BT4014 Final Project Report

AY22/23 Semester 2

Restaurant Recommendation on Yelp

Alif Naufal Farrashady A0218302U

Chng Jin Yang Ray A0218512M

Chua Yee Siong Danyel A0222792A

1.1 Background

Yelp is an online review platform where users can leave a rating and detailed review based on their experiences with businesses. It provides a centralized space for businesses to feature their products and services, and for customers to rate and pen down reviews on the businesses. Since its establishment in 2004, Yelp has helped millions of businesses worldwide to advertise their offerings and increase revenue, as well as help even more customers gain better information about an incredibly rich collection of products and services that these businesses provide. A large proportion of these businesses belong to the restaurant industry which is a competitive market with many options for consumers on Yelp.

As seen in Appendix A Figure 1, restaurants are the leading business category on Yelp. To make informed decisions and pick out their dining experiences, consumers often rely on Yelp for existing information on restaurants. However, the vast amount of restaurants listed on Yelp may make it overwhelming for consumers to come to a decision. In this report, we will explore how bandit algorithms can be utilized to provide users with personalized restaurant recommendations and ease their decision making process.

1.2 Motivation

When it comes to recommendations of businesses, Yelp has implemented an automated recommendation software[1] that filters through businesses and provides what they believe is the most applicable product in response to a user's searches. This system does not guide individuals into a specific business as users would still have to search, sort, filter and weigh their options among the recommendation list to determine whether a restaurant is to their personal liking.

Furthermore, Yelp may be overly rigid in their recommendation algorithm, lacking the fresh element that exploration into lesser known businesses brings. Hence, to complement the already existing recommendation system, Yelp can implement bandit algorithms for personalized restaurant recommendations that can help them explore new options and recommend businesses that a user may not have discovered on their own. This could be achieved by showing users recommendations that are slightly outside their usual preferences, encouraging them to try new restaurants and expand their options.

Users will be given personalized recommendations to restaurants from a specific location of their search. For example, for users who will be visiting a new place on holiday, will be able to obtain curated suggestions for restaurants in the specific area according to their user profile.

The variety and spontaneity in recommendations would improve user experience, encouraging them to spend more time on the website and increase engagement as users are encouraged to explore more businesses on Yelp. As a result, there will be increased page views, longer session durations, and higher click-through rates on ads, increasing Yelp's revenue while improving customer experience on the website/app.

1.3 Problem Statement

[1] https://www.yelp-support.com/article/How-does-Yelp-determine-its-search-results

The problem that we aim to address would be the overwhelming number of restaurant options on Yelp and rigidity of recommendations for users. By framing the restaurant recommendation problem as a multi-armed bandit problem, it is possible to apply bandit algorithms to provide personalized restaurant recommendations to users, based on a restaurant's previous ratings and reviews, along with the user profile while maintaining a level of spontaneity by balancing exploration and exploitation of restaurants.

Additionally, recommendations through bandit algorithms can drive business to quality restaurants that may not have as many reviews or ratings as larger chains, providing the chance for exploration. This will improve the diversity of reviewed restaurants on Yelp. Ultimately, the use case for bandit algorithms in Yelp's case would be to have a live feedback loop on its website that allows users to rate and provide feedback on recommendations which can help us understand whether a user responds well to the explorative restaurants.

1.4 Highlights of Key Findings and Implications

After evaluating a pool of bandit algorithms using evaluation replay on two separate restaurant review datasets segregated by location; Las Vegas and Toronto, we observed that Softmax performed the best, achieving the highest cumulative reward and average reward for both datasets. This could have been due to the softmax function allowing exploration for arms with lesser expected reward with a non-zero probability, allowing for a smooth tradeoff between exploitation of the current best arm and exploring other arms. Performance for contextual bandits across the datasets were inconsistent, and could be attributed to limited breadth of user features that could be generated with the open source data Yelp provided, although the LinUCB algorithm showed potential in delivering personalized recommendations. We still noted a potential for bandit algorithms for allowing chances for exploitation in recommending restaurants to users, where Yelp can utilize and improve on, capitalizing on more user features and data available to them.

