# Yelp Recommendation System and Sentiment Analysis

Shrikant Mudholkar, Varsha Bhanushali, Monas Bhar

**Abstract**

*In the era of Data Science and Social Computing, the role of customer reviews and ratings can be instrumental in predicting the success and sustainability of businesses. In this paper, we have used Yelp data, to build a restaurant recommendation system for individuals and groups. We have extracted collaborative and content-based features to identify customer and restaurant profiles. In particular, we implement Alternating Least Square, an algorithm to perform Collaborative filtering, to create a recommendation engine using geospatial location for users. We created a flask application where existing user can be recommended restaurants based on their previous reviews and new users can be add their preference to get recommendations. We have created a model which would predict the rating based on their reviews 89% accurately. Sentiment analysis of customer reviews has a crucial impact on a business's development strategy. We have performed sentiment analysis using various deep learning techniques to predict the rating given by users based on their reviews. We applied Latent Dirichlet Allocation(LDA) algorithm of topic modeling to further classify the restaurants based on the topics the reviewers discuss about. We get the top 15 topics of each business based on the patterns of corpuses. We have performed Sentiment analysis using Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) to determine the polarity of the reviews given by users.*

## I. Introduction

A vast database of reviews, ratings, and general information provided by the community about businesses, Yelp provides consumers with a myriad of options and information even when searching for an especially specific service or goods niche. However, although all required information may be present to make an informed choice, it is often still difficult by just looking at the raw data. Reading all the reviews of a single business alone is time consuming and requires more effort than the average user is willing to expend. As a result, we believe users could greatly benefit from a recommendation system and a sentiment analysis would benefit the businesses to understand the reviews of their customers.

Inspired by our recent experience searching Yelp for restaurant, we built a restaurant recommendation system. Recommendation systems provide personalized, relevant recommendations to users and have been used in various domains, such as retail, movie-going. Currently, Yelp, the leading publisher of reviews of local businesses in the world, does not provide recommendations. Instead, users should filter, sort, then read reviews to determine whether a business can provide them with what they want. A personalized recommendation system will provide a better user experience by incentivizing users to review and rate more in return for better restaurant recommendations; this in turn gives Yelp more data that can be used to further improve the recommendation system.

Businesses often want to know how customers think about the quality of their services to improve and make more profits. Restaurant goers may want to learn from others' experience using a variety of criteria such as food quality, service, ambience, discounts and worthiness. Yelp users may post their reviews and ratings on businesses and services or simply express their thoughts on other reviews. Bad (negative) reviews from one's perspective may influence potential customers in making decisions, e.g., a potential customer may cancel a service and persuade other do the same.

.

## II. Data and Methods

### A. Data Description

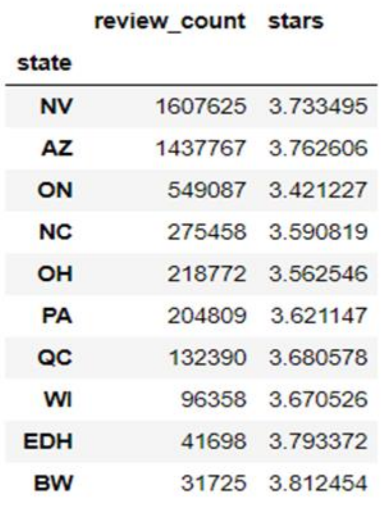
The dataset was originated from the online Yelp Dataset Challenge, consisting of five parts which provides us with 566,000 basic business information (e.g., hours, address, ambience), 2.2 million customer reviews as well as 519,000 tips by 552,000 users. The total size of data is about 2.39GB. For this analysis, we focused on reviews for restaurants and used the customer reviews and business attributes data. These two datasets are both in JSON format. After filtering restaurants out of all business, there were 1,363,242 customer reviews collected from 77,445 different restaurants. Most of those restaurants are in Arizona, Nevada and North Carolina and the dataset includes a huge variety of cuisine types

Attributes in review data include business id, full address, price range, business categories etc. The attributes we used were business id, business categories, review content and rating. Specifically, review content was the corpus for our analysis; rating was the identifier for discriminating positive or negative sentiment; business id served as the key for data munging and the business categories served as the key for grouping.

