Team Name: SSHH
Team Members:
UNI: hs3159 **Name**: Harguna Sood, **github id**: harguna
UNI: hj2533 **Name**: Hritik Jain, **github id**: hritik25
UNI: sk4661 **Name**: Sachit Kumar, **github id**: sachit-kumar
UNI: sv609 **Name**: Siddhanth Vinay, **github id**: sidvin97

REPORT

About the Yelp Dataset:

https://www.yelp.com/dataset/challenge

- Number of ratings: 6,685,900
- Number of Users: 1,637,138
- Number of Businesses: 192,609

Though the Yelp Dataset contains businesses of various categories, for this project we build a recommender system to recommend Restaurants/Food businesses to users.

Objectives:

Our objective is to recommend restaurants/food businesses which we think will be the most relevant to a user. For this, we take into account factors such as the time of the day and the most probable city a user currently is in to make our recommendations. Our hope is that our recommendations are accurate and users will go to our recommended businesses. Alongside, we also want to cover as many businesses as possible while making recommendations, which we will measure through coverage.

Sampling methodology:

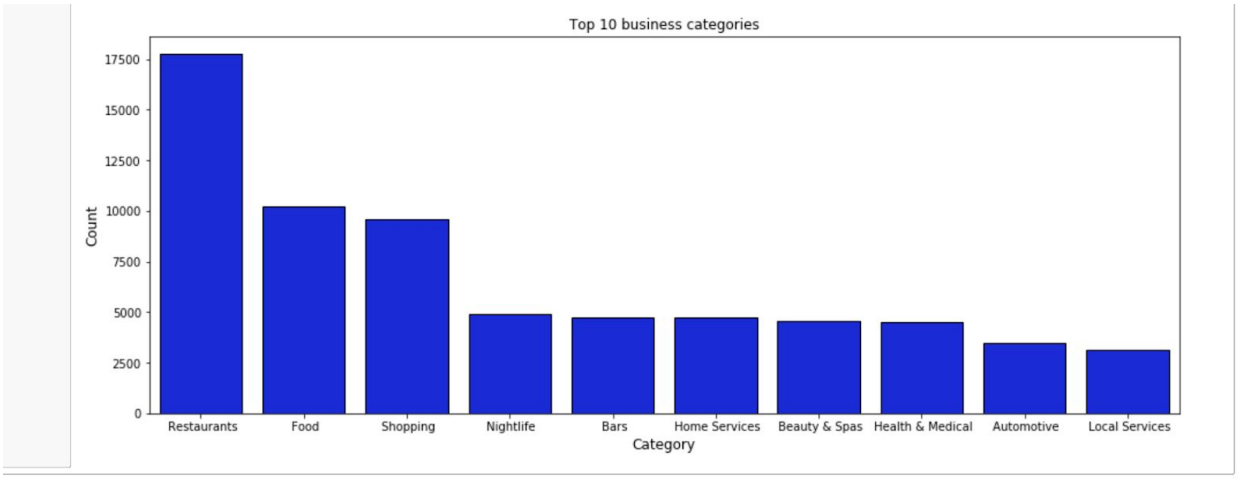
To achieve our objective, our sample contains only those businesses which have both “Food” and “Restaurant” listed in the business category field. We initially planned to consider a sample of all the businesses in only one city and use those businesses for our model. However, we felt that the model would become city specific. Since we wanted to build a generalized recommendation system, where we can also use the location factor in training our models, it made sense to sample using our current sampling strategy instead.
We finally filter for only the active users who have rated these businesses (users who have rated at least 5 of our sample businesses).

Sampling Statistics:

- Number of ratings: 629,155
- Number of Users: 52,550
- Number of Businesses: 20,824

The reasons why we choose to recommend both food and restaurant businesses:

1. After an EDA of the Yelp dataset, we see that the top 2 categories of businesses that were reviewed are food/restaurant.



2. It seems very intuitive that users would want recommendations of restaurants and food businesses as part of their everyday life. It also seems intuitive that if a particular user rates a gym/hardware store highly, he would stick to their service rather than exploring other/newer options.
3. When we individually filtered on Food/restaurant, we observed that our sample contained businesses with spurious categories such as Health spa and gym. This would thus not be very representative of what we are trying to recommend, i.e., restaurants/food businesses.
4. Since this sample has at least 50000 users and 20000 businesses, we feel that this sample is a large enough representation to accurately train and test our models on.

Metrics:

We use the following metrics to evaluate our models:

- **R-squared**: This helps us understand how well our model performs compared to a baseline model which predicts average of the user ratings.
- **RMSE**: This helps us evaluate the accuracy of our models. Predictions which are far off from the actual ratings will be exaggerated by RMSE. It is sensitive to outliers.
- **Catalog coverage**: Gives the percentage of the movies in the dataset present in any of the top k recommendations we give a user. This helps us evaluate what percent of the

businesses are being recommended to a user, that is - how many businesses are we helping by recommending them through our model.

