



COMP-SCI 5560 (SS16): Knowledge Discovery and Management
Third Project Report

RESTAURANTS RECOMMENDATION SYSTEM

Team #5 - Wisdom

Team Members

Samaa Gazzaz (9), Pooja Shekhar (38), Chen Wang (44), Dayu Wang (45)

Revision of Problem Statement

Motivation

During this iteration, we revise our motivation as follows. In building a restaurant recommendation system, we focus on the extraction of useful information from the user reviews in order to implement a novel ranking of restaurants to recommend. Next, the visualization of the recommendation is a key part of this iteration.

For Wisdom Restaurant Recommendation System, our aim is to utilize the power of user feedback in expanding our understanding of a restaurant quality. Instead of focusing on the general star rating employed by most currently existing recommendation systems, we focus on mining the user reviews for main perspectives regarding restaurants' quality. For our implementation, we focus on: service, ambiance, cost and food quality. Being able to utilize user reviews gives our system a huge advantage over existing recommendation systems.

Objective

Our objective in this iteration is to finalize the implementation of the major parts of the project. Meaning, during this period, we focus on implementing the algorithms, visualizing the reviews, and creating the user UI. For the algorithm part, we have multiple implemented algorithms including: the algorithm for comparing between two restaurants for deciding which one to recommend, correlation algorithm in which we find and learn the correlation between different aspects for ranking restaurants, and the main Wisdom algorithm in which we focus on the actual implementation of the recommendation process. Each of these algorithms will be discussed in detail along with examples in the implementation section.

In addition to implementing the algorithms as part of the backend implementation of the system, we also focus on the frontend implementation, by developing the UI for interacting with the system in addition to the pathway between the frontend and the backend implementation. Further information is provided in the implementation section.

Expected Outcome

The expected outcome from this iteration is mainly the finished implementation of the system. During this period, we collected more data in order to feed into the training model. Moreover, we were able to extract the categories' ranks (i.e service, ambiance..etc.) and correlation for different restaurants. This indeed helped with the recommendation process and with the novelty of this project. As a result, we will use the ranks of the different categories extracted from the reviews as our base for creating the pentagonal analyzer component. We are able to get 8 recommended restaurants depending on user preference and our own learning model.

For categories' ranks extracted for each restaurant to recommend, we are using that information in visualizing the quality of the restaurant recommended. More, we expect the user to provide some information in addition to the learnt model in order to be able to provide the best possible recommendations.

Project Domain

The main domain in our project is the recommendation algorithm. We aim to implement an algorithm for each of the following. First, we implement an algorithm that will enable the extraction of the value representing each category (i.e. service, food...etc). For this algorithm, each category is not only affected by the reviews of the restaurants, but also the values of other categories. In other words, we also implement a correlation algorithm which will calculate and estimate the correlation between different categories. For example, if a restaurant gets a high score for ambiance, it probably offers good food, thus the food category ranking will be boosted.

Second, we work on an algorithm for comparing restaurants being recommended. Namely, we develop our own algorithm for comparing the different categories for each restaurant (after adding weights) **Figure 1**. In this part, we aim to be able to choose a restaurant to be recommended such that it aligns users' preference with the best restaurant match. Finally, after comparing restaurants in order to choose the best available resource, we develop our Wisdom algorithm in which we provide a ranked list of recommended restaurants to the user.

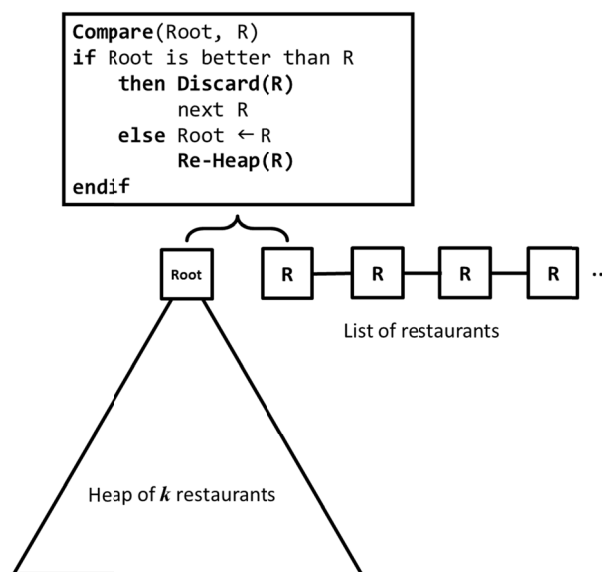



Figure 1. A heaping algorithm to select the top k restaurants amongst the entire list of trained restaurant data. The algorithm is good in space ($O(1)$ space complexity) but not good in time ($O(n \log k)$ time complexity).

Another aim in this iteration is working on the visual representation of the system. We have the necessary values to feed to the pentagonal analyzer as results from the previously mentioned algorithms. In addition, we have implemented a web page frontend for our system which will come in handy for users interaction. A screenshot of the web page is provided in **Figure 2**.

COMP-SCI 5560 (SS16) - Class Project



Wisdom Restaurant Recommendation System

Team 5: Samaa Gazzaz (9)
Pooja Shekhar (38)
Chen Wang (44)
Dayu Wang (45)

Please start your search here.

