Topic Modelling Based Recommender system

Siddharth Kanojiya, Keerthana Jayaprakash, Sneha Bezawada

CMPT 733 Big Data, Spring 2018, Simon Fraser University

1 MOTIVATION AND BACKGROUND

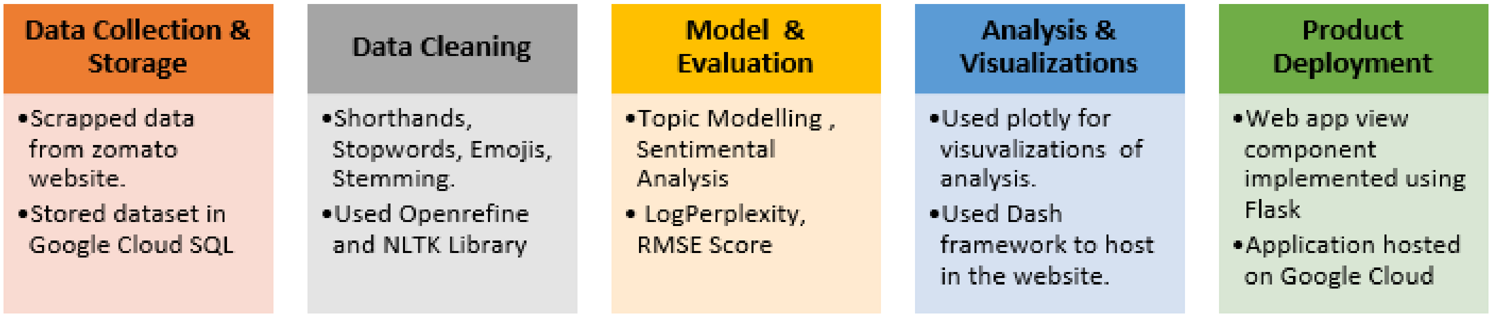
Imagine you are very hungry, its getting too late, and want to visit a fast food place. So you go to a restaurant search portal, say Zomato.com[12], and what do you see?, too many reviews to read, countless categories of restaurants and never ending search filters to apply; eventually you give up and pick some restaurant even if you are unsure. Wouldn't it be better if someone remembers what you like, dislike, and accordingly suggest restaurants? This question is the motivation for our project. We choose Zomato.com because it does not have a personalized recommendation system for its users and has a large collection of user reviews. A personalized recommendation system would make things simpler for the user by providing the user with a list of restaurants that may be closely aligned with their palette. The greatest impact of the proposed recommendation engine will be enhanced user experience. Other significant impact is increased business for the restaurants. There are proven applications like Netflix and Amazon who have gained significant revenues using a recommendation system. This application will be an add on feature to the existing Zomato app.

2 PROBLEM STATEMENT

Large data and vast choices make users confused and finding a restaurant that the user might like, a tedious and time-consuming process. Traditional recommendation systems that only use the user ratings to recommend items. These systems do not consider of other vital information like the ambience, the way a restaurant functioned etc. which cumulatively form the ratings. To identify better similarities and likeness among users, the user review text provides us with this vital information. Our goal is to build a personalized recommendation system for Zomato to help the users find their potentially favourite restaurants based on their ratings and reviews. It also helps to find trends and answers for the questions.

1. Which are the most suited restaurants for a user that he can visit next?
   Identify the user-restaurant similarity based on the user reviews and ratings using topic modelling based collaborative filtering.
2. What are the things people really like and dislike in Greater Vancouver area when it comes to food and restaurants?
   Perform sentimental analysis on the reviews to find both the good and bad that the user writes about the restaurant and how it correlates with the ratings.
3. How busy are the restaurants in Vancouver?
   Identify the restaurant serving hours throughout the week.
4. Are only international cuisines ruling the restaurant market?
   Identify the cuisines that have most footfalls. Identify the cuisines that are highly rated by the users.
5. If someone wants to start a new restaurant in Vancouver, which cuisines will attract more customers.
   Identify the most popular restaurants in Vancouver and find the type of cuisines they serve.