## *B. Data Cleaning*

The Dataset originally consisted of different categories of business. We clean and filtered the data by cherry picking the categories related to restaurants and included only those business which are related to restaurant category. Further we calculated the mean stars and evaluated the top states with most review count and mean stars.

We eliminated the states with significantly low number of review count and stars. We segregated and stored the data in different files with the respective to states. We also performed join on the three-different dataset of users, review and business to form a combined file consisting of users related to a business with the user review data.



| state | review_count | stars |
|---|---|---|
| NV | 1607625 | 3.733495 |
| AZ | 1437767 | 3.762606 |
| ON | 549087 | 3.421227 |
| NC | 275458 | 3.590819 |
| OH | 218772 | 3.562546 |
| PA | 204809 | 3.621147 |
| QC | 132390 | 3.680578 |
| WI | 96358 | 3.670526 |
| EDH | 41698 | 3.793372 |
| BW | 31725 | 3.812454 |

*Figure 1: Total review count and average stars grouped by state*

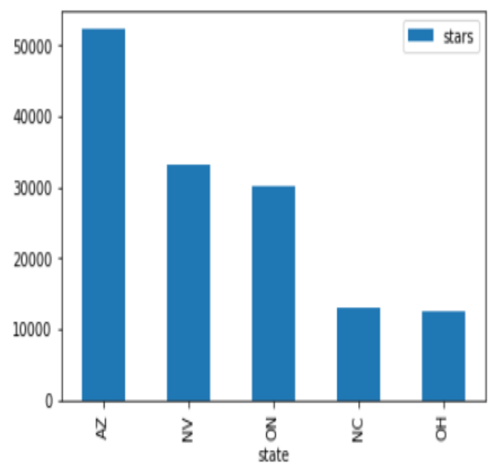


*Figure 1: Top 5 state selected for further analysis*

## *C. Code with Documentation*

[https://github.com/shrican/yelp-recommendations-sentimental-analysis/tree/master/src/scripts](https://github.com/shrican/yelp-recommendations-sentimental-analysis/tree/master/src/scripts)

## *D. Methods and Algorithms*

### TASK DEFINITION

We use data provided by Yelp, described further in the Data section, to build a recommendation system that provides personalized restaurant recommendations to users. Since different people have different food preferences and dietary restrictions, we perform careful feature selection to take advantage of the information reflected in a Yelp user’s reviews.

### COLLABORATIVE FILTERING

Collaborative filtering (CF) is commonly used for recommender systems. These techniques aim to predict user interests by collecting preferences or taste information from many users. In other words, CF fills in the missing entries of a user-item association matrix. The underlying assumption is that if person A agrees with person B on one issue, A is more likely to have B's opinion on another issue than that of a randomly chosen person.

Mathematically, this is done by low-rank matrix factorization, combined with a minimization problem (see picture below). The often-sparse user-item rating matrix R is approximated as a product of user matrix U and item matrix PT, which are built of latent factors. We then form the cost function J and try to minimize it. Currently in the spark.ml library, the alternating least squares (ALS) algorithm has been implemented to learn these latent factors. Additionally, since we directly rely on the user rating itself, our approach is often referred as "explicit.".

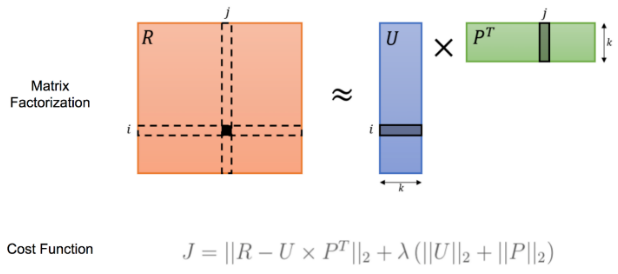


*Figure 2: Cost function for ALS*

For our project, we pre-train the model and save it. When the recommendation engine boots up, it will load the model and use it for prediction. This architecture is designed so that we can keep training multiple models offline as new data comes in. Once a new model is ready, the recommender engine will make the switch by editing one line of code.

