category
1st Sentence	Positive **(+1)**	{food, service}
2nd Sentence	Negative **(-1)**	{ambience}
……………….	…………………….	
Nth Sentence	Neutral **(0)**	{cost}

Fig. 2. Feature-based opinion mining

Sentiment analysis for business vector

Our goal is to develop a feature-based opinion mining for yelp reviews. Therefore instead of aggregating positive and negative sentiment for all the reviews, we will calculate sentiment score with respect to feature set for all the reviews and then aggregate it based on feature set. Thus, rather than determining whether a review of a business is, on-the-whole, positive/negative, the aim is to extract user attitudes towards the features such as food,decor and service of the business. These attitudes can then be aggregated across users to give a clear picture of exactly what users like/dislike about a particular restaurant/business and then we can represent a business with a feature vectors based on users reviews.

Now to predict the category of review, we first find the sentiment of each individual sentence of review and then we will predict about which feature the user was talking about in that particular sentence. The algorithm steps are also shown in flowchart in figure 3.

First we divide a single review into multiple sentences and then calculate the sentiment score for each sentences. We have used supervised approach to find out sentiments for individual sentences. We trained our model by using already labelled sentences to get the sentiments of sentences. We labelled each sentence as positive/negative/neutral for sentiment analysis. We gave +1 for positive sentiment, -1 for negative sentiment and 0 for neutral sentiment.

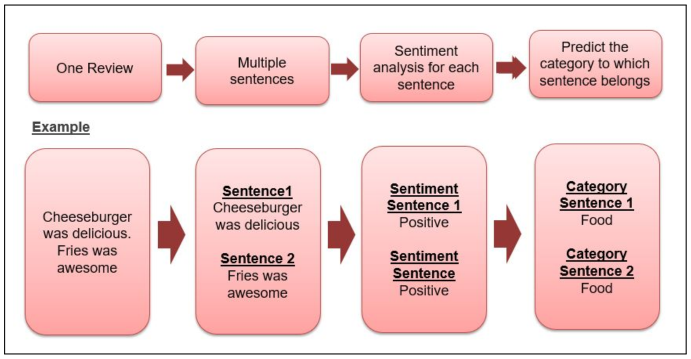

One Review
Multiple sentences
Sentiment analysis for each sentence
Predict the category to which sentence belongs
Example
Cheeseburger was delicious. Fries was awesome
Sentence1 Cheeseburger was delicious
Sentence 2 Fries was awesome
Sentiment Sentence 1 Positive
Sentiment Sentence Positive
Category Sentence 1 Food
Category Sentence 2 Food

Fig. 3. Algorithm steps: Review to feature vector

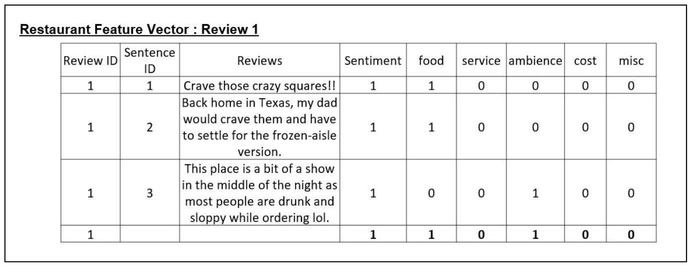
Word Embedding: Aspect prediction for sentence

After getting sentiments for each sentences, we need to find the category to which the sentiment belongs to. For this, we removed stop words for each sentence and try to predict the category about which the user is saying in that particular sentence of review. For this, we tried many different approaches such as we tried to find the subject in each sentence but since these are the review data therefore it is difficult to get the proper subject in each sentence as the review data are not grammatically correct most of the times. We also tried to get the named entity for each sentence but we were getting null named entity for maximum of the sentence therefore we extracted noun words from each sentences and try to predict the category for these words using "Word-Embedding". This approach worked for us and we got good results with this approach.

After extracting 'Noun' words from each of the sentence we try to predict the category to which the noun word belongs to. We used 'Word2Vec' model for this. We trained a model with all the review text. Word2Vec then translates each word into a vector of 200 features. These vectors are located in a feature space of 200-dimensions, with similar words closer together and unrelated words farther apart. For example, the vectors for "hamburger" and "food" should be pointing in nearly the same direction, but the vectors for "hamburger" and "service" would be far apart.Therefore, in this way we can get similarity between each noun word extracted and each of the 5 feature vector and can assign the category for which maximum words belongs to. If a word does not belong to any of the category then it will be assigned category as "misc".The algorithm steps are well depicted in figure 5.