3 DATA PIPELINE


Data Collection & Storage
•Scrapped data from zomato website.
•Stored dataset in Google Cloud SQL
Data Cleaning
•Shorthands, Stopwords, Emojis, Stemming.
•Used Openrefine and NLTK Library
Model & Evaluation
•Topic Modelling , Sentimental Analysis
• LogPerplexity, RMSE Score
Analysis & Visualizations
•Used plotly for visuvalizations of analysis.
•Used Dash framework to host in the website.
Product Deployment
•Web app view component implemented using Flask
•Application hosted on Google Cloud

3.1 Data Collection Storage

Zomato APIs to scrape data was very limited (5 reviews for a restaurant) and hence we need to find an alternative way to collect the data. We scrapped 5000+ restaurant profile using webscrapper.io. Restaurant profile dataset contains restaurant information such as name, address, restaurant rating, opening hours, average cost for 2, restaurant type, cuisines offered, extra features availability like WIFI, delivery takeout, parking etc. We also wrote a python selenium code to scrape the restaurant reviews that contains user ID, restaurant name, user rating and review text. We collected 100K reviews for all the 5000+ restaurants. The data are collected in a .csv file and imported into google cloud SQL.

We also collected restaurant inspection dataset from Vancouver coast health department to perform analysis on the restaurant closure and the reasons for the closure. The dataset was in a .pdf extension and we converted into .csv file and stored it in the google cloud SQL.

3.2 Data Cleaning

For most of the cleaning on the **restaurants** data we used OpenRefine[1], an open source data wrangler by Google. For **reviews** data, being natural text we mainly had to clean/ translate raw text into measurable text form. In order to protect personally identifiable information, we masked usernames. For **inspections** data, which were PDF files, we first copied them to MS Excel and wrote a Visual Basic Macro script[7] to get the content structured as a table. Overview of all the cleaning activities are as follows:

Operation	**Examples**	**Tools used**
Text Parsing	’[”opening-hours”:”Mon11:30AM to 9:30PMTue11:30AM to...Sun12Noon to 9:30PM”]’ to [Mon-Sun]day closed boolean fields	OpenRefine[1], Pandas
Resolve postal code	49.1183738000,-122.8901053000 to V3X 3K1	OpenRefine and Google Maps API[2]
Translate shorthands, smileys and emoticons	”2D4” to ”TO DIE FOR”, ☹ to SAD	NLTK [4] and Pandas[6]
Convert semi-structured text to tabular format	-	Excel VBA Macro[7]
Mask usernames	Johnsmith to USER0067	Pandas
Remove stopwords	this,that,is etc	Python NLTK
Stemming	[bad, badly] to ’bad’	Python NLTK

Table 1: Cleaning activities

3.3 Data Analysis and Modelling

3.3.1 Recommendation Model

Standard Collaborative Filtering (CF) algorithms make use of interactions between users and items in the form of implicit or explicit ratings alone for generating recommendations. Similarity among users or items is calculated purely based on rating overlap in this case, without considering explicit properties of users or items involved, limiting their applicability in domains with very sparse rating spaces. In Zomato there is considerable amount of contextual data available for restaurants in the form of user reviews which could be utilized to improve recommendation quality. Our idea is to learn the latent features of users and items through topic modeling. Combining latent space based similarity with rating overlap-based similarity, we proposed a hybrid similarity score to refine the neighborhood formation, which helps in alleviating sparsity problem as it allows calculation of similarity between users even if they do not have any overlapping ratings.

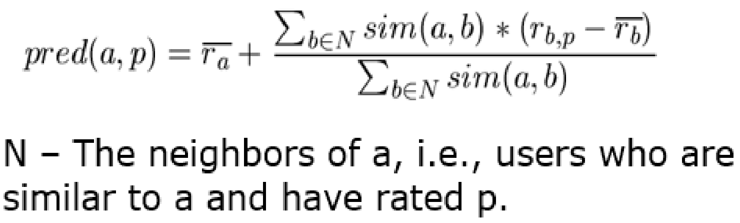
Tools Used: We used Spark Mlib libraries to implement LDA. We opted for this because Spark provides high scalability and easy integration of other tools and libraries for large scale text processing. We used to Python Pandas, Numpy libraries to build the recommendation model as it provides wide range of functions for creating and manipulating matrices(Similarity matrices).

Proposed Approach:

- Load the restaurant reviews data into a dataframe and aggregate all the processed reviews by restaurant ID. This will be the restaurant ID, reviews dataframe will be our item document corpus.
- Perform LDA on the corpus and generate the restaurant topic probability distribution.
- For each user add up item-topic distributions multiplied by normalized user rating, corresponding to each users interests, to generate each users topic-distribution vector.
- Calculate the similarity between a pair users in this latent topic space using euclidean distance metric. Euclidean distance measure is more suitable measure to find distance between two probability distributions.
- Calculate the rating overlap based similarity using standard CF algorithm approaches like cosine similarity.
- The hybrid similarity score is calculated by adding the latent topic space similarity and rating overlap based similarity. They are then scaled so the value lies between 0 and 1.
- We use this hybrid similarity measure to predict the rating of users from prediction formula(Figure 2) of User based CF algorithm.
- From the restaurant which the user hasnt rated, pick the top 10 restaurants with highest predicted rating.

