Yelp - Personalized Restaurant Recommendation

D'Souza, Ajay
ajaydsouza@gatech.edu

Abstract

Datatouille - Personalized Restaurant Recommendation using Yelp reviews : Mine Yelp restaurant reviews to extract the dish names a user likes in a restaurant. Use these dish names to recommend new dishes along with specific restaurants that specialize in those dishes.

Contents

List of Figures

List of Tables

1 Introduction

Currently when recommendation systems recommend restaurants to users using a star rating, the dish names that appeal to a users palate generally remain as unlabeled latent features in that recommendation in the currently popular matrix factorization family of algorithms.[1]

The goal of this project is to be able to understand user tastes and recommend to the user one or more specific dishes, followed by a recommendation of restaurants in the users geographic neighborhood that specialize in those dishes [2]. We want to expose dish name which is central to restaurant recommendation as a labelled feature in our recommendation system.[3]

2 Problem Definition

Choosing a restaurant is a very personal choice. Using star ratings does not provide a personalized basis for recommendation. The Yelp Challenge dataset provides close to a million tuples of (user, restaurant, timestamp, review). From the reviews in each tuple we will extract the dish names. If the User has mentioned a dish we assume the user likes the dish. *Like* and *dislike* will be stored as binary values. Next, if the user expresses a positive sentiment of the experience of having that dish in that particular restaurant, we assume that the restaurant *specializes* in that dish. We keep a count of the number of users expressing positive sentiment for a dish-restaurant tuple as a measure of the restaurant specializing in that dish.

We build a sparse table $User \times Dish$. We perform Item Based Collaborative Filtering and User Based Collaborative Filtering to generate *top-N* recommendations of dishname and restaurant specializing in these dishes for a given user. Results will be filtered by the geographic location of the user.

[1] Heilmeier Question No.2
[2] Heilmeier Question No.1
[3] Heilmeier Question No.3

3 Survey

3.1 Dish name Extraction

[7] presents a hybrid approach where a gazetteer-driven NER algorithm is used to partially label the corpus. The partially labeled corpus is then used to train a sequence model that induces labels on the remaining entities.

[8] discusses the degradation of performance of NLP tools when applied to social media. It discusses the POS tag patterns that can be used to get named entities.

[10] was studied while evaluating CRF approach in [9] for Random Forests based classification.

[11] was studied to understand the variable importance feature in Random Forests.

3.2 Sentiment Detection

[14] discusses the methods of Naive Bayes (NB), SVM and Max Entropy(MAXENT) for sentiment detection of movie reviews. The authors build a labeled vocabulary of unigrams and bi-grams from the corpus of labeled movie reviews. Classifiers are trained using this vocabulary. The NB, SVM and MAXENT classifiers reported comparable accuracy of around 80% with the test data

[18] specifies a way to handle negation words by appending NOT_ to every word between the word with negation till the next punctuation mark. Using this approach words like $isn't good$ will get negation tags and are likely to be seen as negative as they should be.

[16] and [17] provide a unsupervised method for sentiment classification. The method picks the adjective and adverb phrases from the POS tag patterns shown in table (1) below.

It then performs a web search to compute their Pointwise Mutual Information (PMI) score as in (1), and the Semantic Orientation as shown in (2). The review is taken to be positive if the average SO of all the phrases in a review is positive.

	First Word	Second Word	Third Word (Not Extracted)
1.	JJ	NN or NNS	anything
2.	RB, RBR, or RBS	JJ	not NN nor NNS
3.	JJ	JJ	not NN nor NNS
4.	NN or NNS	JJ	not NN nor NNS
5.	RB, RBR, or RBS	VB, VBD, VBN, or VBG	anything

Table 1: Phrases to be extracted for PMI

$$PMI(word_1, word_2) = \log_2 \left(\frac{p(word_1 \& word_2)}{p(word_1)p(word_2)} \right) \tag{1}$$

$$SO(phrase) = PMI(phrase, excellent) - PMI(phrase, poor) \tag{2}$$

3.3 Recommender Systems

[20] discusses the NMF matrix factorization method for recommender systems. For sparse matrices and for serial computation Stochastic Gradient Descent is recommended as an suitable choice.
[21] details the implementation of the Stochastic Gradient Descent method for matrix factorization.
[22] describes how Item Based and User Based Collaborative Filtering can be used to determine similarities between items' consumption and user consumption respectively. It also explains how to make a recommendation using this two approaches.