2.1 Data and Context

Our Primary dataset is from the Yelp Open Dataset (source: https://www.yelp.com/dataset), a popular dataset consisting of 6 json files which describe real-world users, businesses, reviews, tips and check-in information from Yelp. For the purpose of this project, we will only use 3 out of 5 files describing businesses, reviews and users - business.json, review.json and user.json. In the dataset, there are 6,990,280 reviews, 150,346 businesses and 1,987,897 users.

With the sheer size of the dataset, we decided to confine our analysis to only reviews and businesses in Las Vegas and Toronto as proofs of concept. These two regions were shortlisted for analysis as they have the largest number of reviews compared to the other metropolitan districts available in the Yelp Dataset. We further filter the businesses to contain only restaurants as our project focuses on restaurant recommendations.
Even with the filtering of locations for our analysis, there still exists a huge amount of restaurants that have been reviewed on Yelp for both Las Vegas and Toronto, 5899 and 7148 restaurants respectively. Utilizing bandit algorithms with that amount of arms will complicate the

problem and require very high computational power to train the bandit algorithms. For our analysis, we aim to end up with around 20-40 restaurants for each region, to balance between problem complexity and computational runtime. To further filter the pool of restaurants for our analysis on Las Vegas and Toronto, we look at the average number of stars and the review count of the restaurant.

The stars that a restaurant can have from a single review ranges from a categorical scale from 1 to 5. Considering that if a restaurant has on average stars below 3, a significant portion of its customers provided a below average rating, and likely signals inherent dissatisfaction with the restaurant, be it in terms of food quality, or customer service. It may not make sense to recommend such restaurants. Thus, we will keep restaurants that have at least an average star rating of 3.

We also want restaurants analyzed to have sufficient reviews, so that we have adequate review data that we can train our bandit algorithms on. Las Vegas has more restaurants that have a high review count as compared to Toronto. For Las Vegas, we filtered for restaurants with at least 2000 reviews, while for Toronto, restaurants were filtered for at least 500 reviews. After all the filtering, we will be working with 38 restaurants for Las Vegas, and 24 restaurants for Toronto.

2.2 Feature Engineering

Initially, the ratings given in each review is an ordinal categorical variable, going from 1 star to 5 stars. This initial rating will be converted into a binary variable, with 1 representing reviews that gave 4 or 5 stars, and 0 representing 3 stars. This was done for a few reasons. Firstly, this allows for clear separation in the sentiment towards the restaurant. Ratings with 4 or 5 stars are clearly positive, while those with 3 or lower stars are not. This is in alignment with the objective of the recommendation system, which is to deliver recommendations for users that they will enjoy, which can be defined as having a positive sentiment. As such, this new binary variable for the rating can simply be interpreted as to whether the user enjoyed the restaurant or not. Secondly, this helps to solve the issue of consistency amongst how different users will rate restaurants. Different users will have different standards of what constitutes a 5 star review and the same experience from the same restaurant can be interpreted differently by different users. Hence, generalizing into positive and non-positive experience will help to reduce the impact of subjectivity in the ratings. Lastly, representing the rating as a binary variable will allow some bandit algorithms to model the reward distribution as a Bernoulli variable, which is computationally convenient. This will be further discussed later in the report.

Each business has a ‘categories’ attribute, which in the case of restaurants, can include information such as the cuisine served at the restaurant. After looking through the various categories available, 23 of them were selected, based on whether they made sense and whether there were many restaurants having that category, such that niche categories with very little restaurants are excluded. Overlapping or similar categories were combined into a singular feature. For example, the categories ‘American (New)’ and ‘American (Traditional)’ are merged into 'American’. Besides cuisines such as ‘American’, ‘French’, ‘Italian’, other information was

captured, such as the style of the restaurant, like ‘Bakeries’, ‘Cafes’, ‘Music Venues’ and ‘Nightlife’. Furthermore, each business also has a ‘neighborhood’, which will be added as another feature, to capture any geographical patterns in the restaurants, such as in the case where certain neighborhoods have better restaurants. This will be represented as a set of dummy variables. The full list of unique neighborhoods and cuisine attributes for Las Vegas and Toronto can be found in the Appendix.

Now that each restaurant has the cuisine features, these will be used to generate user features. Since each review is linked to a restaurant and to a user, it is possible to capture the preferences of each user with regards to cuisine. This is done by aggregating the reviews by user and summing over the cuisine features. To account for differing total number of reviews, the ratio of each cuisine visited is calculated by dividing over that user’s total number of reviews. This should be able to capture the user’s preferences more accurately. For example, if a user has 4 reviews and 2 of those reviews were for Thai restaurants, the ‘user_Thai_ratio’ feature will be 0.5. Note that since each restaurant can span multiple categories, the ratios across the cuisines need not sum up to 1.