1. Looking for a restaurant for?
☐ Breakfast
☐ Lunch
☒ Dinner

2. Do you have **COST EXPECTATION?**
☐ Price No-Higher Than \$\$\$

3. Other kinds of service?
☐ Complimentary Wi-Fi Service
☐ Under-600-Calories **ENTREE** available

Figure 2. Screenshot of the system frontend implementation

Data Collection

Dataset Used:

The dataset used in this increment underwent a lot of preprocessing in the pipeline to extraction of keyphrases from all reviews and sentiment score related to each using Alchemy API. We chose most popular 1909 restaurants where popularity was based on review count of the restaurants. We extracted useful reviews from them and we are left with a smaller dataset containing more information. This was followed by topic discovery and keyphrase extraction which continued to make the dataset more precise and easy to work on. With the sentiment score attached to each keyphrase and a class attached to it, we calculated average sentiment score for each class - food, service and ambience. Now the dataset was fed to the recommendation engine to provide with efficient results and commendable runtime performance.

Tasks and Features

During this iteration, we worked on finishing the workflow components left. First, we finish implementing the pipeline reaching the trained model. Next, we arrive at the desired ontology to help understand the reviews. Finally, we implement our recommendation system using the algorithm we developed.

Pipeline:

For our project, understanding the reviews is a huge part of our work. In order to be able to rank restaurants being recommended, we depend on comments and how positive they are. Since comments reflects a more specific categorization of the quality of a restaurant, we would like to use those information for ranking recommended restaurants. Consequently, we will create an ontology that will enable the easy detection of the state of the reviews. In other words, whether the reviews contain the following topics: food, ambiance, service, and cost quality.

In order to be able to create that ontology, we need to create a model that will facilitate the extraction of terms related to each topic from the text reviews. The first step is providing good data to train our model as to what each topic means. For this step, our training data is a set of four topic specific data repositories, where each repository contains as much terms related to that specific topic as possible (Figure 3).

Food
Meat
pepper
Rib
Mac
cheese
bacon
Beef
lingonberry
chicken
Cooked
Duck
sandwiches
Ham
Cocktail

Figure 3. Example terms included in the food repository used to train the bayes classifier to recognize the food related terms.

Training the model is done using a Naive Bayes Classifier (Figure 4) where the prior are the repositories mentioned above. The posterior of the model is going to be the category to which term belongs (whether its service, ambiance or food related...etc.)

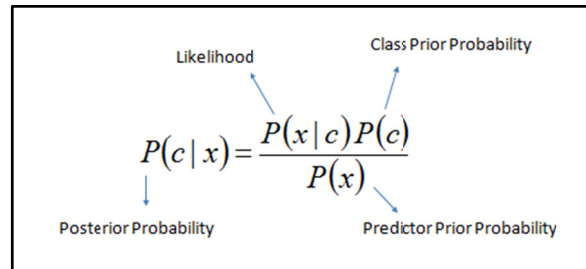


Figure 4. Naive Bayesian Classifier model

After the model is ready, we test it using the text reviews from Yelp users. Before we start using these reviews, we send them through the pipeline by implementing the following: NLP, IR/IE, LDA, FV. Next, the machine learning step is where we test the Naive Bayes Classifier; training and test information are provided in the following (Figure 5).

```
Corpus summary:
  Training set size: 2,149 documents
  Vocabulary size: 10,741 terms
  Training set size: 138,823 tokens
  Preprocessing time: 78 sec

Finished training LDA model. Summary:
  Training time: 12.6 sec
  Training data average log likelihood: -756.1
```

Figure 5. Statistic results regarding the training and testing of the Naive Bayes Classifier

Ontology:

The ontology for the project have been used for the implement the actual restaurant recommendation system, it have been separate as 4 aspects: Food, Service, Ambiance and Cost, and all of them have a lot of instances. For another part, we can 2 classes word to show the Compliment and Criticism. And because all of them has correlation, so there is no disjoint on it. The related figure has been showed on the Figure 6 and Figure 7.

