Rectastic : Sakshi Bhargava, Piyush Bhargava, Chhavi Choudhury

Progress Report: Yelp Recommendation Engine

1. Executive Summary:

Yelp Business Rating prediction is essentially a cold start problem, as large proportion of users and businesses are new in the test set. This makes the problem uniquely challenging. The dataset is segmented into cold start, warm start and hot start recommendation problem and we try to get best performance of each segment. The baseline model with complete dataset gives a poor RMSE of 1.39. However, ALS on hot-start data, improves the performance to 1.29. A preliminary Random Forest model (RMSE of 1.295) also performs better than the baseline model. These initial results are promising and we plan to improve performance by concentrating on each segment of data separately. We are doing this project on Spark and the algorithms are implemented in scalable way.

2. Data Exploration & Analysis Results:

The data available on Kaggle is a detailed dump of Yelp reviews, businesses, users, and check-ins for the Phoenix, AZ metropolitan area. The data overview is as follows:

	Training Data	Test Data
Distinct User Count	45,981	11,926
Distinct Business Count	11,537	5,585

Table 1: Data Overview

Being a recommendation problem, the test data consists of all the four cases of known/unknown user and business information (cold start, warm start and hot start cases) and the distribution is:

	Known User	Unknown User
Known Business	33.2%	41.1%
Unknown Business	11.2%	14.5%

Table 2: Known/Unknown User and Business segments in Test Data

The global average rating from reviews in training data was observed to be 3.76. Besides adopting different techniques for these four subsets of test data, we will also be doing feature engineering to enhance our results. To identify the critical features that will potentially enhance our results and for other useful insights, we performed EDA consisting of univariate summary data and bivariate analysis on the data. The key inferences and findings from the analysis are:

- Occurrence of low ratings, in the 1−2 stars range, for a business that has a majority of ratings in the 4−5 range. These lower ratings are said to "sandbag", which account for a good portion of error in our predictions.
- Instability of the mean predictors is due to the difference in the amount of user information and the amount of business information. This results in bias in predictions towards information rich entity among the user-business pairs.
- Clustering based on city and categories of business was intuitive from the distribution of ratings across cities and categories of business, indicating segments of similar behavior

3. Feature Extraction

We have performed certain feature extractions such as creating dummy variables to capture all the categories that a business is part of. This was needed since a business can belong to multiple categories such as 'restaurants' and 'pizza'. In total, there were ~550 categories across train and test datasets (Refer Fig 1 below). We plan to continue working on identifying other impactful features by segments.