Business table

After predicting category for each of the sentence in each review, we need to aggregate sentiment score for each business. For this we will take sum of all the sentiment score corresponding to one particular review id and then take average of all the sentiment score of reviews for particular business afterwards. In this way, we will get feature vector for each of the business.The feature vector for one particular review id is as shown in following figure 9.



Restaurant Feature Vector : Review 1

Review ID	Sentence ID	Reviews	Sentiment	food	service	ambience	cost	misc
1	1	Crave those crazy squares!!	1	1	0	0	0	0
1	2	Back home in Texas, my dad would crave them and have to settle for the frozen-aisle version.	1	1	0	0	0	0
1	3	This place is a bit of a show in the middle of the night as most people are drunk and sloppy while ordering lol.	1	0	0	1	0	0
1			**1**	**1**	**0**	**1**	**0**	**0**

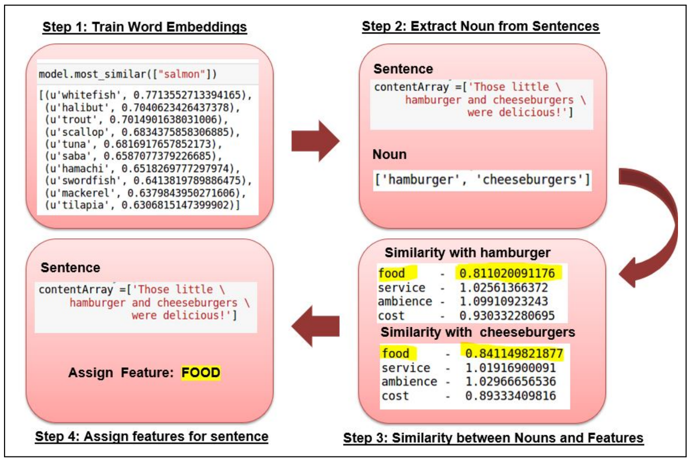
Fig. 4. Business feature vector representation

User table

Once we get the feature vector representation for each of the business, we can get the user(C_k) feature vector by using following equation.

$$C_k = \frac{\sum_{i=1}^{N} w_i * R_i}{\sum_{i=1}^{N} w_i} \quad \textbf{(1)}$$

where N is the total number of restaurants which he visited already and gave w_i rating to i^{th} restaurant and R_i is the feature vector of i_{th} restaurant.


Step 1: Train Word Embeddings
Step 2: Extract Noun from Sentences
Sentence
Noun
Similarity with hamburger
Similarity with cheeseburgers
Sentence
Assign Feature: FOOD
Step 4: Assign features for sentence
Step 3: Similarity between Nouns and Features

```
model.most_similar(["salmon"])

[(u'whitefish', 0.7713552713394165),
 (u'halibut', 0.7040623426437378),
 (u'trout', 0.7014901638031006),
 (u'scallop', 0.6834375858306885),
 (u'tuna', 0.6816917657852173),
 (u'saba', 0.6587077379226685),
 (u'hamachi', 0.6518269777297974),
 (u'swordfish', 0.6413819789886475),
 (u'mackerel', 0.6379843950271606),
 (u'tilapia', 0.6306815147399902)]
```

```
contentArray =['Those little \
     hamburger and cheeseburgers \
                  were delicious!']
```

```
['hamburger', 'cheeseburgers']
```

```
food      -  0.811020091176
service   -  1.02561366372
ambience  -  1.09910923243
cost      -  0.930332280695
```

```
food      -  0.841149821877
service   -  1.01916900091
ambience  -  1.02966656536
cost      -  0.89333409816
```

Fig. 5. Category Prediction for each sentence

Evaluation: Task 1a

Once we get the feature vector for all the user and business we can determine how similar user vector and business vector based on cosine similarity. Mathematically, cosine similarity is the dot product of the two vectors divided by the product of the magnitudes of those two vectors. This calculates the angle between two vectors. The smaller the angle, the more similar the vectors are. Larger angles are more dissimilar and are calculated closer to negative one. Vectors perpendicular to one another will have a cosine similarity of zero.Therefore, based on cosine similarity we can recommend business to user based on its taste.