Model Evaluation:

LDA Model:Spark Mlibs LDA model takes two main parameters. The number of topics K and number of iterations maxIter. To find the ideal values of K and maxIter we performed parameter tuning. We divided the data into training and test sets. Built the LDA model on training set for different combinations of K and maxIter, and calculated the logPerplexity score on the test set for each combination. We found the lowest logPerplexity score for K = 10 and maxIter = 40.



$$pred(a,p) = \overline{r_a} + \frac{\sum_{b \in N} sim(a,b) * (r_{b,p} - \overline{r_b})}{\sum_{b \in N} sim(a,b)}$$

N – The neighbors of a, i.e., users who are similar to a and have rated p.

Figure 1: Prediction Formula

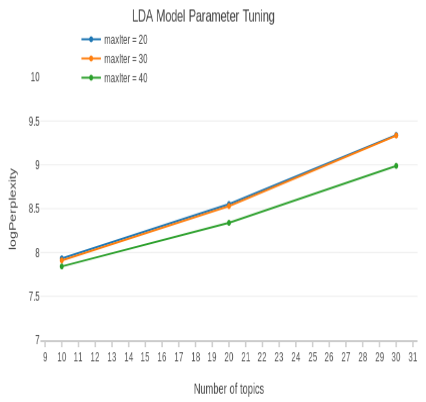


Figure 2: LDA Evaluation

Recommendation model:We randomly split the restaurant user ratings data into training and test set. Built the recommendation model on training set and evaluated the the model on the basis of RMSE score on test set. We compared our model against User based and Item based CF recommendation models and found that our model performs better than them. All the three models were built from scratch without the use of packages. We couldnt find any packages that implemented standard User and Item based CF algorithms. Also, since our main idea was to incorporate reviews into the recommendation model, It was difficult to achieve this with the use of packages which require the input data to be in a particular format.

Model	RMSE
LDA Based CF	3.636
User Based CF	3.647
Item Based CF	3.650

3.3.2 Exploratory Data Analysis

Performed EDA to find the trends and patterns of the data and present it in a meaningful manner. We used Plotly to plot the graphs and hosted the graphs in the webpage using dash framework. The dash framework is then integrated in flask along with the recommendation web application. Our graphs help us to find deep insights that helps Zomato and restaurants to improve their business and helps new entrepreneurs to rightly choose the location, type of restaurant, cuisines that are trending and popular among Vancouver localities. Following are few insights we gathered from our data.

- People are happy with Fine Dining restaurants because despite being costly they are comparatively rated high.
- After fine dining, Bar and Lounge are expensive restaurant type with average cost ranging between 65 to 85 CAD for lunch and 65 to 130 CAD for dinner.
- As non-Asian cuisines are rated high, but they are few in numbers, new restaurants offering those cuisines can make good business.
- Monday and Sunday seem to be off day for most casual dining restaurant. This might be because Fridays and Saturdays are the busiest days.
- Customers negative feedback are mostly on the food, service and owner.
- 2016 had the highest number of restaurant closures compared to 2015 and 2017. More than 20 restaurants are closed in month of October in 2016.

User Behaviour Analysis:This analysis was carried out on one specific user to find out what he/she like based on the reviews he has given in the past. For this we first identified the users the most number of reviews and picked a user with neutral rating distribution i.e whose mean score was between 3 and 3.5. We then analyzed the most frequent bigrams, frequent adjectives used in his positive and negative reviews. Interestingly we found that this user seems like eating at food trucks, had bad experience with chinese and indian food and doesnt like dry, soggy and hard food.
Inspection Data Analysis:Our main aim with this analysis was to analyze the factors leading to restaurant closures and help new restaurant owners to invest wisely in resources to prevent this. We analyzed the number of restaurant closures for the last 3 years. The number of closures increased from 2015 to 2016 and decreases slightly in 2017. Unsanitary conditions and Pest Infestations are the top reasons for restaurant closures. We also analyzed the reasons for restaurant closures and recovery time(the time taken to reopen the restaurant once closed). Dealing with sewage contamination and getting appropriate permits take the highest recovery time.
Sentiment Analysis:We performed sentiment analysis on user reviews to identify the tone of the review as positive, negative or neutral.

Customers seem to prefer restaurants with fresh authentic food, great service and ambience. Interestingly Japanese cuisine features prominently in both positive and negative sentiments. The prominent complaint they seem to have is with food delivery time and service.