4 Intuition

The state-of-the-art in restaurant recommendation systems is to base recommendations on star ratings made by all users, which can leave out significant details such as restaurant specialty and user preference. Since Datatouille uses a user's own reviews to understand which dishes a user likes and a restaurant's reviews to determine the restaurants specialty, a recommendation can more accurately pair a user with a restaurant and make a truly personalized recommendation[4].

[4]Heilmeier Question No.3 - Why it will be successful ?

5 Proposed Method

The following figure gives a high level overview of the different components of Datatouille:

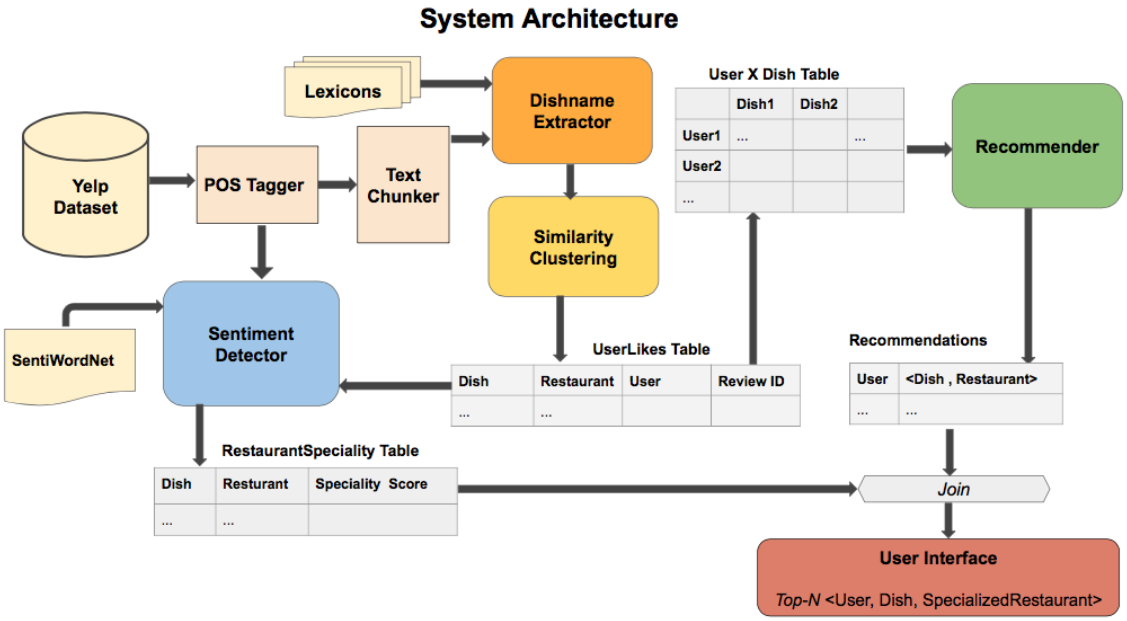

Figure 1: Components of Datatouille

5.1 DishName Extraction

We use Random Forests classifier as below [8].

1. Label Data

   From a mix of reviews we generate a set of 9500 n-grams (n = 2 to 6). The n-grams are labeled as either DISHNAME or NOT_DISHNAME. We divide these into a test set of 1500 items and training set of 8000 items.

2. We POS tag each sentence as in figure 2

   E.g. For sentence "I ordered Buddha Spring Roll and chicken fried rice", we get

3. Text is chunked, relevant n-grams in noun phrases and n-grams outside noun phrases are extracted as shown in figure 3.