We evaluated our results using RMSE. We predicted the rating by finding out the cosine similarity value and then normalizing the similarity value in the scale of (1-5) as the lowest rating is 1 as given by user to business while highest rating is 5. We calculated RMSE value for this approach and our RMSE value comes out to be 1.63 which is not that large value.

In the next section, we will discuss collaborative approach to solve this problem. Then we will discuss our evaluation results with different algorithm approaches.

Algorithm 2: Collaborative Filtering Approach

Recommendations based on collaborative filtering take users preference into consideration to predict what other businesses the user may like [2]. It assumes that if user u_1 and user u_2 both like an item b_1 and user u_2 also likes b_2 then user u_1 is more likely to prefer item b_2 as well. For purpose of this task the item that the user might be interested in is a restaurant. This is the key concept behind collaborative filtering. Here we implement different types of collaborative filtering techniques including item based, user based, nearest neighbor and SVD based to recommend restaurant to users.

We have the user behavior data in the reviews file, which contains the restaurants the user has previously rated and /or reviewed. In this approach, we build the user-business matrix from the user behavior data where each $< u_i, b_i >$ represents the rating r_i given by the user u_i to restaurant b_i and use this data for recommendations. Collaborative filtering deals with very sparse data. In our case the data was above 98.7% sparse after removing rare businesses (i.e business with less than 20 reviews) and rare users (i.e users with less than 24 reviews). The data is split in 75 : 25 ratio where we train on 75% of the data and test on the remaining 25% of the data.

- Item based: For item based collaborative filtering we calculate the similarity between items for the users that have rated both the items.

- User based: For user based collaborative filtering we calculate similarity between users for the items that both have rated [1].

- Neighborhood based: This is a memory based CF technique in which a subset of users are chosen based based on either cosine or Pearson similarity and a weighted combination of their rating ratings is used to predict ratings for a user. In item based KNN each item is represented as a vector of user ratings. New ratings are predicted based on the similarity of item i with every other item. Similarly for user based KNN each user is represented as a vector of ratings for restaurants and predictions is done by computing the similarity of this user vector with all other users.

- Model based (SVD): Singular Values Decomposition (SVD) is a model based collaborative filtering approach to predict the rating for a user item pair based on previous rating information. SVD is a matrix factorization technique that factors a $m \times n$ user-item matrix in to three matrices as shown in 6.

$$\underset{m\times n}{\overset{X}{\begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & \\ \vdots & \vdots & \ddots & \\ x_{m1} & & & x_{mn} \end{pmatrix}}} = \underset{m\times r}{\overset{U}{\begin{pmatrix} u_{11} & \cdots & u_{1r} \\ \vdots & \ddots & \\ u_{m1} & & u_{mr} \end{pmatrix}}} \underset{r\times r}{\overset{S}{\begin{pmatrix} s_{11} & 0 & \cdots \\ 0 & \ddots & \\ \vdots & & s_{rr} \end{pmatrix}}} \underset{r\times n}{\overset{V^{\mathsf{T}}}{\begin{pmatrix} v_{11} & \cdots & v_{1n} \\ \vdots & \ddots & \\ v_{r1} & & v_{rn} \end{pmatrix}}}$$

Fig. 6. SVD Matrix Factorization [3].

Results for Task 1

A overall comparison of all the algorithms in task 1 is shown in 7. We can see that model based SVD approach yields the best results for recommendations.

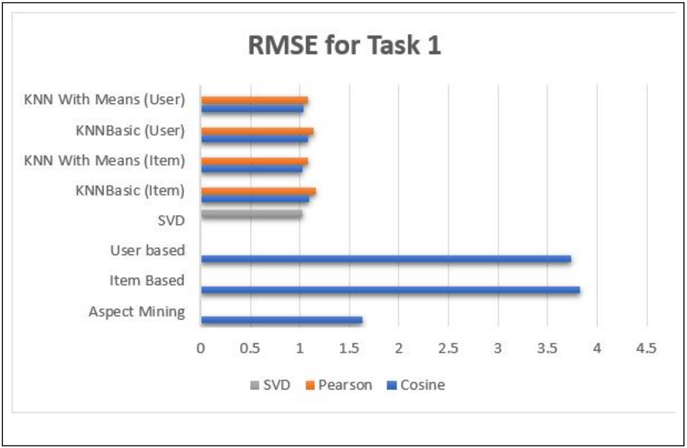
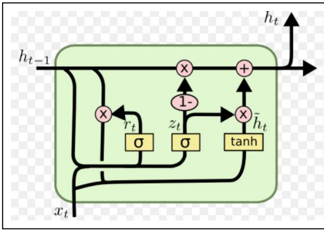