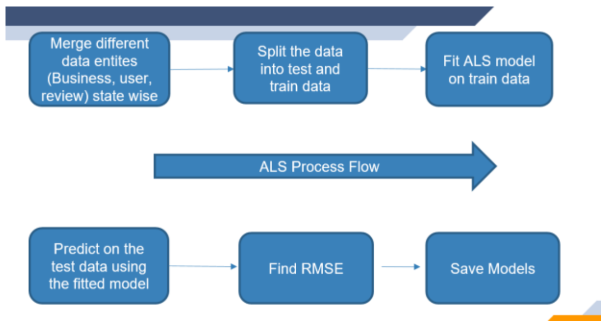
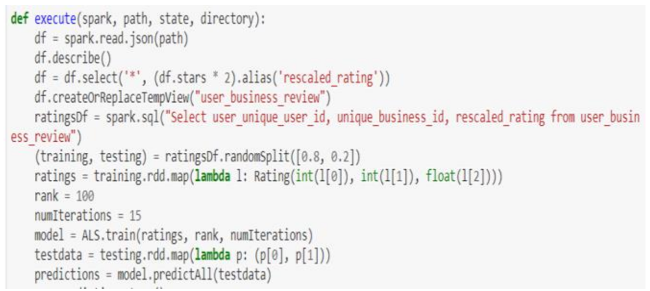

Merge different data entites (Business, user, review) state wise
Split the data into test and train data
Fit ALS model on train data
ALS Process Flow
Predict on the test data using the fitted model
Find RMSE
Save Models

*Figure 3: Process of Collaborative filtering*

We used Spark ML to build the ALS Matrix Factorization Model for restaurant recommendation. We have created for different ALS models (one per state) which we pickle and store in the file system after creation. This model can help us get restaurant recommendations for an existing user as well recommendation for users when given a restaurant.

For each state, we load the JSON file containing a combined view of user reviews and restaurants. We select user_id, business_id, and rating and do an 80:20 random split on the data. The training data is used to train the ALS model and the testing data is used to compute the RMSE. The models are pickled and saved so that the flask application can consume it.



```
def execute(spark, path, state, directory):
    df = spark.read.json(path)
    df.describe()
    df = df.select('*', (df.stars * 2).alias('rescaled_rating'))
    df.createOrReplaceTempView("user_business_review")
    ratingsDf = spark.sql("Select user_unique_user_id, unique_business_id, rescaled_rating from user_busin
ess_review")
    (training, testing) = ratingsDf.randomSplit([0.8, 0.2])
    ratings = training.rdd.map(lambda l: Rating(int(l[0]), int(l[1]), float(l[2])))
    rank = 100
    numIterations = 15
    model = ALS.train(ratings, rank, numIterations)
    testdata = testing.rdd.map(lambda p: (p[0], p[1]))
    predictions = model.predictAll(testdata)
```

*Figure 5: Code snippet for creating test and train data*

## SENTIMENT ANALYSIS

### *Topic Modelling*

Sentiment Analysis, also known as opinion mining, is the process of determining whether a text unit is positive or negative. It can have a wide range of applications such as automatically detecting feedback towards products, news and characters or improving customers' relation model.

Various approaches have been used to evaluate the sentiment underneath the words and expressions or documents. Some of the most common machine learning algorithms used in NLP fields include Naive Bayes (NB), Maximum Entropy (ME), Support Vector Machine (SVM) (Joachims, 1998), and unsupervised learning (Turney, 2002).

The overall sentiment polarity showed a preference on service in the reviews, which might allude that customers 'self-select' the food they like. In the other hand, we could mine many valuable insights that cannot be directly revealed on the dashboard of websites like Yelp. Yelp's dashboard merely shows an overall rating towards a business rather than several ratings for various aspects of businesses, while we considered it at a more granular word-level.