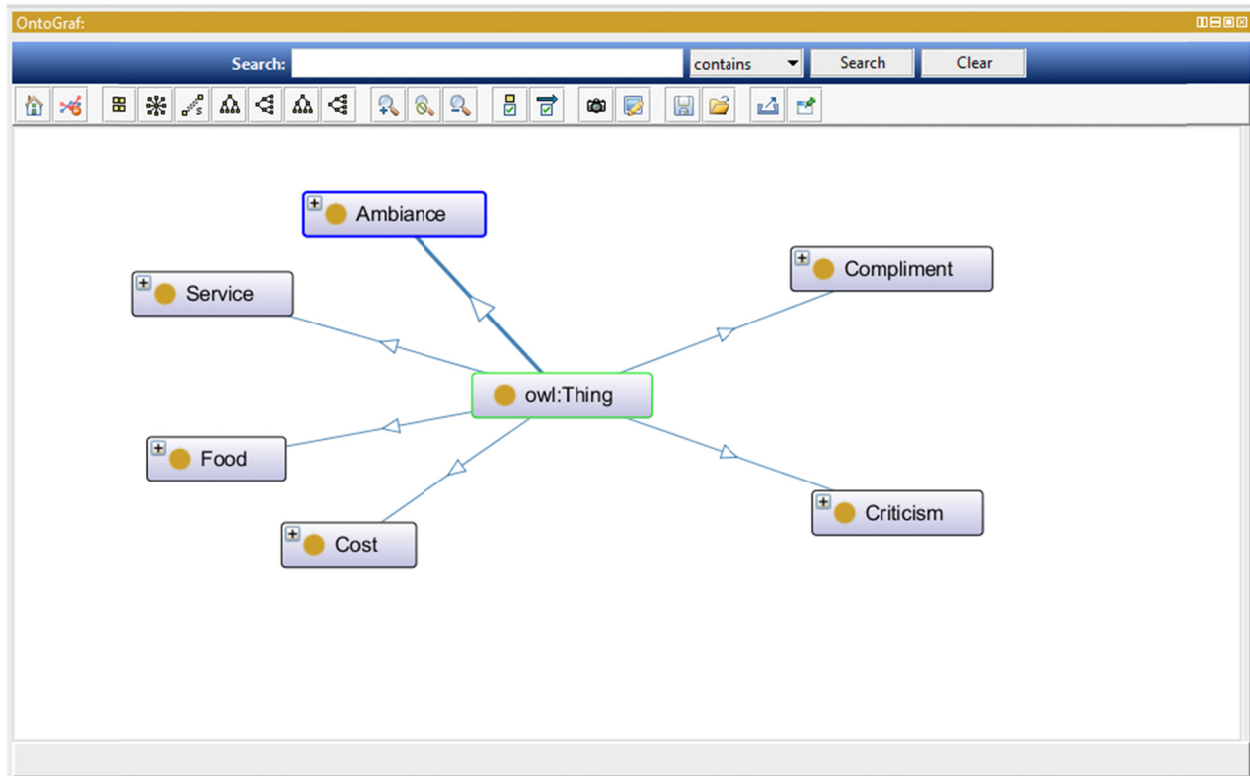


Figure 6. The screenshot of the Ontology Graph

Using this ontology, we can rate the 4 aspects (Food, Service, Ambiance and Cost) based on the Compliment and Criticism words. And our recommendation system will recommend the restaurant with the rating of these parts, the restaurant with high rating and lower variance will be recommended at first.

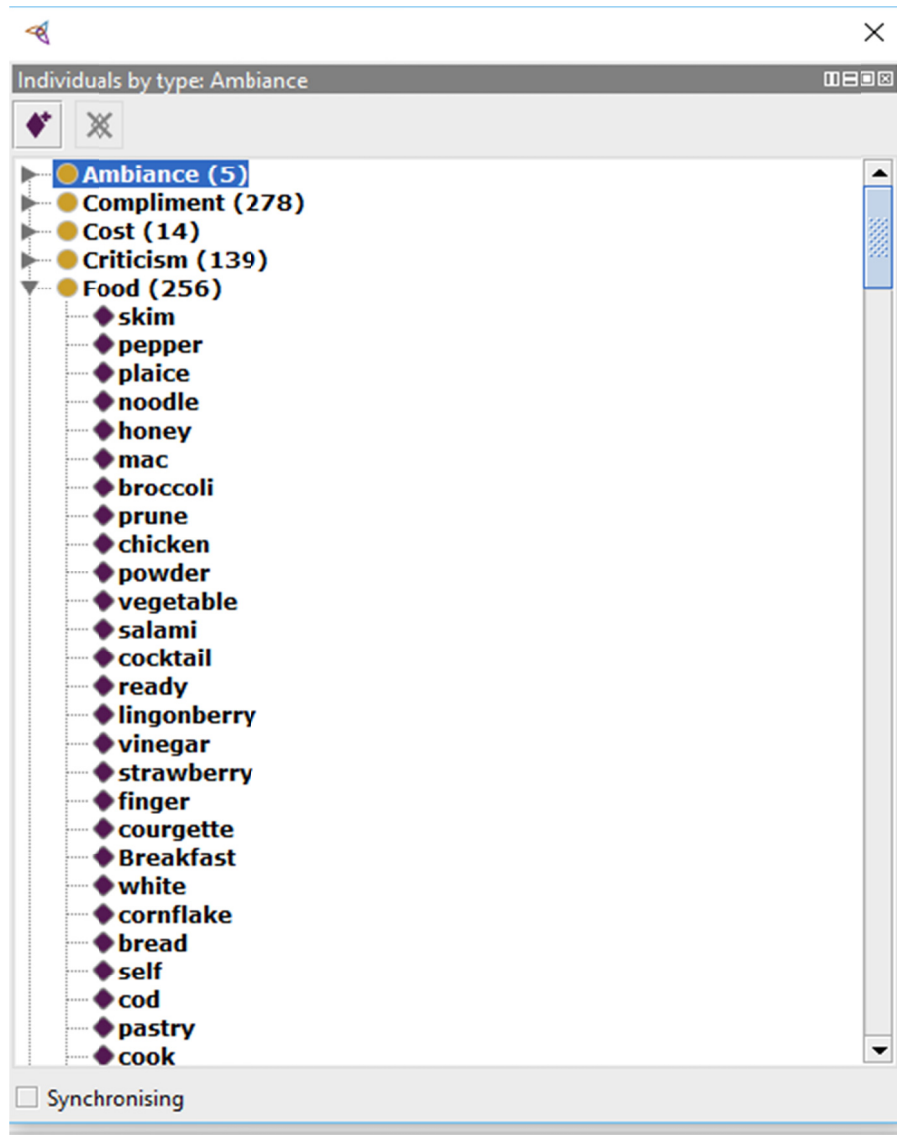



Figure 7. The screen shots of the Ontology instances. Food class have been showed.