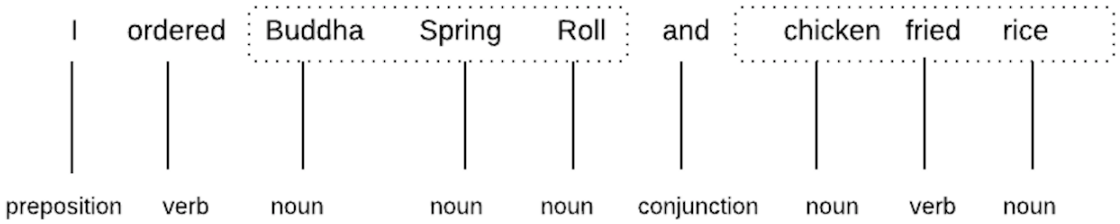

Figure 2: Dish Name Extraction - POS Tagged Sentence

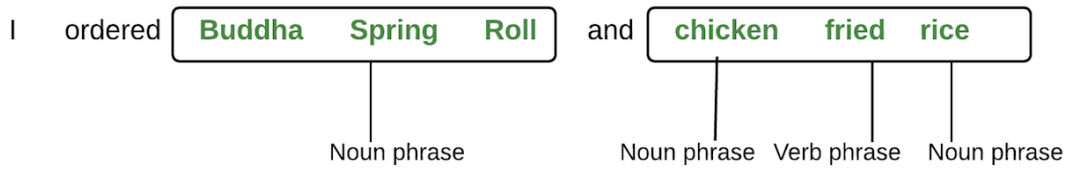

Figure 3: Dish Name Extraction - n-grams

4. Food lexicons were created using readily available information on the web, we generate the following boolean feature vector for each selected n-gram.

   (a) Begins with Noun ?

   (b) Begins with *relevant adjectives* that commonly appear in food names ? E.g.: spicy

   (c) Begins with *relevant verbs* commonly used in Yelp restaurant reviews ? E.g. ordered.

   (d) Ends in Noun ?

   (e) Has a food context clue ?. This is guessed from the presence of *cooking style* terms. E.g. fried, *Common Food* nouns E.g. Pizza and *relevant adjectives* in the n-gram

   (f) All words in a noun phrase ?

   (g) Is dish name ? (training label)

5. Random Forests with the following parameters was used to train the classification model

   (a) Number of trees: 10

(b) Quality of split function: Gini

(c) Maximum features: Square root of number of features

5.2 Similarity Clustering

The dish names from different reviews may differ due to spelling errors, use of short names (substring of the main name). Levenshtein distance is used to find similar dish names. A graph of similar dish names as adjacent nodes is created. The graph is sorted by node degree and clusters of one hop neighbors are created. Each cluster represents a single dish entity.

5.3 Sentiment Detection

We chose to use Naive Bayes (NB) classifier for sentiment detection, since a large domain specific corpus is available for training it. Levenshtein distance is used to determine similar dish names and a graph of dish names with similar names is created.

1. Manually labeled 3000 random sentences as positive, negative or neutral on sentiment. This is the test set for measuring classifier performance.

2. From the proportion of labels in the training data compute the class priors. For training, we use the overall rating as the class label. A rating of $4, 5$ is taken as positive, 3 as neutral and $1, 2$ as negative.

3. POS tag each review sentence, remove stop words and nouns

4. Use NOT_ as the negation tag on every word between the negated word and the next punctuation

5. Build an effective vocabulary of uni-grams and bi-grams to build a feature vector.

6. Train the NB classifier by calculating the probability of each uni-gram or bi-gram in the feature vector with Laplace smoothing as

$$P(w_i|c_j) = \frac{count(w_i \in c_j) + 1}{\sum_i count(w_i \in c_j) + |V| + 1} \tag{3}$$

$$|V| = \text{size of the vocabulary used} \tag{4}$$

7. Additional features where engineered for the following factors

(a) Overall rating provided by the user

(b) sentiment polarity for words in the $SentiWordNet$ lexicon

With the trained NB model we can classify a sentence as below

(a) Tokenize the sentence to be classified. Construct a binary array for presence of n-grams in the feature vector of the trained NB classifier.

(b) Compute the probability of a sentence belonging to a class as

$$C \leftarrow \arg\underset{j}{max} P(c_j) \prod_i P(w_i|c_j) \tag{5}$$

5.4 Recommendation Engine

We employ the following algorithm similar to [22] for Datatouille recommendations.