Additionally, the standard deviation in terms of ratings for each user is calculated and added as an additional user feature. This is meant to capture the volatility of the ratings for each user.

After feature engineering, each review will now include the restaurant features, which consist of the cuisine and neighborhood features, along with user features, which consist of the cuisine ratios along with the standard deviation of the ratings. The rating of the review has also been modified to be a binary variable. These transformations will be applied to the Las Vegas and Toronto reviews, giving us 2 separate sets of reviews to work with. The final Las Vegas dataset has 88854 reviews, while the final Toronto dataset has 15573 reviews. The Las Vegas dataset has significantly more reviews because Las Vegas restaurants have much higher reviews per restaurant.

3.1 Methodology

As we are unable to perform online evaluation to run our recommendation system, we will instead perform offline evaluation using evaluation replay on our dataset, using a variety of bandit algorithms, including contextual bandits. Due to the high number of reviews, bootstrapped sampling is not used as it will introduce longer training time and the current dataset is large enough to provide a representative sample of the general Yelp user population. 2 separate rounds of evaluation replay will be run, one each for the Las Vegas and Toronto set of reviews.

After filtering the reviews and restaurants, along with adding the restaurant level and user level features through the feature engineering process mentioned earlier, evaluation replay is conducted on the reviews. Each restaurant is one arm. At each time step, a review is shown to the bandit algorithm, which will recommend a restaurant. For contextual bandits, both the restaurant level and user level features will be given as well. The idea is that the restaurant features, which are at the arm level, will allow the contextual bandits to better model the reward

distribution of the restaurant, while the user features will allow the contextual bandits to deliver personalized recommendations, taking into account the taste and preferences of each user.

If the recommended restaurant matches the actual restaurant on the review, the bandit algorithm will update itself based on the reward, which is the stars of the review, which as mentioned earlier, reviews with 4 or 5 stars have a value of 1, while those with lower stars have a value of 0. If it does not match, the algorithm does not update and that review is 'discarded'.

The following bandit algorithms will be used to conduct the evaluation replay:

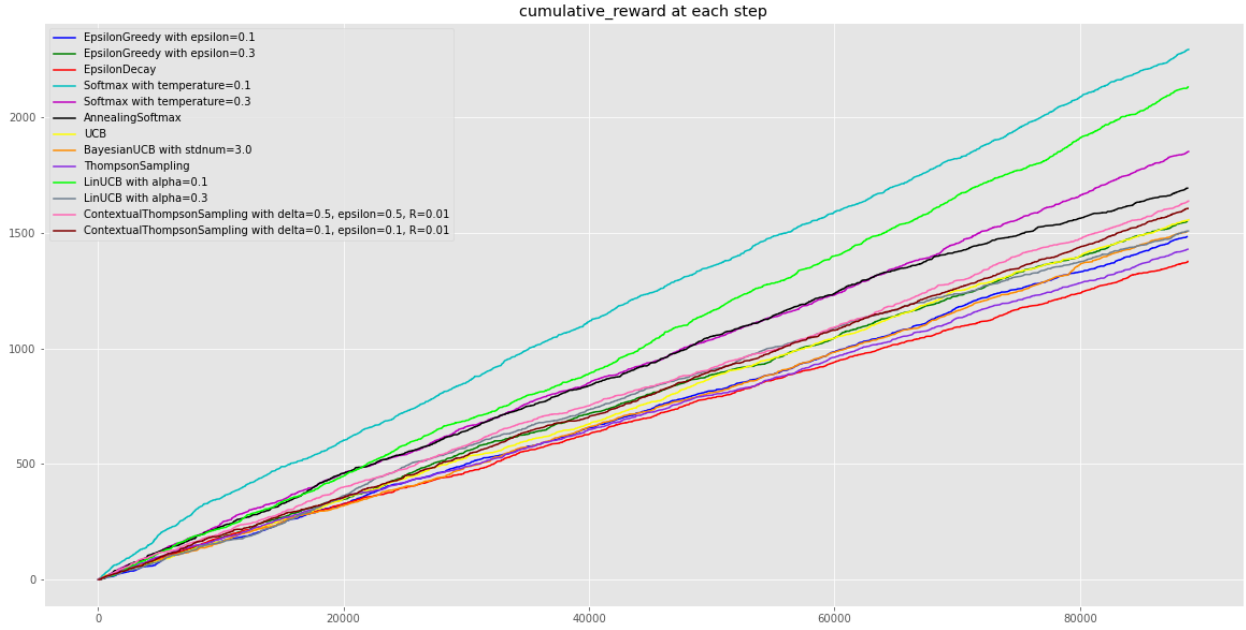
Model name	**Parameters used**
Non-contextual Bandit Algorithms	
Epsilon-greedy	epsilon=0.1 epsilon=0.3
Epsilon-decay	
Softmax	temperature=0.1 temperature=0.3
Annealing Softmax	
Upper confidence bound (UCB)	
Bayesian UCB	stdnum=3
Thompson Sampling	
Contextual Bandit Algorithms	
Linear UCB (LinUCB)	alpha=0.1 alpha=0.3
Contextual Thompson Sampling	delta=0.1, epsilon=0.1, R=0.01 delta=0.5, epsilon=0.5, R=0.01