Following are some of our graphs:

In figure 3, we found the correlation between restaurant ratings and other restaurant features such as takeout, delivery, parking, WIFI and alcohol. Its very interesting to that the correlation between rating and parking is negative. This shows people does not consider parking feature as an important factor while rating the restaurant. Our graph also says that restaurant who are providing takeout must ensure they provide better service to the customer every time as it highly correlated with the rating.

Figure 4 compares reason for restaurant closure and the duration for recovery across the 3 years from 2015 2017. Time taken to get a valid permit decreased significantly from 2015 to 2016 and increased slightly in 2017. Similarly, recovery time for restaurants has decreased for pest infestation and potable water. This might be because of cheap availability of resources to deal with these problems. Sewage contamination seem to have very high recovery period in the year 2017. New restaurants will be benefited from knowing these reasons, so they can invest resources accordingly.

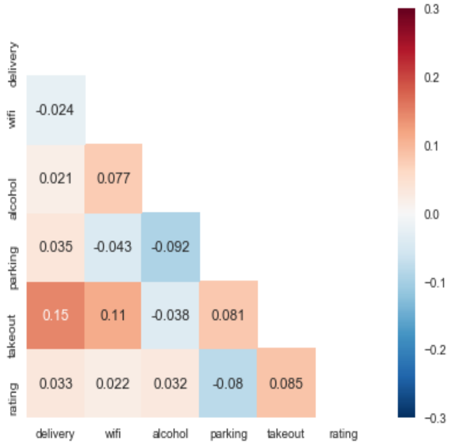


Figure 3: Correlation Matrix - Restaurant Features and Rating

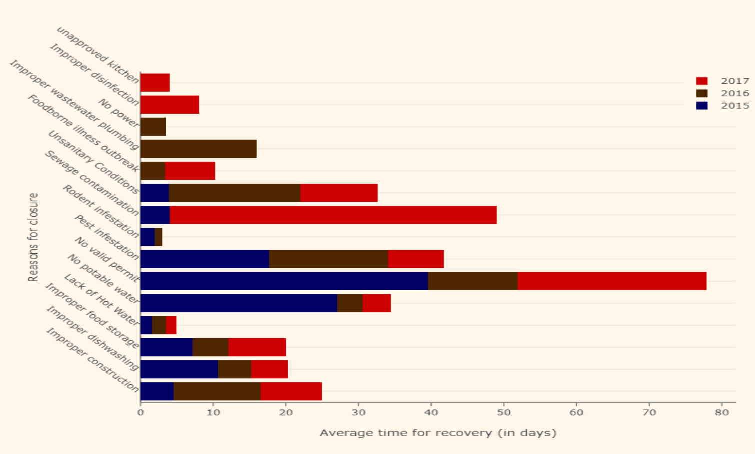
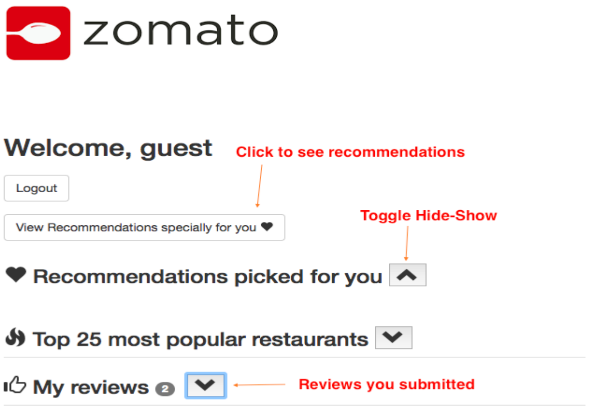


Figure 4: Restaurant Closure Reasons Vs Recovery Duration

4 DATA PRODUCT

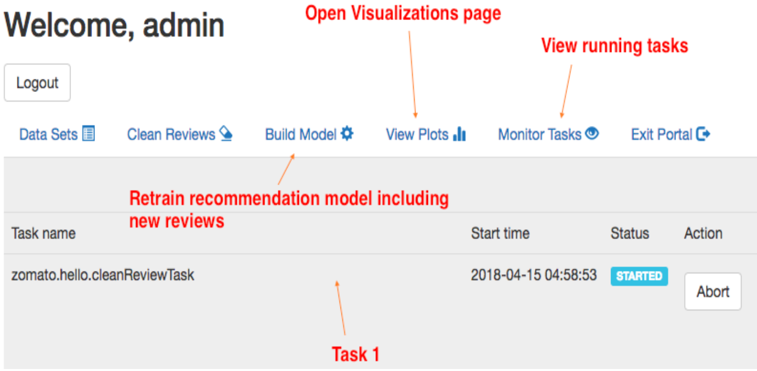

zomato
Welcome, guest
Logout
View Recommendations specially for you ❤
❤ Recommendations picked for you
Top 25 most popular restaurants
My reviews 2