Fig. 7. Results for Task 1

TASK 2: PREDICTION OF NUMBER OF BAD REVIEWS FOR A BUSINESS

The goal of this task is to predict the number of bad reviews a business can get based on the past reviews trend. As we try to model a sequence data for our prediction, we need to take care of the auto-correlation issue that may exist in the data. For this reason, we have chosen three different approaches, namely, deep learning, time series based and machine learning. The first approach uses a Recurrent neural network which can retain memory of the past to predict the current instance. As normal RNN could suffer from a gradient vanishing problem. So we have decided to use LSTM model a variant of Recurrent neural model. Below is the architecture of how LSTM model looks like



$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t])$$
$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t])$$
$$\tilde{h}_t = \tanh(W \cdot [r_t * h_{t-1}, x_t])$$
$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

Fig. 8. LSTM Architecture

Since the number of data points to train the model is not very large for a single business, we don't expect the deep LSTM model to perform well, so we decided to use a statistical time series forecasting method namely Auto Regressive Integrated moving average model(ARIMA). Lastly in machine learning approach we use supervised learning where we have different features like the different attributes of review like usefulness, cool etc, business stars and tf-idf scores of words in review. We use this as features and ratings as output and try to fit different models like Naive Bayes, Linear Regression, Logistic Regression, SVM and Random Forest to get the ratings for the test data

Data Preparation

The main challenge for any model to work efficiently is to provide the model the data in the form it expect to get. So, we had to pre-process the data in order to feed it correctly to the network. Now, since we don't have the past reviews count directly in the data-set, we found the count by count the reviews given by all the user to a particular business per day. Thus we aggregated the count of reviews on Business Id to prepare of sequence. Further we segregated the normal count of reviews to count of bad and good reviews based on the sentiment of the reviews we had calculated in the part of this paper. So, in total we had 2800 reviews on which the model has been built. For machine learning approach the main pre-processing part was to sort the data by date

Methodology

Machine Learning approach

From the reviews data we take attributes of reviews such as usefulness, coolness or how funny the reviews are as features. We also take the rating of the business and the TF-IDFscores of each word in reviews as features. This gives us the impact of each word in the review as feature. Since the dimension of the features were very high for businesses (greater than 5000 dimension) we applied Principal Component Analysis(PCA).[4] PCA is a technique that reduces the dimension by converting the possibly correlated features to a linearly uncorrelated features called principal components. We found out that 3 principal components explained 97.5% of the variance in original data. We took this as input features to train the data and ratings as the class or output and applied different models, namely, Naive Bayes, Linear Regression, Logistic Regression, SVM and Random Forest and tried to fit the data. To evaluate the results we find out how many negative reviews (< 3 stars) exist in actual test data and their ratings. Then we compare that review ratings with which the ML models have predicted and calculate the RMSE.

Recurrent Neural Network Approach

From the data-set prepared as mentioned above, we passed the data to the LSTM network in sequence order, where we have next number of review as the output of the LSTM result.

The Parameters of the LSTM model is as follows:

Hidden Layer : 1 Hidden units : 10 Look Back : 4 epoch : 1000 batch size : 32

ARIMA approach

Since we had limited amount data for a particular business, we can't completely rely on the results of Neural network. So, we decided to build a model based a statistical approach which can give us a good result even in less data. To apply ARIMA model straight to the data, we need to first make sure that the data is stationary. Firstly we applied Dickey Fuller's test to check for the stationary condition. On applying Dickey fuller test we found that the dataset is stationary based on the t-stat value of the Dickey Fuller test. Then we went ahead to apply ARIMA model on the count of the review Negative review data-set to calculate our RMSE value.