As mentioned earlier, the ratings in the reviews have been modified to be a binary variable and we will be assuming that the reward follows a Bernoulli distribution. Bayesian UCB and Thompson Sampling will make use of this assumption and use the Beta distribution as the conjugate prior. The prior alpha and betas for both these models will also be set to 1, thus utilizing a uniform prior. Additionally, for the contextual bandits, the model parameters are disjoint.

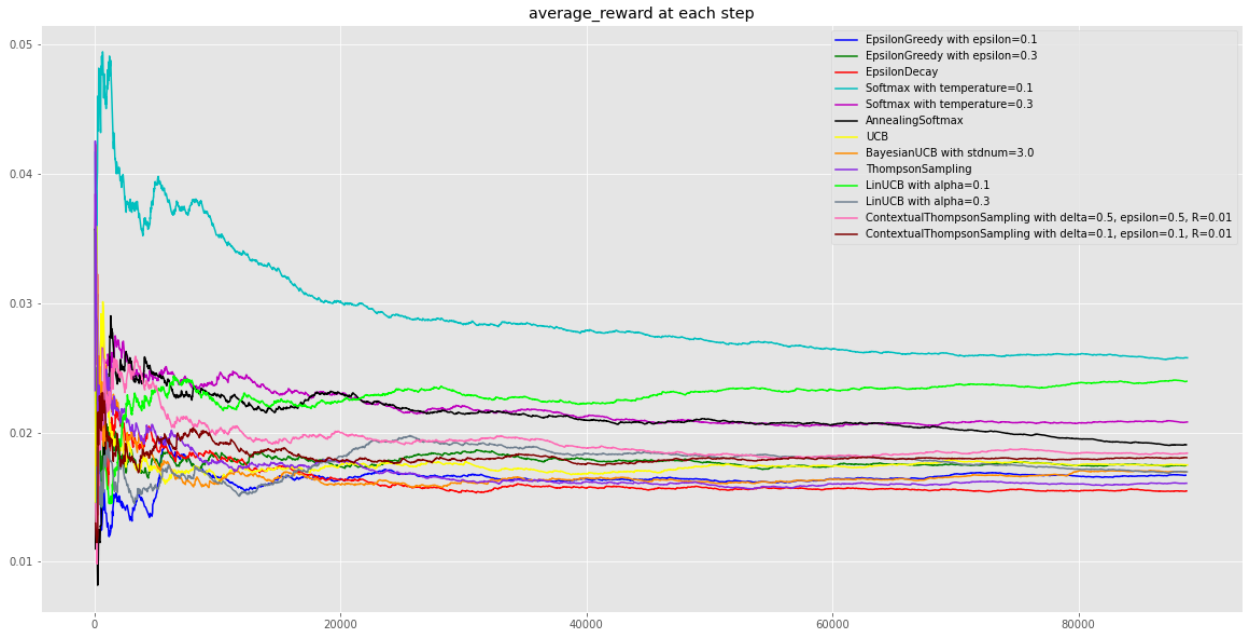
It should be expected that the contextual bandits perform better, given that they have access to the restaurant and user level features, which should allow them to model the rewards of the arm more accurately, given the profile of the user. This allows the contextual bandits to deliver personalized recommendations, which should make for a more effective recommender system. On the other hand, the non-contextual bandit algorithms are unable to take advantage of the availability of the features and will not be able to deliver personalized recommendations. The various bandit algorithms will be assessed based on the average reward and cumulative reward, which will be plotted against time step, allowing for analysis on how these metrics have changed over time.