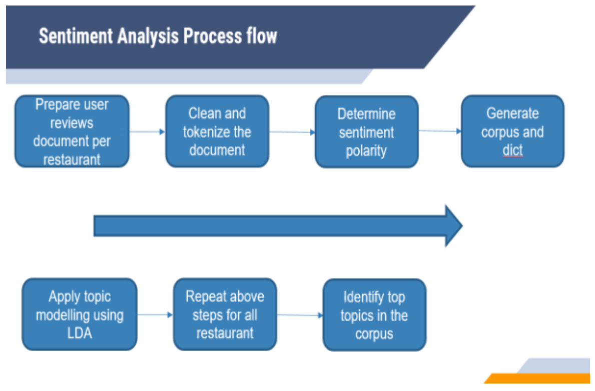

Sentiment Analysis Process flow
Prepare user reviews document per restaurant
Clean and tokenize the document
Determine sentiment polarity
Generate corpus and dict
Apply topic modelling using LDA
Repeat above steps for all restaurant
Identify top topics in the corpus

*Figure 4: Sentiment Analysis process flow*

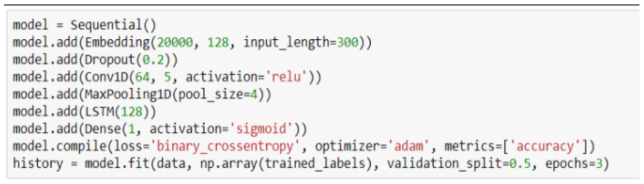
### *Steps for topic modelling:*

1. Combine all reviews of a single restaurant into a single document
2. The entire document is tokenized and cleaned using stop words provided in the Gensim library.
3. We then used Textblob on this cleaned text to determine sentiment polarity
4. We then used Gensim library for first creating a dictionary followed by corpus by using the cleaned text
5. Once the corpus and dictionary are created, we used an unsupervised learning technique called LDA for topic modelling
6. We repeat this process for all the restaurants across all the states

7. We then use LDA model to identify the top topics in the corpus and store the top 15 tokens in each topic in a dictionary which stored along with the calculated polarity of the reviews and other details

## DEEP LEARNING

The merged review-business data were randomly separated into training, and testing set according to ratio 5:5. Specifically, we assumed and labeled reviews with ratings greater or equal to 4 as positive while the rest as “negative”. This decision was made based on our observation of the distribution of ratings.



```
model = Sequential()
model.add(Embedding(20000, 128, input_length=300))
model.add(Dropout(0.2))
model.add(Conv1D(64, 5, activation='relu'))
model.add(MaxPooling1D(pool_size=4))
model.add(LSTM(128))
model.add(Dense(1, activation='sigmoid'))
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
history = model.fit(data, np.array(trained_labels), validation_split=0.5, epochs=3)
```

*Figure 5: Code snippet for building neural network*

The network starts with an embedding layer. The layer lets the system expand each token to a more massive vector, allowing the network to represent a word in a meaningful way. We used Tokenizer to vectorize the text and convert it into sequence of integers after restricting the tokenizer to use only top most common 20000 words. The model used pad sequences to convert the sequences into 2-D array.

Accuracy and validation accuracy are similar for the epochs. LSTM outperforms the other models when we want our model to learn from long term dependencies. LSTM’s ability to forget, remember and update the information pushes it one step ahead of RNNs.

***Steps for sentiment analysis using RNN:***

1. Load all the reviews, parses them into JSON, and stores them in a list
2. Take a sample of the Yelp reviews which contains the same amount of positive (four or five-star reviews) and negative (one, two, or three-star reviews)
3. Tokenizing the texts using Keras
4. Build a neural network by adding layers of RNN and LSTM
5. Train the model with our neural network