1. A $User \times Dish$ table is given as input. Create a *Dish* table by dropping the *User* column.

2. Perform Item Based CF:

   A $Dish \times Dish$ similarity matrix is formed using *Cosine similarities* between the dish vectors from the *Dish* table.

3. Perform User Based recommendation:

   Create the $User \times Dish$ *Recommendation Matrix* where every entry has a score calculated thus:

   For every user U and every dish D that user has consumed:

   (a) Identify *Top-N* similar dishes of D from $Dish \times Dish$ similarity matrix and get similarity scores in d_s vector.

   (b) Get a consumption record as c_r vector of the user U from $User \times Dish$ for each of the similar dishes.

   (c) For each similar dish, $Score_{u,d} = \frac{d_s^T.c_r}{L1Norm(d_s)}$ is calculated and updated in the matrix at (u, d)

4. For a given user, we identify a row in $User \times Dish$ *Recommendation Matrix* , we return Top-N dishes by sorting the values of the user row.

5.5 User Interface

UI is through a web interface. For a chosen user it displays the name of the restaurant along with the dishes recommended. The display is on a interactive map with marker sizes showing the strength of the recommendation.

5.5.1 Data Visualization

We intend to have visualization of aspects of algorithm for data extraction and the sentiment detection.

5.6 Innovation

1. We demonstrate how to choose and use labeled features with existing recommendation algorithms for deriving relevant recommendations instead of a star rating. This idea can also be extended to algorithms that uses matrix factorization with star ratings where important features remain latent with no control over them.

2. For sentiment detection we used an ensemble of NB classifier based on corpus vocabulary and the average Sentiment Orientation of a sentence based on sentiment polarity of sentiment expressing words from SentiWordNet.

3. Use a combination of NLP and online lexicons to train a simple Random Forest classifier using a smaller corpus of training data for extracting dish names.

5.7 Implementation Details

5.7.1 Dish Name Extraction

Feature vector generation for dish name extraction is implemented in Java using Stanford POS and Apache OpenNLP libraries. The classification algorithm is implemented in Python using scikit-learn's RandomForestClassifier.

5.7.2 Sentiment Detection

Sentiment detection is implemented using python. The python nltk module is used for tokenizing, POS tagging and stop word removal. The $SentiWordNet$ lexicon is used for sentiment polarity. The classifier was trained using ten fold cross validation. A labeled data set of 3000 reviews was used for testing.

5.7.3 Recommendation Engine

The recommendation engine is built as detailed in proposal section using python along with the pandas and scipy library. The pandas library assisted in keeping track of data while the scipy library provided a function to compute cosine similarities.

5.8 UI

The UI pulls together the original Yelp dataset for information about users and businesses with the recommended dishes and restaurants for a user and displays the information on Google Maps.

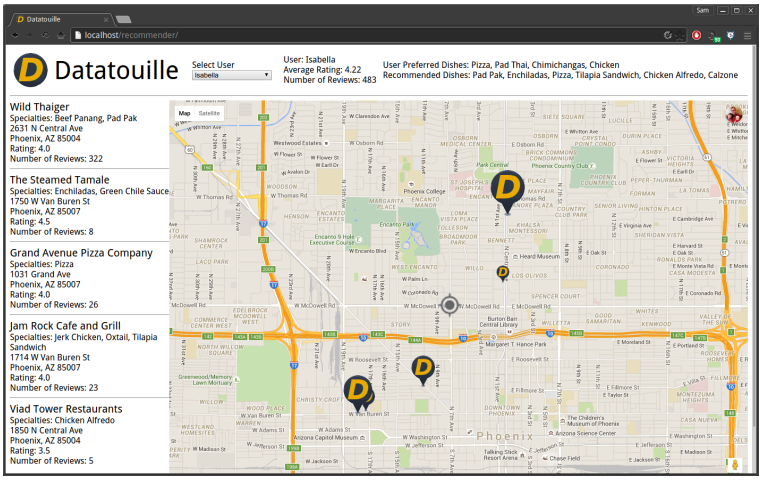

Figure 4: Datatouille UI

For the demo, select a user from the combo box. The user's Yelp information will be displayed at the top of the page alongside the dishes extracted from the user's reviews and a list of suggested dishes. The recommended dishes are used in a query on the table of restaurant specialties. For portability, sqlite was used as the database back end, but it lacks the math functions (sin and cos) required to calculate the distance from the user's current location to the restaurant. With a more robust database solution, the final goal can be fully realized.