3.2 Performance analysis

3.2.1 Las Vegas

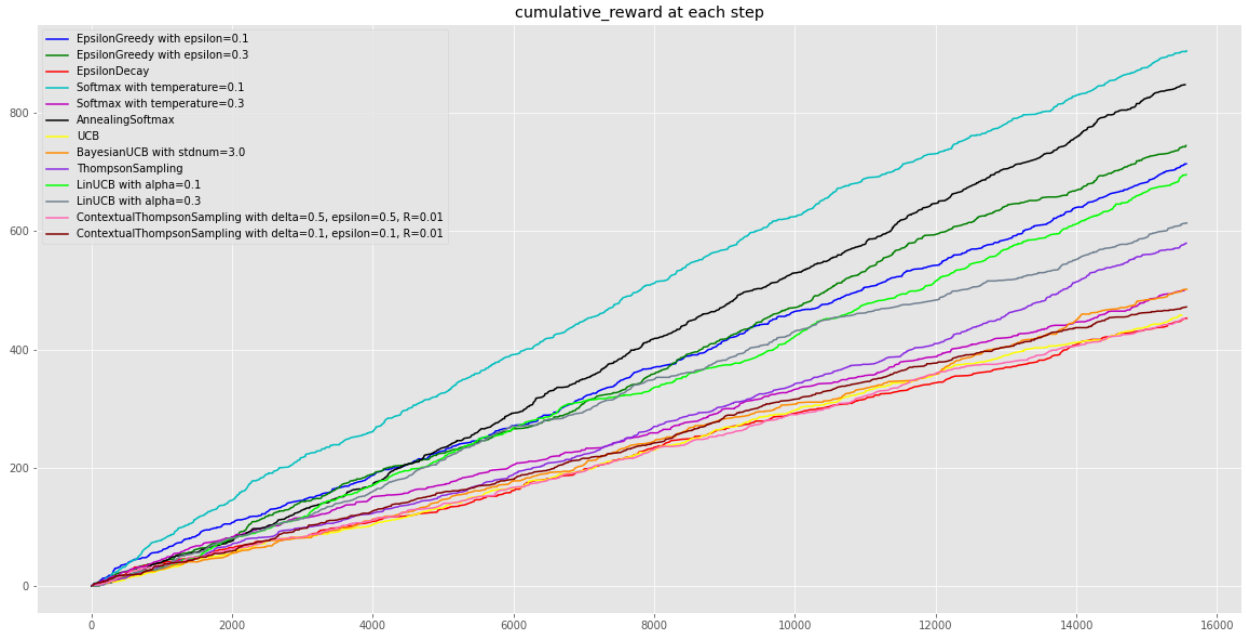


For Las Vegas, in terms of cumulative reward, Softmax (temperature=0.1) was by far the best performing algorithm, followed by LinUCB (alpha=0.1). Subsequently, we have Softmax (temperature=0.3) and Annealing Softmax. After that is Contextual Thompson Sampling (delta=0.1, epsilon=0.1, R=0.01) followed by Contextual Thompson Sampling (delta=0.5, epsilon=0.5, R=0.01). The poorest performing algorithms are Epsilon-decay and Thompson Sampling. Outside of the 3 Softmax variants, the best performing algorithms are the contextual bandits, which is a good sign as it suggests that the features were useful in understanding the distribution of the rewards for the arms. In terms of the hyperparameters, for Softmax and LinUCB, lower values of temperature and alpha performed better, suggesting that favoring exploitation over exploration was better. On the other hand, for Epsilon-greedy and Contextual Thompson Sampling, higher values of epsilon, and delta and epsilon, respectively, did better, which meant that more exploration was beneficial.