Recommendation System:

Our recommendation system will give the user a web page to input their demand, The **Figure 8** showed that, we have some option buttons on the web page: Breakfast/Lunch/Dinner, Cost expectation, Other services (Does Wi-Fi available, does under-600-Calories entree available). And the user can also input a text by themselves, for instance, the user can input their address and what type of food do they like.

COMP-SCI 5560 (SS16) - Class Project



Wisdom Restaurant Recommendation System

Team 5: Samaa Gazzaz (9)
Pooja Shekhar (38)
Chen Wang (44)
Dayu Wang (45)

Please start your search here.

1. Looking for a restaurant for? ☐ Breakfast ☐ Lunch ☒ Dinner

2. Do you have COST EXPECTATION? ☐ Price No-Higher Than \$\$\$

3. Other kinds of service? ☐ Complimentary Wi-Fi Service
☐ Under-600-Calories **ENTREE** available

Figure 8. User input interface of the restaurant recommendation system

Based on the user input, we give a satisfaction that to show the user's request, the more fitting with the request, the higher satisfaction the restaurant have. And we always recommend the restaurant with higher satisfaction in front.

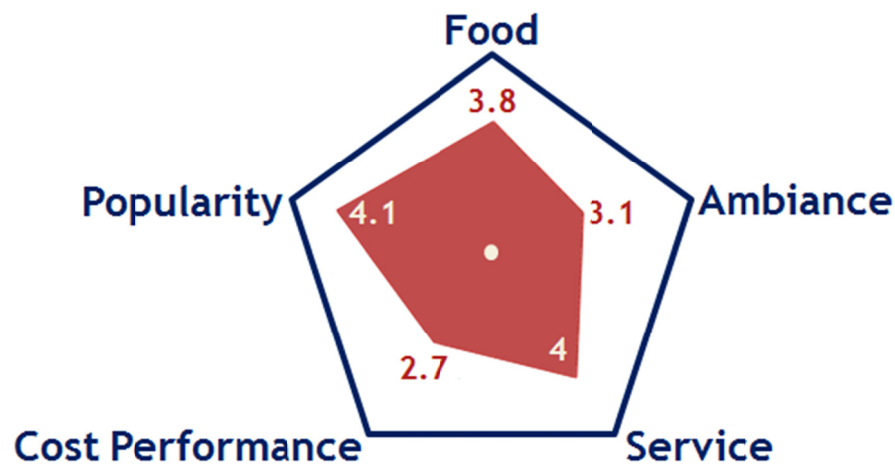
Between the same satisfaction level, we will give a recommendation based on the ranking of the 5 aspects: Food, Service, Ambiance and Cost, Popularity. (The popularity just based on the number of reviews, it doesn't need to do NLP and so on) Our system use the TF-IDF and feature vectors to process the reviews of the restaurants in the datasets, then get the comments about these aspects. Based on the NPL and Ontology, we get a rank of these parts. But we always consider these parts will influenced by each other. As we know, if a restaurant have a bad ambiance and service, we won't feel the food taste good even if the food quality is very well. On the contrary, if a restaurant which has a very good ambiance and service but just with normal quality food, we always feel the food taste good just because the ambiance and service influenced you. So we need train the machine and get the correlation between these parts and modify the grade of these parts constantly.

After this, we can give a proper recommend. At first, if the sum point have a very high level, we will recommend it at front, but if their sum point is closed, then we will recommend the restaurant based on the variance, the restaurant with lower variance will be recommended at first, because we think the restaurant with all aspects well is better than a restaurant with excellent food but awful service.

And if all of the parts have the same score between two restaurant, the restaurant will higher grade of food quality will be recommended at first, because as the restaurant, we always think the food is the most important thing.

Generally speaking, our system will recommend the restaurant based on this order:

User request -> High ranking in all aspects -> Low variance -> Food.



Data Processing	Comparison
1. Find synonyms for corners.	1. Satisfaction degree
2. Generate feature vectors.	2. General rating (average)
3. Find relations.	3. Standard deviation
4. Ontology learning	4. Food quality

Figure 9. The details of data processing and comparison

Implementation Specification

Software Architecture:

For our recommendation system, the training model is like this. Figure 10 showed us the details about our architecture: The NPL doing the first step of the dataset, then the words after processing can be used for the feature vector generator. The feature vector is constituted by TF-IDF, LDA Processor. The data processed by the feature vector generator and the relation analyzer can be used by the Ontology Processor. The machine can be trained by this way over and over.

The second part shows us the layers of the actual model. The first layer is the algorithm and Web Handler, the second layer is Correlation Learning Model and User Input Handler, and the third layer is the correlation Learning Model and UI.

As the architecture showed us, the system using the NPL to process the reviews of restaurants, then these data using to do TF-IDF, and LDA processing. Then the data process using the ontology model, became 4 parts of details about the restaurant: Food, Ambiance, Service, Cost performance. And the popularity is be reflected by the number of reviews. Then the data can be used to training and recommendation.