6 Experiments and Evaluation

Dish name extraction accuracy was measured for precision and recall on a labeled test data set 1500 labeled sentences

Sentiment detection accuracy was measured on a labeled test data set of 3000 labeled sentences.

For measuring performance of recommender system we propose to use reviews of a subset of users as the test set. We will use their reviews up to some point in time t_1 to make a recommendations. We will then measure the recommendations made with the actual restaurants reviewed by the same user after that point in time t_1

Dish name extraction was evaluated using Google-Refine which first yielded roughly 87,000 different dishes for the Phoenix area. Google-Refine's cluster feature provided a way to see if top clusters were dishes or not.

The data was then filtered when building the $User \times Dish$ to only include the dishes that occurred 50 times or more, this left 688 dishes. After a manual review it was found that only 6.98% were terms that did not correspond to dishes; these dishes were omitted from data.

Due to paucity of time sentiment detection is not integrated in the UI. Instead of using sentiment detection, the recommendation system uses the presence of a dish in the dish extraction as an indication of user preference and restaurant specialty.

To test the UI, several users were chosen from the drop down box. Their dish recommendations were compared to the extracted dishes for consistency. The restaurant recommendations were confirmed based on the results. The best test of the UI and the entire project as a whole would be to let numerous people try the project and record their experiences with a questionnaire. Time did not permit this.

Recommendation was verified manually while testing the UI. The sample below shows how relevant dishes are recommended to a used based on preferred dishes. In this example Matthew likes Fish Tacos and only taco related items are recommended. Also, Matthew only has one preferred dish because he only has two reviews so only one dish was extracted for that user.

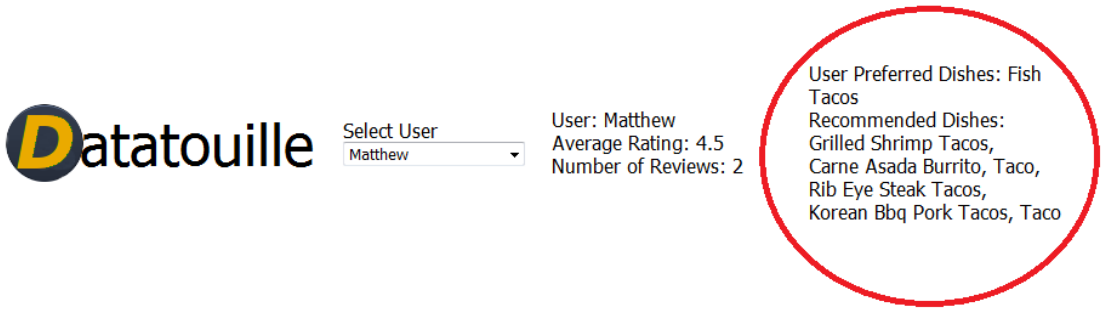

Figure 5: Sample Recommendation from UI

7 Results and Observations

The following results where observed [5]

- Performing Principal Component Analysis using these feature vectors generated for dish name extraction shows separation of the DISHNAME and NOT_DISHNAME labeled training data (6), thus providing a validation for the features engineered.