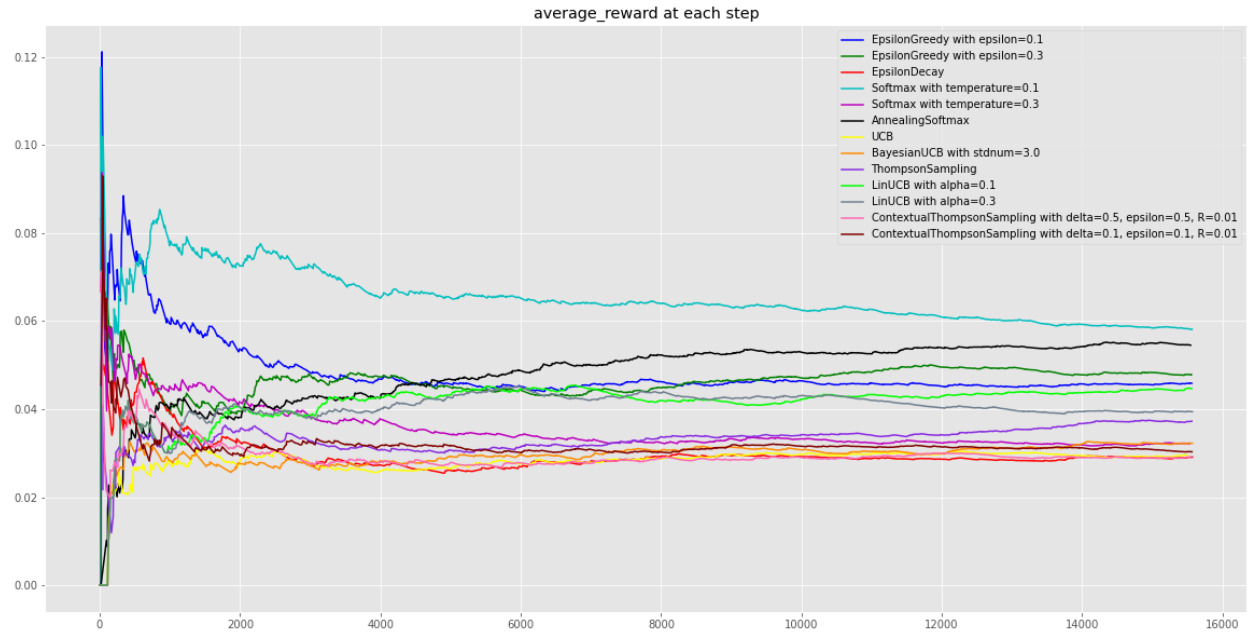


For average reward, the order follows the same as cumulative reward. In terms of convergence, Softmax (temperature=0.1) actually takes a longer time compared to the other algorithms, all of which generally stabilizes at around the same time. Once an algorithm converges, it is unlikely that further exploration will lead to significant improvements in rewards. It is possible that Softmax (temperature=0.1) takes a longer time to learn the underlying patterns in the rewards of the arms, but this process has allowed it to achieve better long-term performance, as compared to the others which converged earlier and plateaued at a lower level of reward.

3.2.2 Toronto



For Toronto, in terms of cumulative reward, Softmax (temperature=0.1) also came out on top. There are variations to the subsequent rankings as compared to Las Vegas. Annealing Softmax was the next best performing, followed by the two Epsilon-greedy algorithms, and subsequently LinUCB (alpha=0.1). It turns out that the poorest performing algorithms are Epsilon-decay, UCB and Contextual Thompson Sampling (delta=0.1, epsilon=0.5, R=0.01). Softmax (temperature=0.1) once again had a big gap over all other algorithms early on and gained the highest cumulative reward. In this case however, the contextual bandits did not perform very well, with the best performing contextual bandit, LinUCB (alpha=0.1), ranking 5th at the end of the evaluation replay. The two Contextual Thompson Sampling algorithms performed poorly. However, this could be due to the Toronto dataset having significantly lesser reviews, as compared to Las Vegas, due to Las Vegas restaurants having a higher review counts. More reviews in the Toronto dataset could have allowed the contextual bandits to improve with more information about the features. In terms of the hyperparameters, the results are the same as Las Vegas. For Softmax and LinUCB, lower values of temperature and alpha also performed better. For Epsilon-greedy and Contextual Thompson Sampling, higher values of epsilon, and delta and epsilon, respectively, performed better.