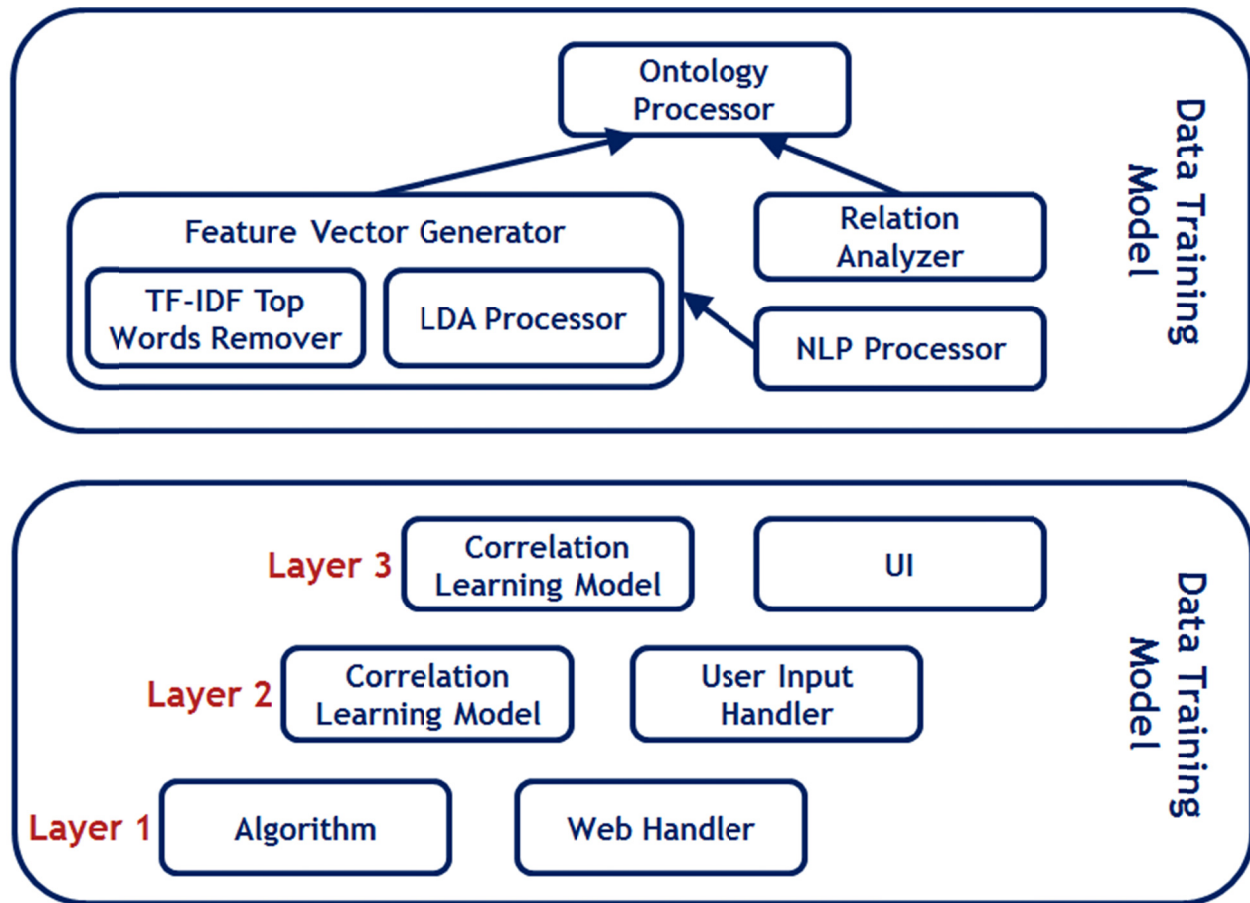


Figure 10. The architecture of recommendation system

Algorithm of Pentagonal Analysis

In this section, we are talking about the algorithm inside the *Pentagonal Analyzer* component, which is implemented from scratch since it is designed for our restaurant recommendation system only.

In the last project report, we know the basic knowledge about the tournament tree. In it we have $(2n - 1)$ units of space to store all the restaurants' information (n leaf nodes for restaurants and $(n - 1)$ internal nodes). And in the leaf level, the information of restaurant have been stored. When some information be updated, it should generate indicators to represent the top 8 restaurants. By comparing restaurant 1 and restaurant 2, the better restaurant goes up to the next level. We do it over and over, until the whole tree has been filled, then the root will be the best restaurant in all of them. The Figure 11 is showing the details about this step.

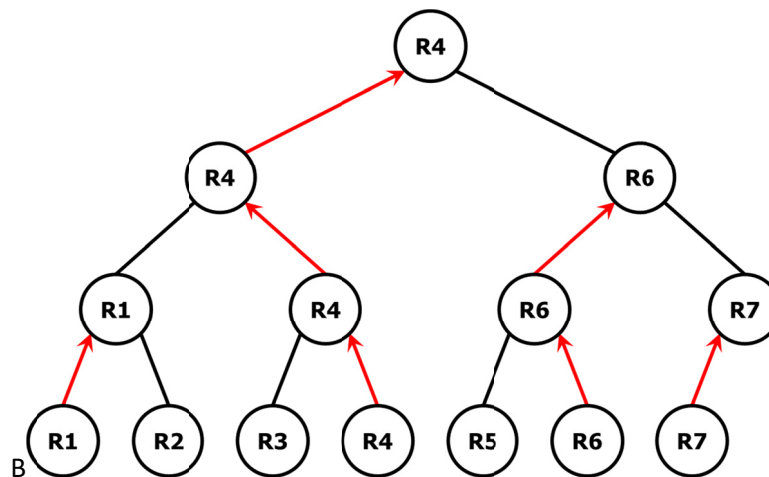


Figure 11. A *Tournament Tree* Selection Algorithm for restaurant recommendation system. R1 - R7 are restaurants stored on the leaf level. By comparing two restaurants next to each other, the “winner” (better restaurant) is pushed up to the next level (red arrows in the picture). After going through every node in the tree, the “final winner” (R4 in the picture above) will be placed at the root of the tree.