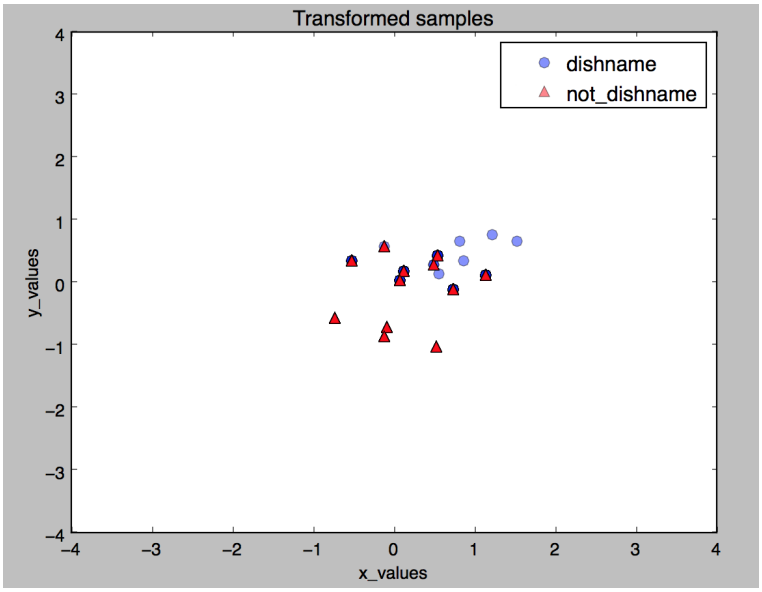

Figure 6: Dish Name Extraction - PCA

[5] Heilmeier Question No. 9

- The ROC plot for the classification model created with 1500 test n-grams is shown in figure (7) below. A prediction probability$\geq$ 0.8 is chosen.

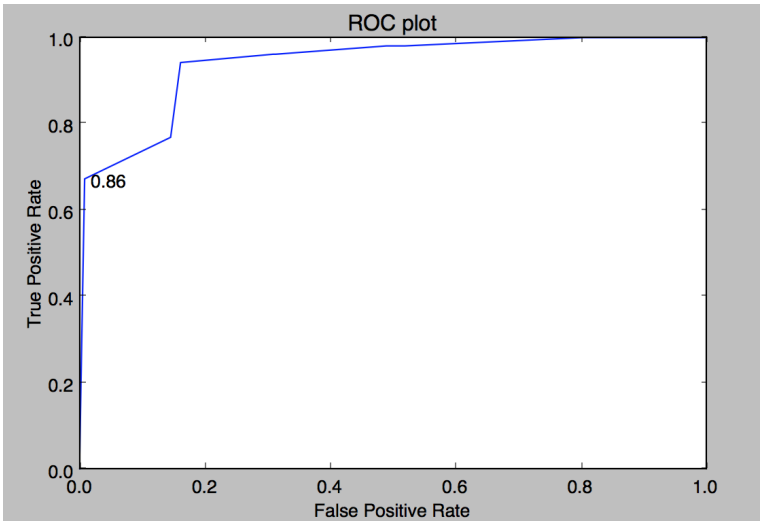

Figure 7: Dish Name Extraction - ROC plot

- The confusion matrix for dish name extraction based on a 1500 n-grams labeled test data is below (8). Precision: 76% and Recall: 71%

n = 1500	**Predicted NO**	**Predicted YES**	
Actual NO	1436	12	1448
Actual YES	15	37	52
	1451	49	

Figure 8: Dish Name Extraction - Confusion Matrix

- Sentiment detection was tested on a labeled set of 3000. The Naive Bayes classifier gave an accuracy of 65.67% on this labeled set

- (2) is the confusion matrix generated for the sentiment classifier

	Predicted Positive	Predicted Neutral	Predicted Negative	Total
Actual Positive	1146	345	77	1568
Actual Neutral	186	382	76	644
Actual Negative	67	282	439	788
Total	1399	1009	592	

Table 2: Confusion Matrix for Sentiment Detection

- (3) is the Precison and Recall table for the sentiment detection. The tests showed the best precision and recall for the positive class. This serves the projects requirement as we are interested only in reviews which are positive.

Class	Precision(%)	Recall (%)
Positive	81.92	73.08
Neutral	37.85	59.32
Negative	74.15	55.71

Table 3: Precision Recall Matrix for Sentiment Detection

- Due to paucity of time Sentiment detection module is not integrated with the UI. However these test results indicate that it will integrate well with the UI and enhance user experience.
- The $User \times Dish$ matrix is sparse but shows adequate overlap of different users and different dishes. This implies good recommendation model using this matrix is possible. A 1000 x 800 segment of the original matrix is shown here.
- Recommender system chose dishes within a high degree of similarity to the user's reviewed dishes.