For average reward, the order follows the same as cumulative reward. In terms of convergence, due to the comparatively smaller dataset for Toronto, we see slightly more fluctuations. Softmax (temperature=0.1) no longer seems to take the longest time to converge. If we look at the other algorithms, like Annealing Softmax, there are still movements in average reward across the time steps. We cannot say for sure if Softmax (temperature=0.1) will converge to a higher reward as compared to Annealing Softmax when the time steps are larger.

3.2.3 Summary of Findings

Taking the findings from both the Las Vegas and Toronto simulations, it is clear that Softmax is the best performing algorithm. One possible reason as to why Softmax was able to do well is due to how the softmax function allows for a smooth tradeoff between exploitation of the current

best arm and exploration of the other arms, where it allows for exploration of less promising arms with a non-zero probability. For many of the other bandit algorithms, their approach to choosing the arm is through 'argmax', even though they implement an exploration bonus, once the sum of expected reward and bonus is calculated, they will pick the highest arm with probability of 1. Softmax will only assign a higher probability to the arm with the highest expected reward, such that it can still end up exploring other arms. Furthermore, a temperature of 0.1 was able to do better than the Annealing Softmax variant, suggesting that a constant exploration rate was beneficial, as compared to a declining one.

Also, it can be noted that Bayesian UCB and Thompson Sampling performed poorly compared to the other non-contextual bandit algorithms. Again, it might be due to an issue in the assumption of the probability distribution. Bayesian UCB and Thompson Sampling assumed that the reward followed a Bernoulli distribution. It is possible that this assumption is inappropriate and that the reward does not truly follow the Bernoulli distribution. This might explain the relatively poor performance of these algorithms. One alternative would have been to use the original rating scale of 1 to 5 and model the reward as a Gaussian distribution instead, or given the discrete nature of the ratings, as a Multinomial distribution. Nevertheless, as mentioned earlier, there are strong justifications for representing the ratings as a binary variable.

With regards to the contextual bandits, LinUCB was able to outperform Contextual Thompson Sampling for both Las Vegas and Toronto. One reason as to why this might be the case is that Contextual Thompson Sampling uses the multivariate Gaussian to sample the parameter vector $\tilde{\mu}$. It is possible that the multivariate Gaussian is not a suitable distribution to use in this case and hence Contextual Thompson Sampling performs poorly compared to LinUCB, which makes no assumption about the distribution of the rewards. Nevertheless, LinUCB performed well overall, which suggested that it was able to utilize the given context of the restaurant and user features to deliver personalized recommendations.

As for the model hyperparameters, there is no clear pattern when looking at the results. For some algorithms, parameters that favor more exploration did better, while for others, the opposite is true.

3.3 Limitations

There are few limitations in the proposed recommendation system utilizing bandit algorithms. Firstly, with regards to the dataset, with more features, it could have been possible to improve the performance of the contextual bandits further. For example, we did not have access to certain user data, such as gender, age and nationality, which could have been used to model the tastes and preferences of the users more accurately.

Secondly, the contextual bandits used disjoint model parameters. It might be possible that using joint model parameters for certain features may have been more appropriate and could have improved performance. However, experimenting with this will be a bit tricky and it will be

computationally expensive to try multiple variations where certain features are joint and others are disjoint.

Lastly, it might be possible we did not use the optimal hyperparameters for the various bandit algorithms, which could have improved performance. However, similar to the limitation on disjoint model parameters, further tuning on the model hyperparameters will take more time. Given that the hyperparameters are continuous, it is difficult to ascertain which values to test.