In this algorithm, if we just choose one best restaurant, we still need to take $O(n)$ time complexity, but if we choose more than one restaurants, it will be different. Suppose we want to choose the top first and second best restaurants, we run the tournament tree to get the first best one, it take $O(n)$ time complexity, but when we are finding the second best restaurant, it just need $O(\log n)$ time complexity. In the Figure 12 we can see, when we choose the second best restaurant, we just need compare one time in one level of the tree, which is $O(\log n)$.

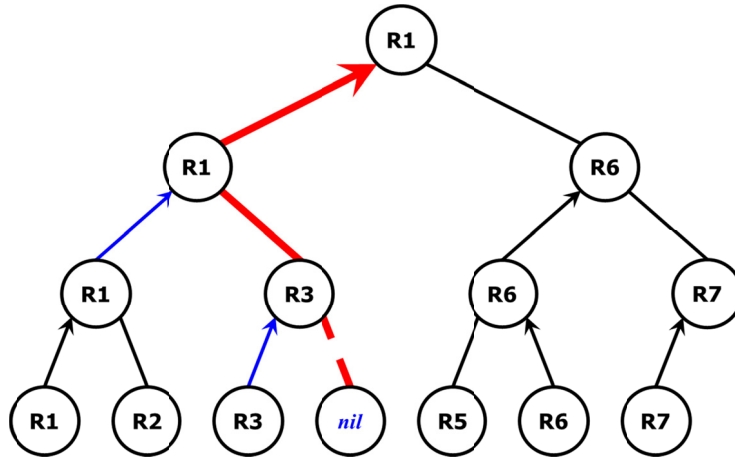


Figure 12. The tournament tree selection algorithm to choose the second best restaurant after the best restaurant was taken. We only need to update one path (red path) instead of the entire tree to obtain the second best option.

Generally speaking, the time complexity to select the top one and two restaurants is $O(n + \log n)$. Furthermore, the time complexity to select the top k restaurants is $O(n + k \log n)$, which is proved to be the fastest sequential algorithm. So with this algorithm, we can save a lot time when we choosing more than one recommendation. However, the demerit of the algorithm is that beside the storage of restaurants' data, we need an extra $O(n)$ of space to build the tournament tree. So if the data is pretty large, the algorithm requires extra space (about double space) to work.

UML Models:

The UML Model is the same with last report, it have been demonstrated in Figure 13.