0.1. Results

LSTM Results

RMSE value = 1.4614

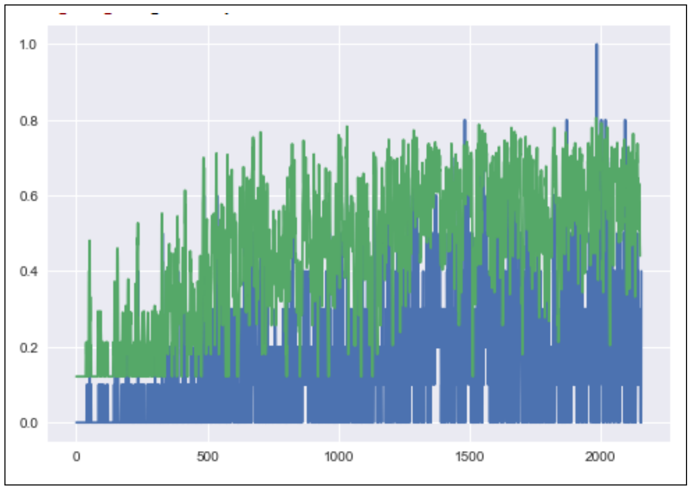


Fig. 9. Green: Predicted Blue: Original

Now, to look for the best parameters of the LSTM model, we tried running the model with various look back and found that look back of 4 gave us the minimum RMSE. Below is the plot justifying our findings of the look back.

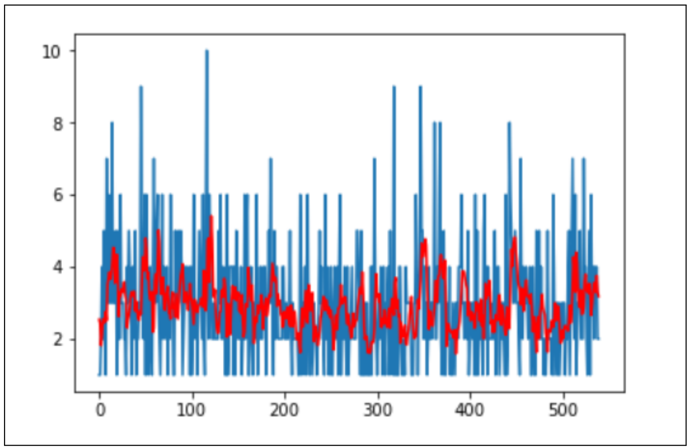


Fig. 10. Negative Review Trend based on ARIMA

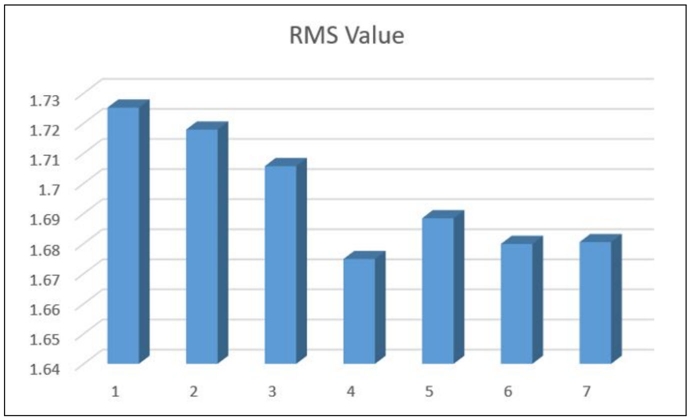
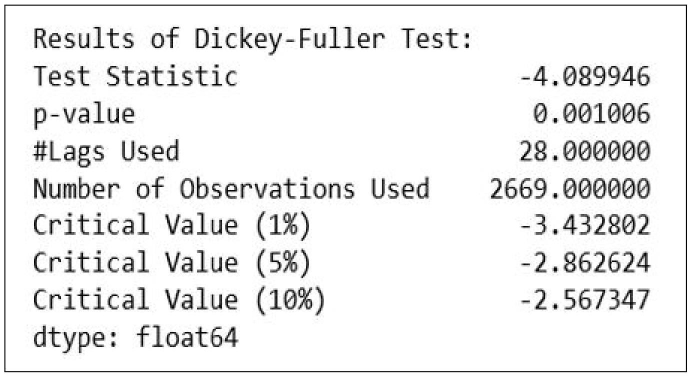


Fig. 11. Plot of RMSE value vs lookback

ARIMA Results

RMSE value: 0.2641 Dickey's Fuller test to justify the stationary condition required to apply ARIMA model on any time series data.



```
Results of Dickey-Fuller Test:
Test Statistic                   -4.089946
p-value                           0.001006
#Lags Used                       28.000000
Number of Observations Used    2669.000000
Critical Value (1%)              -3.432802
Critical Value (5%)              -2.862624
Critical Value (10%)             -2.567347
dtype: float64
```

Fig. 12. Dickey Fuller Result

We found that the value of t-stat is less than the critical value at all the confidence interval, thus we can apply ARIMA model without incurring any bias to the results.