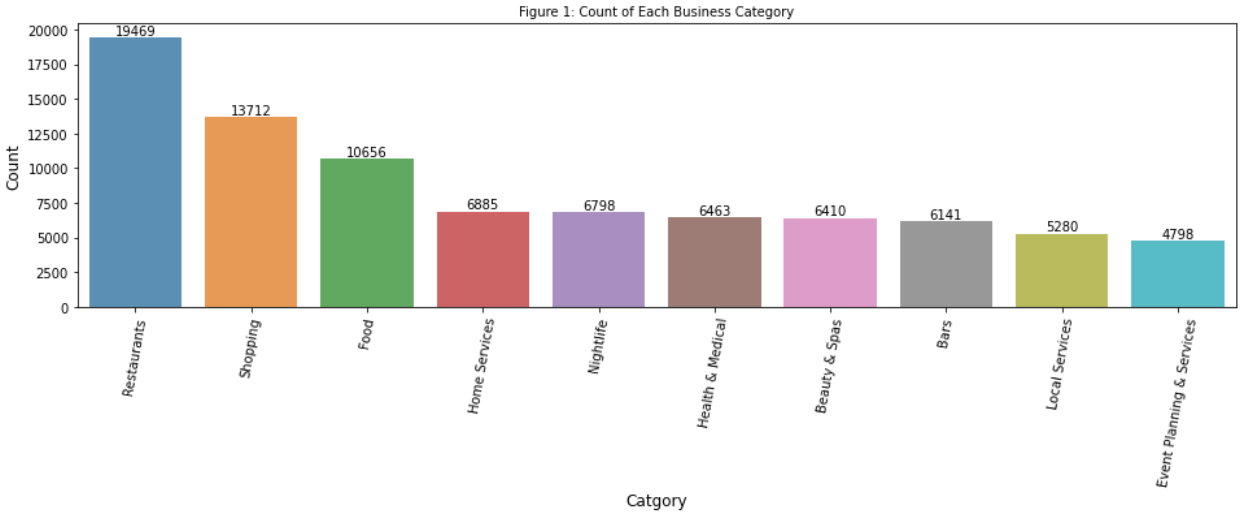
4. Conclusion

Overall, taking the findings from both the Las Vegas and Toronto results, there is clearly scope for using bandit algorithms to deliver recommendations towards Yelp users. The ability of bandit algorithms to explore different restaurants allows for varied recommendations, while also being able to exploit restaurants that have a track record for positive reviews. The Softmax algorithm was the clear standout, likely due its ability to explore arms with lower expected rewards as it assigns them a non-zero probability. The performance of contextual bandits varied across the datasets, possibly due to the limited number of user features that could be generated from Yelp's open source data. Nevertheless, the LinUCB algorithm showed potential in leveraging a user’s profile to provide personalized recommendations. With the availability of additional user data and the chance to perform online evaluation, it is possible that bandit algorithms can allow Yelp to deliver an improved user experience by supplying suitable restaurant recommendations.

Appendices

A. Chart for Businesses Types on Yelp

In Appendix A, the figure states the number of each business categories on the Yelp dataset. As seen, Restaurants are the leading business category on Yelp with 19469 businesses available on Yelp. Other categories include shopping, home services, bars and more.



B. Data Columns

In Appendix B, the available information from the data we used are in the columns as stated below, from each dataset respectively.

1. business.json contains details of each business with the following columns:
   `'Business_id','name','neighborhood','address','city','state', 'postal_code','latitude','longitude','stars','review_count','is_open', 'Categories'`
2. Reviews.json contains reviews for all categories with the following columns:
   `'review_id','user_id','business_id','stars','date','text','useful', 'funny', 'cool'`
3. User.json contains information of a Yelp user's account with the following columns:
   `'User_id','name','review_count','yelping_since','friends','useful', 'Funny','cool','fans','elite','average_stars','compliment_hot', 'Compliment_more','compliment_profile','compliment_cute', 'Compliment_list','compliment_note','compliment_plain', 'compliment_cool','compliment_funny','compliment_writer','compliment_photos'`

C. List of Neighbourhoods and Cuisines for Restaurants in Las Vegas and Toronto

In Appendix C, the full list of neighborhoods and cuisines for the restaurants in Las Vegas and Toronto is as stated in the table below.

Las Vegas		Toronto	
Neighborhoods	**Cuisines**	**Neighborhoods**	**Cuisines**
Chinatown	American	Distillery District	American
Downtown	Asian Fusion	Downtown Core	Asian Fusion
Eastside	Bakeries	Entertainment District	Bakeries
Southeast	Bars	Financial District	Bars
Westside	Breakfast & Brunch	Kensington Market	Breakfast & Brunch
The Strip	British	Liberty Village	Chinese
	Buffets	Parkdale	Desserts
	Burgers	Ryerson	Italian
	Cafes	Queen Street West	Japanese
	Caterers	Wychwood	Korean
	Chinese		Mexican
	Desserts		Nightlife
	French		Seafood
	Italian		Thai
	Japanese		Vietnamese
	Korean		
	Mexican		
	Music Venues		
	Nightlife		

	Salad		
	Seafood		
	Thai		
	Vietnamese		