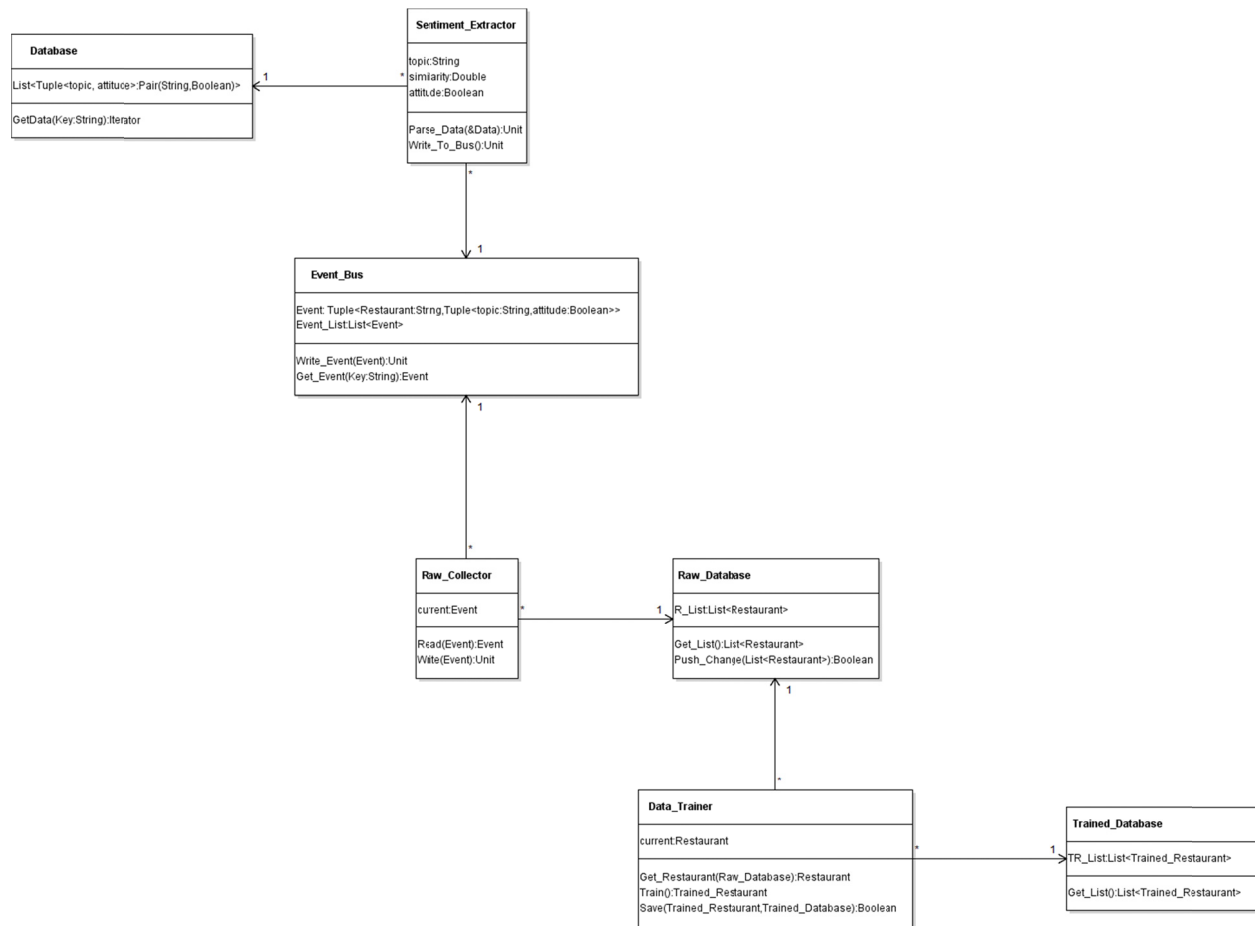


Figure 13. The UML diagram of the core engine of restaurant recommendation system. The final trained data is stored in Trained_Database.

Workflow Diagram :

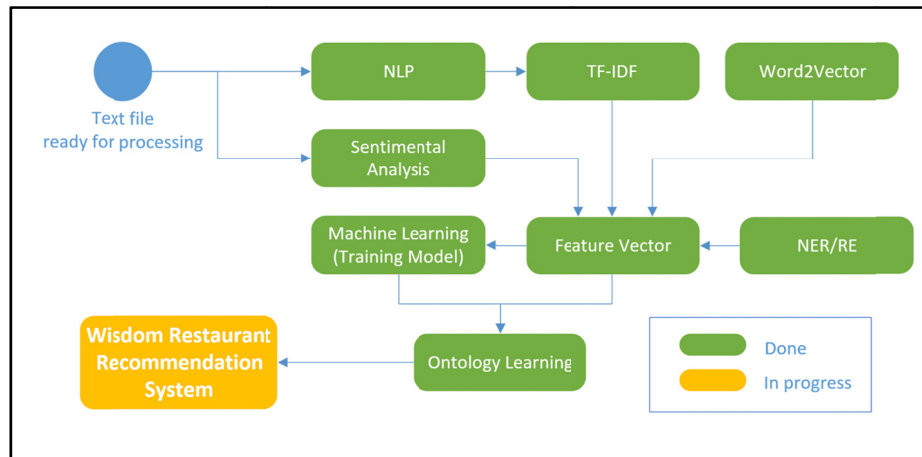


Figure 14. An updated workflow diagram of the recommendation system, with green components indicating the completed part and red components indicating the parts being implemented.

Existing Services:

Gson:

Google Gson provides method that can convert a json string to java objects, and it can also convert a java objects to json string. The java objects can be hierarchical. In our recommendation system, the outputs are a json string. And it can't be used as input of other parts directly, so we always need convert the json string to java objects. It's very useful to us to process the dataset, join two parts class, and make an output.

Java servlet:

A Java servlet is a Java program that extends the capabilities of a server. It's a class of java programming language. The server can keep the "request-response" model to call on the application. Even if servlet can respond all the kind of request, but it always be used to extend the applications based on web server. Thus, a software developer may use a servlet to add dynamic content to a web server using the Java platform. The use of servlet have be showed on the **Figure 15**. In our recommendation system, the use of the Java servlet is like this figure, the web page translator Tomcat gives us the active form, then the Java servlet help us translation it, then we can use the user's request and data.