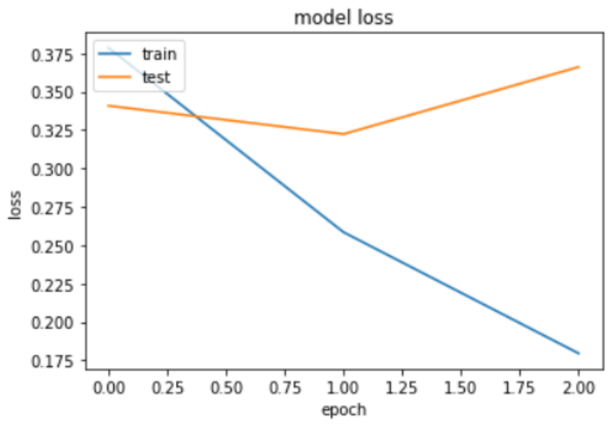


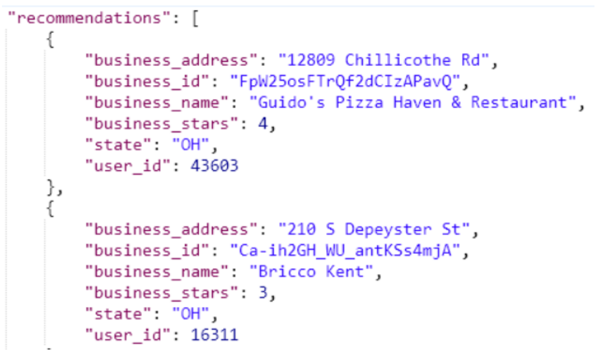
*Figure 8: Change in network plateau*

## III. RESULT

### 3.1. Restaurant recommendation using Collaborative filtering

We tried out various recommendation models for recommending restaurants to a user. With SVD, the RMSE of the model wasn’t as per standards and the model was taking more time that expected to train.

Then we moved ahead with Collaborative filtering. Based on the ALS model, we have trained our data against various user preferences and taste choices. If the user is already registered we recommend restaurants to user based on their past preferences and if the user is new to the system, we determine the restaurants based on their restaurant categories and business choice. When a new user comes into the picture, we take its preferences by giving it few options to choose from, and then we recommend them restaurants that would match their preference.



```
"recommendations": [
    {
        "business_address": "12809 Chillicothe Rd",
        "business_id": "FpW25osFTrQf2dCIzAPavQ",
        "business_name": "Guido's Pizza Haven & Restaurant",
        "business_stars": 4,
        "state": "OH",
        "user_id": 43603
    },
    {
        "business_address": "210 S Depeyster St",
        "business_id": "Ca-ih2GH_WU_antKSs4mjA",
        "business_name": "Bricco Kent",
        "business_stars": 3,
        "state": "OH",
        "user_id": 16311
```

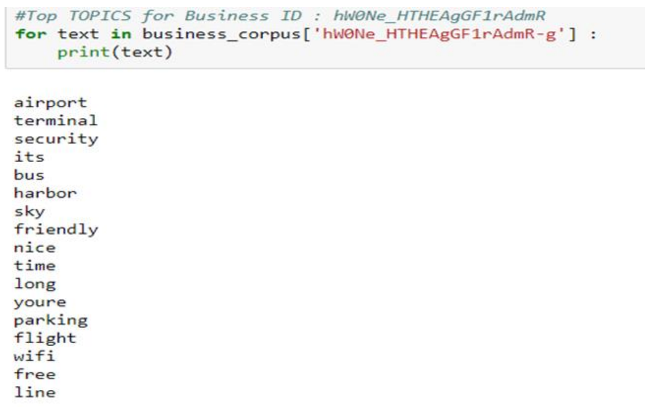
*Figure 9: Restaurant recommended for a user*

Currently, we created a flask application which is a REST application for getting the recommendation. This application needs to be called from any REST client like postman. In future, we can create a web application which would be user friendly for inputs.

### 3.2. Sentiment Analysis using Topic Modelling

Our dataset is modelled to determine the top topics for a business id. LDA model works with the DT matrix along with the Gensim module to infer top topics from the data. We removed the stop words and punctuations to get a better result from our model. Hence, when we pass the business id to our dictionary, it returns all the top 15 topics along with the polarity (if it's positive or a negative topic) discussed about that business.

In future, we can retrieve all the reviews based on a topic for a business. It will be easier for a user to classify the sentiment about the restaurant in each review.



```
#Top TOPICS for Business ID : hW0Ne_HTHEAgGF1rAdmR
for text in business_corpus['hW0Ne_HTHEAgGF1rAdmR-g'] :
    print(text)
```

```
airport
terminal
security
its
bus
harbor
sky
friendly
nice
time
long
youre
parking
flight
wifi
free
line
```