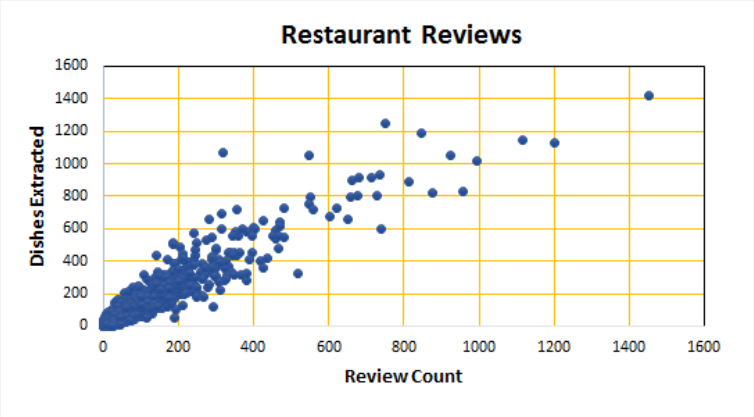

Figure 9: Dishes extracted versus review count. Almost a consistent dish per review trend.

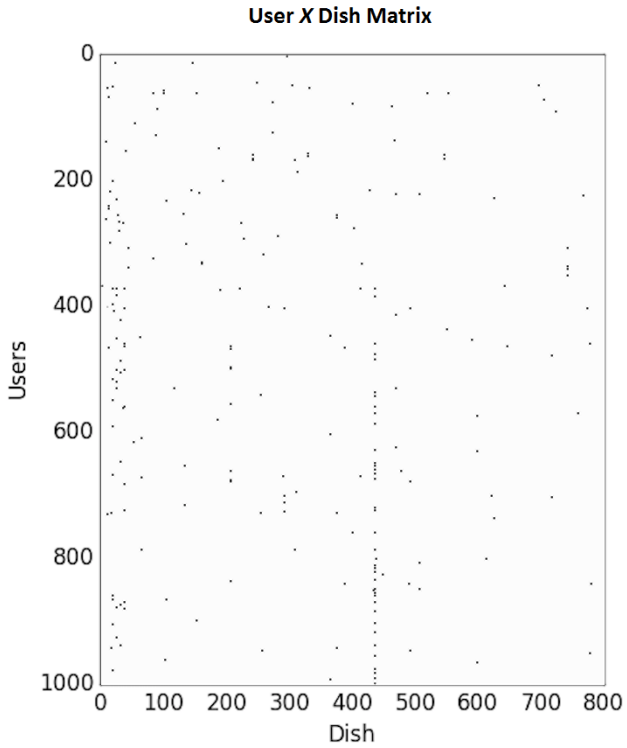

Figure 10: User X Dish Matrix.

8 Conclusions

- Using the dishes a user has reviewed as a basis for recommending a restaurant provides a personalized method for bringing in customers. With dish extraction and sentiment analysis, the preferences of a user and the specialties of a restaurant can be mined from a dataset of user reviews. Dish and restaurant recommendations can be produced that match a user's tastes and presented to the user in an intuitive manner [6].

- The quality of recommendations made can be further improved by using an ensemble of classifiers which use POS tags and sentiment lexicon knowledge as well. We believe that a effective way to judge the efficacy of our approach is to compare it with the recommendations from a $User \times Dish$ matrix. This will reveal if using explicit features enhances accuracy over relying latent features.

- This approach can be applied to other domains as well, where explicit features from review content can be used to enahance recommendation accuracy.[7]

- While the review data is rich in information, any approach that seeks to mine it needs to have domain specific knowledge built into it to improve its tractability for maching learning purposes[8].

- The project was implemeted in around 9 weeks using 4 developers. Primarily open source software was used for implementation [9].

9 References

[1] Michael D. Ekstrand, John T. Riedl., *Collaborative Filtering Recommender Systems*, Foundations and Trends in HumanComputer Interaction Vol. 4, 2011.