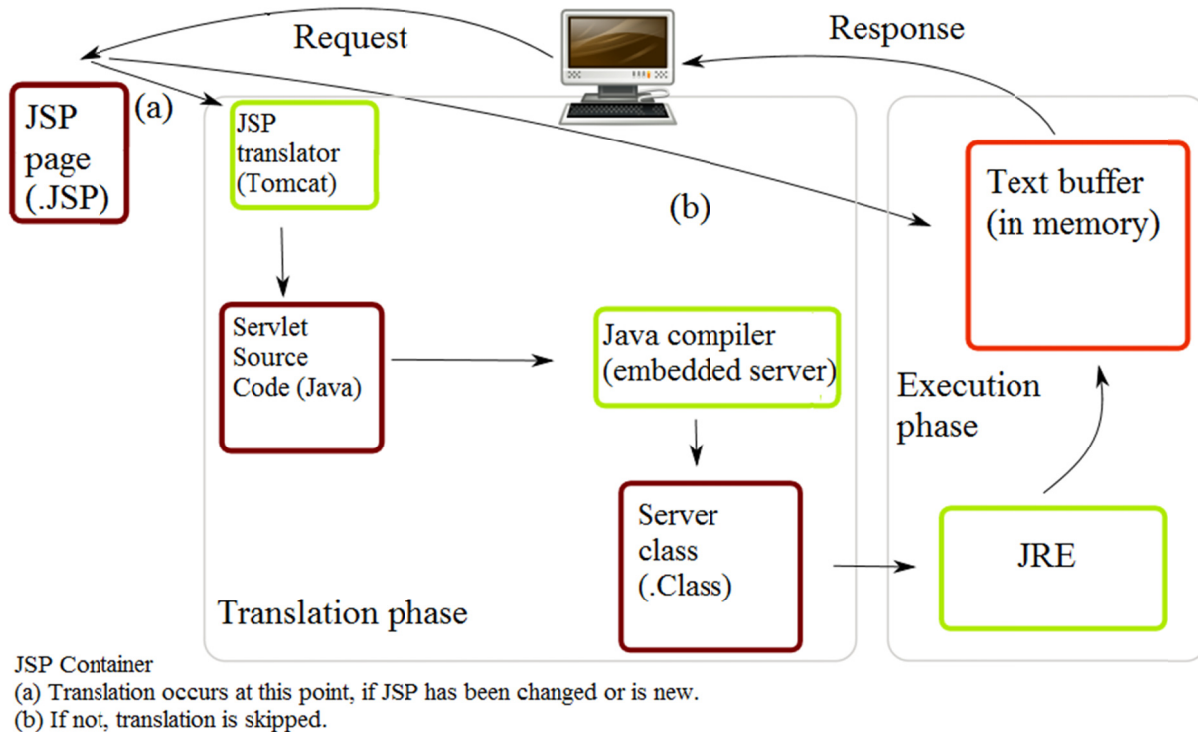


Figure 15. Life of a JSP file. The use of Servlet have been displayed.

(Wikipedia Java servlet) https://en.wikipedia.org/wiki/Java_servlet

LDA, Pipeline, Ontology have been used, the details of them can be see in their parts.

New Services/Features to be Implemented:

As we know, as a recommendation system, the most important thing is the requests of the customers. In our recommendation system, we recommend the restaurant based on satisfaction at first, which means that how much the the restaurant matched with the user's request. As we said in the Recommendation System part, our system offer a web page so that the user can choose their request on it. The request include:

1. Looking for a restaurant for **Breakfast/Lunch/Dinner**.
2. Do you have **COST EXPECTATION?** (Price No-Higher Than \$\$)
3. Other kinds of service: Complimentary **Wi-Fi Service/Under-600-Calories** ENTREE Available.

The user can also input a text to request some more things include the city, the zip, the type of the food and so on.

With these request, our system will give a symbol named satisfaction, it will show the degree of restaurant matched user's request. For example, we give the city and the zip 1 point, the type of food 0.4 point, the Wi-Fi 0.2 point. Because we think the restaurant in the user's city is the most important

things in those request. And the Wi-Fi available is much slighter than the address and food type. The actual value for these request is:

city/zip	1.0
Breakfast/Lunch/Dinner	0.8
Food type	0.4
Cost expectation	0.2
Wi-Fi	0.2
Under-600-Calories Entree	0.2

Then we will give a recommendation based on these request at first. Higher point means more important, so the highest grade of satisfaction will be recommended at first.

If two restaurant have the same satisfaction score, we will recommend the restaurant using our pentagonal method which show the details about the Food, Ambiance, Service, Cost Performance and Popularity. The details of this method have been told in the past two project reports.

Results & Evaluation

Precision

We chose to use Multi Class evaluation metrics to predict the accuracy of Naive Bayes classifier while evaluating our recommendation system since our dataset containing reviews could be distributed on three classes namely - food, services and ambience. The accuracy achieved was 89.66% which showcases a good measure of precision. This is a measure of accuracy of restaurants recommended to the user based on the user inputs.

Runtime Performance

The system provides seven recommendations to a user within 10 secs while working on a dataset containing data of 1909 restaurants. We plan to increase the size of dataset to check the scalability of our system in future.

Project Management

Contribution of Each Member:

Member	Contribution
Samaa Gazzaz	Documentation: <ul style="list-style-type: none">• Motivation, Objective, Expected outcome, Project domain Tasks and Features: <ul style="list-style-type: none">• Pipeline: NLP - IR/IE - LDA - FV - ML Diagram: <ul style="list-style-type: none">• Update workflow diagram
Pooja Shekhar	Documentation: <ul style="list-style-type: none">• Dataset Used, Evaluation Tasks and Features: <ul style="list-style-type: none">• Formatting the dataset to be fed to recommendation engine• Topic distribution using LDA• Keyphrase extraction from reviews• Sentiment score calculation using AlchemyAPI for each keyphrase• Visualization of results using High Charts - UI
Chen Wang	Propose of the Ontology Documentation writing and organization: <ul style="list-style-type: none">• Ontology• Recommendation system• Software Architecture• Algorithm of Pentagonal Analysis• UML Models• Existing Services• New Services/Features
Dayu Wang	Implementation of the whole recommendation system, excluding data training part Algorithm of Recommendation System - Finalization Pentagonal Self-Adjusted Analyzing Model - Finalization

Version Control/Screenshots:

For this project, we used GitHub as the main version control tool. The whole project, in addition to documentation, is up on: <https://github.com/SamaaG/WisdomRecSys>

Concerns/Issues:

The pentagonal representation method, which could be either <svg></svg> in HTML or HighCharts.

Future Work:

Web pentagonal representation of restaurants.