ML Results

RMSE graphs for various Machine Learning models is given in figure 13

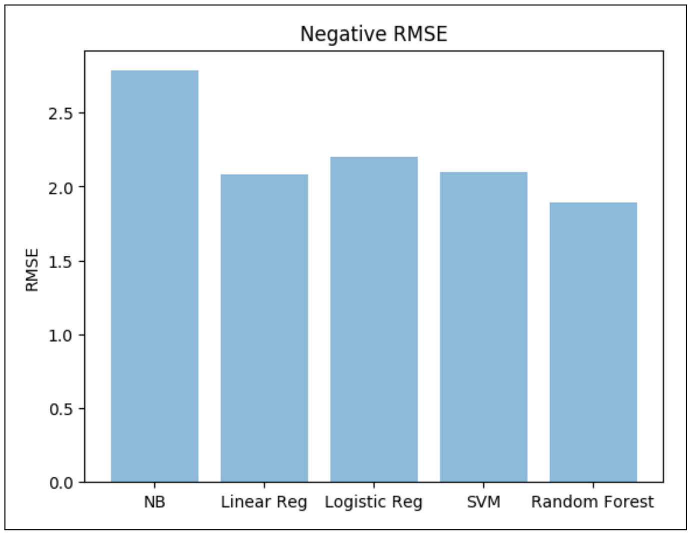


Fig. 13. RMSE for negative reviews ML approach

CONCLUSION

Task 1

As we see that we got RMSE score of 1.63 for aspect based opinion mining which is a quite a good score. Therefore we can use this method to recommend business to user. Also overall we see that KNN and SVD approached have a much lower RMSE compared to item-item and user based collaborative filtering approach. The results also show that matrix factorization approach yields much better results compared to memory based approaches.

Task 2

On seeing the results and comparing with the two approaches i.e neural network approach and ARIMA approach, we found that although the we have less data to train but still LSTM approach outperforms traditional ARIMA model. The reason which we believe is because of less number of predictors to predict the dependent variables.

FUTURE WORK

Task 1

There has been a lot of work in the area of aspect based opinion mining but there isn't sufficient labelled review data to effectively use it in a recommender system. The aspect opinion mining based can be explored more by using labelled reviews to improve the performance of this type of recommendation system.

Task 2

For future work we would like to add other features like sentiment trend across time for a business and other external factors that varies across time. We believe that these extra factors could be useful enough to predict the negative reviews for a business, which could further help us reduce the RMSE.

REFERENCES

[1] Su, X. and Khoshgoftaar, T. (2017). A Survey of Collaborative Filtering Techniques. [online]
[2] https://en.wikipedia.org/wiki/Collaborative_filtering
[3] Cs.carleton.edu. (2017). Algorithms :: Single Value Decomposition.[online]
Available at: http://www.cs.carleton.edu/cs_comps/0607/recommend/recommender/svd.html
[4] http://www.bioinf.jku.at/publications/older/2604.pdf
[5] http://ijssst.info/Vol-15/No-4/data/4923a105.pdf
[6] http://onlinelibrary.wiley.com/doi/10.1002/for.3980020104/pdf

TEAM MEMBER CONTRIBUTIONS

Name	Contributions
Pranab Bhadani	• Project-Idea • Task-1:Sentiment analysis • LSTM for Task-2 • Documentation • Presentation
Hardik Chapanera	• Project-Idea • Task-2:ML approach • Collaborative filtering • Documentation • Presentation
Snehal Vartak	• Project-Idea • EDA • Task-1 • Collaborative filtering • Documentation • Presentation
Ruchi Neema	• Project-Idea • Task-1 • Word Embedding • Documentation • Presentation.

Table 1. Contribution