*Figure 10: Checking the output we get from various types of texts*

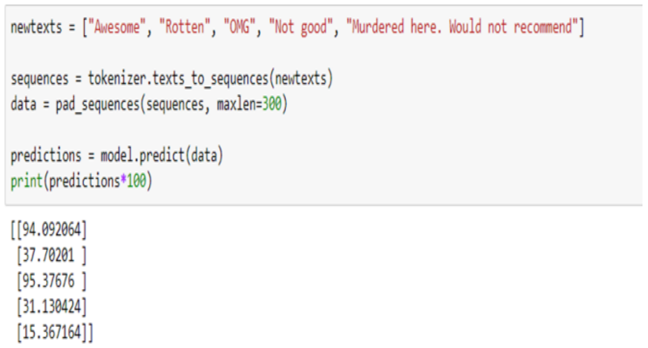
### 3.3. Sentiment Analysis using Deep Learning

We have trained our review dataset to predict the polarity of the new data phrases or the test data. It displays the polarity in terms of percentage. We took 100,000 reviews, half positive and half negative and 3 epochs.

We can see that for the first two epochs, the acc and val_acc numbers are similar. It means that the rules that the network learns on the training data generalize well to the unseen validation data. After two epochs, our network can predict whether a review is positive or negative correctly 85.6 percent of the time.

After the third epoch, the network is starting to memorize the training examples, with rules that are too specific. We can see that it gets 93% accuracy on the training data, but only 85.9% on the held-out validation data. This means that our network has "overfitted", and hence we need to retrain it for only two epochs instead of three.

Additionally, we have trained our model with enough sarcastic inputs to identify the sarcastic review as well and predict the accurate polarity (100 being the most positive word and 0 being the most negative word). We can see below that when we pass texts like "Awesome" and "OMG", we get a polarity of above 90%, which is accurate. When we pass a sarcastic comment, we see that the polarity is below 20%.



```
newtexts = ["Awesome", "Rotten", "OMG", "Not good", "Murdered here. Would not recommend"]

sequences = tokenizer.texts_to_sequences(newtexts)
data = pad_sequences(sequences, maxlen=300)

predictions = model.predict(data)
print(predictions*100)
```

```
[[94.092064]
 [37.70201 ]
 [95.37676 ]
 [31.130424]
 [15.367164]]
```

*Figure 11: Checking the output we get from various types of texts*

## IV. DISCUSSION

We tried out various recommendation models for recommending restaurants to user. Prior to ALS, we tried SVD Model. The RMSE score of this model was not as per recommended standards. Also, it took more time to train the model. As per our observation, SGD is not practical if the dataset size is huge, instead ALS is better.

LDA model and train it on Document-Term matrix. The training also requires few parameters as input. The gensim module allows both LDA model estimation from a training corpus and inference of topic distribution on new, unseen documents.

LSTM outperforms the other models when we want our model to learn from long term dependencies. LSTM's ability to forget, remember and update the information pushes it one step ahead of RNNs.

## V. CONCLUSION

Our hotel recommendation model ALS model had a good RMSE of 5.1. The Recommendation system has an accuracy of 89%. Sentiment Analysis of the yelp dataset using LSTM model (based on RNN) gives the accuracy of 89% in 2 epochs. It also accurately predicts the polarity of the reviews of the users. It also determines accurate polarity for sarcastic reviews. Sentiment Analysis of the yelp dataset using LDA determines the top domains related to the individual business excluding the stop words.

## V. REFERENCES

[1] dos Santos, C. N., & Gatti, M. (2014). Deep Convolutional Neural Networks for Sentiment Analysis of Short Texts

[2] Huang, J., Rogers, S., & Joo, E. (2014). Improving restaurants by extracting subtopics from yelp reviews. iConference 2014 (Social Media Expo).

[3] M. Hoffman and D. Blei. " Online Learning for Latent Dirichlet Allocation." Neural Information Processing

Systems, 2010.
[4] Victor Köhler, https://medium.com/radon-dev/als-implicit-collaborative-filtering-5ed653ba39fe, 2017.
[5] Kaushik, Chetan, and Atul Mishra, "A scalable, lexicon-based technique for sentiment analysis." arXiv preprint arXiv:1410.2265 (2014).
[6] Sahayak, Varsha, Vijaya Shete, and Apashabi Pathan,"Sentiment Analysis on Twitter Data." International Journal of Innovative Research in Advanced Engineering (IJIRAE) Issue 1 (2015).