Click to see recommendations

Toggle Hide-Show

Reviews you submitted

Figure 5: User Home Page



Open Visualizations page

View running tasks

Retrain recommendation model including new reviews

Task name	Start time	Status	Action
zomato.hello.cleanReviewTask	2018-04-15 04:58:53	STARTED	Abort

Task 1

Figure 6: Documents linked via references

One of the major challenge of this project was to have an application that promptly generates recommendations to users. After having researched several methods we chose to periodically retrain the model offline, given its simplicity. Also for this purpose we chose Google Cloud Platform[8] given its elastic nature for instance when we felt that the current configuration is pricey we could easily scale down the configuration within seconds and similarly scale up for computationally expensive jobs. To further reduce the running time of our model we chose an asynchronous distributed Tasks queue manager - Celery[9]. Celery which uses RabbitMQ[10] as a message broker between tasks. We also used Flower[13] which offers rich API to monitor Celery jobs. We claim that Flask[11] is the skeleton of our product as it seamlessly ties up all the components right from databases to visualization.

How it works?

1. Visit - `http://35.227.63.2:5001/login`

2. Sign Up with your favourite username and start fresh, or use 'guest' as username and password.
3. Chose any of the hotel visible to you and submit reviews
4. Recommendations will appear as soon as you start reviewing restaurants
5. At any moment, click the Zomato icon to return to home page
6. To train the model and view visualizations, enter Data science portal, by logging as administrator (Use 'admin' as both username and password)
7. Clean reviews and Build model jobs which are nothing but Spark/ Pandas scripts. These jobs run as asynchronous Celery tasks(so you can start the jobs forget about them and come back later to see the progress).
8. When the tasks enter 'SUCCESS' status, you can login again as the user who submitted the review and view restaurants specially picked for you.
9. Please click View plots to read interesting insights about foods and restaurants in Vancouver

5 LEARNINGS

- Zomato API was not useful to extract relevant data. It is limited to only 1000 calls a day and could provide only 5 reviews for each restaurant which makes it unsuitable for our project. Using standalone web scraper tools and python-selenium web driver is the best way to collect data in such case.
- Using latent features from topic modeling can refine the similarity neighbourhood for collaborative filtering techniques.
- Generating recommendations spontaneously is not a good idea for an interactive web interface. Solution- periodically retrain the model offline and calculate recommendations
- Celery's revoke command isn't guaranteed to terminate tasks
- Gained practical experience of deploying a big data project on cloud and integrate components. Also learned how to manage the infrastructure such that you meet your computing needs within budget.

6 SUMMARY

We relied on the intuition that if we can populate the sparse user-item rating matrix by using latent features from reviews rather than just relying on explicit ratings then we can improve the recommendations. So we performed EDA on user reviews to analyze user's preferences in terms of food, service, take-away, delivery and other factors and also studied the restaurants data to find interesting insights about the cuisines, cost, location etc. After studying the relationships between the facts derived from above step, we performed LDA Topic modelling to assist the Collaborative filtering between users, simply put, if two people talk about same topics they could be similar. As the final step, we wanted to deploy this model such that it can be easily integrated into an existing food restaurant portal system without hampering its current user experience, specifically the response time. As a result, we used Spark and Celery to run jobs in background and in parallel. Furthermore, visualizing the datasets from restaurants, reviews and food inspections gave us interesting insights about the food and restaurants people like in Greater Vancouver area, which can be viewed here - `http://35.227.63.2:5005/`.

References

[1] http://openrefine.org/

[2] https://developers.google.com/maps/documentation/geocoding/intro#reverse-example

[3] https://www.netlingo.com/acronyms.php

[4] https://www.nltk.org/_modules/nltk/tokenize/casual.html

[5] https://en.wikipedia.org/wiki/List_of_emoticons

[6] https://pandas.pydata.org/

[7] https://msdn.microsoft.com/en-us/vba/office-shared-vba/articles/getting-started-with-vba-in-office

[8] cloud.google.com/cloud_platform

[9] http://www.celeryproject.org/

[10] https://www.rabbitmq.com/

[11] http://flask.pocoo.org/docs/0.12/

[12] https://www.zomato.com/vancouver/restaurants

[13] http://flower.readthedocs.io/en/latest/