[2] M. J. Pazzani, D. Billsus,, *ContentBased Recommendation Systems*, 2007.

[3] C. Basu, H. Hirsh, W. Cohen, *Recommendation as Classification: Using Social and ContentBased information in Recommendation*, 1998.

[6] Heilmeier Question No.4
[7] Heilmeier Question No.5
[8] Heilmeier Question No.6
[9] Heilmeier Question No.7 and 8

[4] Peter Hajas, Louis Gutierrez, Mukkai S. Krishnamoorthy, *Analysis of Yelp Reviews*, 2014.

[5] Terry A. Slocum, Connie Blok, Bin Jiang, Alexandra Koussoulakou, Daniel R. Montello, Sven Fuhrmann, Nicholas R. Hedley, *Cognitive and Usability Issues in Geovisualization*, 2001.

[6] Dan Jurafsky, Victor Chahuneau, Bryan R. Routledge, Noah A. Smith, *Narrative framing of consumer sentiment in online restaurant reviews*, 2014.

[7] Andrew Carlson, Scott Gaffney and Flavian Vasile. *Learning a Named Entity Tagger from Gazetteers with the Partial Perceptron*, Carnegie Mellon University, Yahoo! Labs

[8] Alan Ritter, Sam Clark, Mausam and Oren Etzioni *Named Entity Recognition in Tweets: An Experimental Study*, University of Washington

[9] Richard TzongHan Tsai and ChunHui Chou., *Extracting Dish Names from Chinese Blog Reviews Using Suffix Arrays and a MultiModal CRF Model*, Department of Computer Science and Engineering Yuan Ze University, Taiwan.

[10] J. D. Lafferty, A. McCallum, and F. C. N. Pereira, *Conditional random fields: Probabilistic models for segmenting and labeling sequence data*, In Proceedings of the Eighteenth International Conference on Machine Learning, Morgan Kaufmann Publishers Inc., 2001.

[11] Robin Genuera, Jean-Michel Poggi,a,b, Christine Tuleau-Malotc. *Variable Selection using Random Forests*, Universite Paris-Sud 11, France Universite Paris 5 Descartes, France

[12] Charles Sutton and Andrew McCallum. *An Introduction to Conditional Random Fields. Foundations and Trend in Machine Learning*, Vol. 4, 2012.

[13] Alexander Pak, Patrick Paroubek, *Twitter as a Corpus for Sentiment Analysis and Opinion Mining*, Alexander , Universit e de ParisSud, Laboratoire LIMSICNRS, Batiment 508, F91405 Orsay Cedex, France

[14] Pang, Bo; Lee, Lillian; Vaithyanathan, Shivakumar, *Thumbs up? Sentiment Classification using Machine Learning Techniques*, Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 7986 2002

[15] Kushal Dave, Steve Lawrence, and David M. Pennock., *Mining the peanut gallery: opinion extraction and semantic classification of product reviews.*, In WWW 03: Proceedings of the 12th international conference on World Wide Web, pages 519528, New York, NY, USA. 2003

[16] Turney Peter, *Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews* , Proceedings of the Association for Computational Linguistics. pp. 417424, 2002

[17] Turney Peter, Littman L Michael, *Unsupervised learning of semantic orientation from a hundred-billion-word corpus*, Technical Report EGB-1094, National Research Council Canada, 2002

[18] Das Sanjiv,Chen Mike, *Extracting market sentiment from stock message boards.*, In Proc. of the 8th Asia Pacific Finance Association Annual Conference 2001

[19] Liu Bing, *Sentiment Analysis and Subjectivity*, Handbook of Natural Language Processing (Second ed.), 2010

[20] Koran Yehuda, Bell Robert, Volinsky Chris *Matrix Factorization Techniques For Recommender Systems*

[21] Funk SImon, *Netflix Update: Try This at Home*, http://sifter.org/ simon/journal/20061211.html., 2006

[22] Salem Marafi, *Collaborative Filtering*, http://www.salemmarafi.com/code/collaborative-filtering-r/, 2014