Models:

In this section, we describe the models we used and the methodology behind them. Our model exploration aspect is covered inside the notebooks.

- **Wide and Deep Learning Model:** Jointly trains a *wide* linear model component (for memorization) and a *deep* neural network component (for generalization). Memorization means memorizing everyday events (eg, sparrows/pigeons can fly) and generalization means generalizing these learnings to apply to newer things (eg, animals with wings can fly).
  We use the following feature set for building our wide and deep model:
    - Wide component:
        - A feature set which comprises a multi-hot encoding of categories that a business corresponds to (we consider only categories that correspond to at least 500 businesses so that it is feasible to train our model) - we end up with a set of 35 categories.
    - Deep component:
        - Continuous features - We use features such as the average user rating, the average business rating, the number of ratings a business has and the number of fans a user has
        - Categorical embeddings - We pass state and city as categorical embeddings for the deep component of the model.
        - We add the following derived features as well to the deep component of our model:

            - **Derived User Features:** We are making new features the represents the taste of users. In this, we will try to see how many times the user has rated a restaurant with a particular category.e.g if the user has rated Italian food the most, this will give us some sense that user likes Italian food. We will make columns for all the categories we have and in each cell will be the number of times the corresponding user has rated it.

            - **Derived Business features:** We are also making new, derived features to represent businesses. The idea here is we give more weightage to the ratings where more people found it useful and then get weighted average for the

business based on that.e.g. there is a business A which has two reviews: 4.5 and 1, and 100, 10 people found them useful respectively. Here the weighted average for that business will be: (4.5*100 + 1*10)/110 = 4.18. This is an idea to capture the more relevant reviews to get the average rating for the business.

- **Field-aware Factorization Machine Model (FFM) model:**

  Factorization Machines are a general predictor working with any real valued feature vector. For user-business rating pairs, the real valued vector is created by concatenating the one-hot-encodings of user_id and business_id, and similarly the one-hot-encodings for any other categorical features which we include in our model. The numerical features are appended to the real valued vector directly.

  FMs solve the problem of considering pairwise feature interactions. It models pairwise feature interactions as the dot product of low dimensional vectors(length = k). In contrast to SVMs, FMs model all interactions between variables using factorized parameters. Thus, they are able to estimate interactions even in problems with huge sparsity (which is the case with one-hot encoded feature vectors), where SVMs fail. Also, the model equation of FMs can be calculated in linear time and thus FMs are very fast to train and optimized.

  FFMs:
  In FFMs, each feature has several latent vectors. For example, in our case, when we consider the interaction term for *Las Vegas* and *evening* (city and time_of_day fields respectively), the hidden feature for *Las Vegas* would have the notation $w_{\text{Las Vegas, evening}}$, where evening represents the field for the feature *evening*. Similarly for user_id field, a different parameter $w_{\text{ESPN,<user_id>}}$ would be learnt. FFMs have proved to be vital for winning the first prize of three CTR (Click through Rate) competitions in recent past, and appeared to be well suited to our problem.

- **ALS Baseline:** We build an ALS baseline model that is a collaborative filtering technique commonly used for recommender systems. It aims to fill in the missing entries of a user-item association matrix in order to make recommendations.

- **Bias only baseline:** This is a model used to make predictions by simply taking into account the global bias, an individual user's bias and an individual business's bias.

Business rule:

We take all the businesses that have not been recommended to any user by our models and that have a rating of above 3.5, and randomly recommend 3 of these businesses from the user's city additionally to each user. This will simultaneously not only improve the coverage, but will also improve serendipity, since our model is trained towards a user's preference and this will enable us to recommend good businesses that are outside a user's normal taste.

Results:

	RMSE	**R2**	**Catalog Coverage**	**Catalog Coverage with Business Rule**
Wide and Deep	1.11	0.32	52.06%	78.51%
Field aware FFM	1.20	0.21	13.64%	74.57%
ALS baseline	1.34	0.01	8.86%	-
Bias only baseline	1.43	-0.11	100%	-

Limitations:

- **Wide and Deep model:**
  - Could not consider all the businesses for making recommendations and evaluating coverage due to lack of computational power.
  - Could not consider all categorical features to train the model due to lack of computational power.
- **FFM:**
  - We did not utilize user and business specific features to predict ratings. These features can be included in the model to make the recommendations more accurate.

Conclusion:

After building and evaluating our models, we see that our models meet our objectives. Recommendations seem to be accurate in both models judging by the RMSE and R-squared, while outperforming the baseline models. The original catalog coverage of Wide and Deep is very good, while the catalog coverage of Field-aware FFM is low. However, after adding our business rule, we see that the overall catalog coverage for both models are a lot and thus meet our objective of covering a large proportion of the total businesses.

If we were a company we feel that our models would be a good starting point. We could further build on our models by including more user and business related features. (Eg, in the FFM model we could include business stars, the ratings for each kind of category, etc). We could also perhaps build an ensemble of our two models which could potentially improve the accuracy of our predictions further.