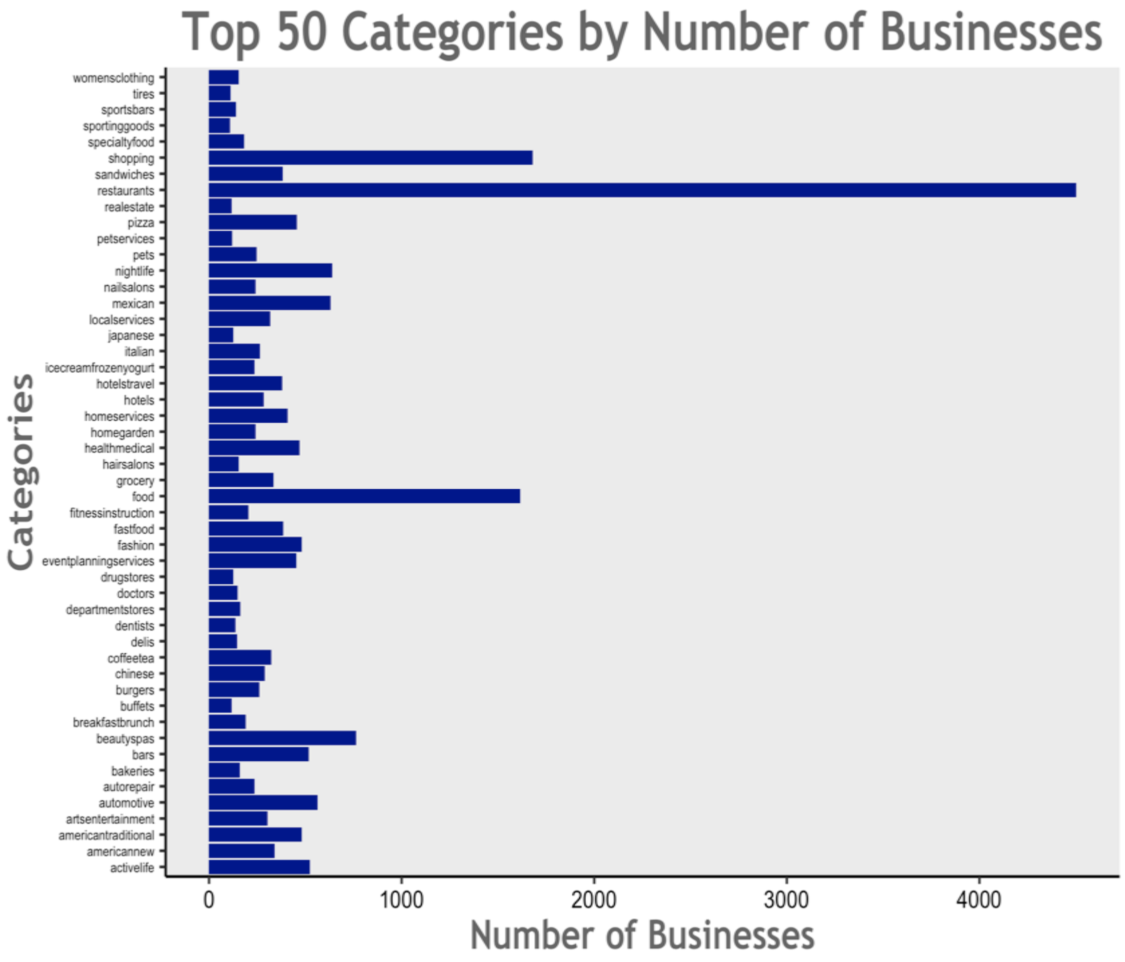

Figure 1: Distribution of Categories in the dataset

4. Baseline Model

Our baseline model takes into account the mean ratings of the users and the mean ratings of the businesses to predict a rating. We notate these as follows: **μ** is the average rating for all user review ratings included in the training set (for the Yelp set **μ** = 3.7667). ***u*** is the vector containing average ratings for each user and subtracting **μ**. ***b*** is the vector containing average ratings for each business and subtracting **μ**. ***R*** is the predicted ratings matrix. We then uses the following equation to predict user ***i*** 's rating for business ***j*** . Using this predictor, we achieved a **Kaggle RMSE of 1.39.**

$$R_{ij} = \mu - u_i - b_i$$

5. Singular Value Decomposition using ALS

Singular Value Decomposition (SVD) is a latent factor method. A matrix is approximated with SVD by multiplying two generated feature matrices **P** and **Q** with rank **k**, such that the product PQ^T is close to the original matrix. For example, assume $\mathbf{R_{ij}}$ corresponds to the rating for ***user***$_i$ and ***business***$_j$. We then minimize the squared differences between the ratings matrix R and the SVD approximated matrix, $\widehat{R}$. To make the predictions more accurate we subtract **μ** from the matrix to normalize it before decomposing it. After performing the decomposition, we add the **μ** back to the final predictions.

$$\hat{R}_{ij} = P_i Q_j^T + \mu$$

The objective function of SVD with ALS is given by the following equation.

$$f(x) = \underset{P,Q}{\operatorname{argmin}} \sum_{i,j \,\epsilon\, R} \left(R_{ij} - \mu - P_i Q_j^T\right)^2$$

We also include a weighted lambda regularization (Tikhonov regularization) to avoid over fitting to our training set.

$$f(x) = \underset{P,Q}{\operatorname{argmin}} \sum_{i,j \,\epsilon\, R} \left(R_{ij} - \mu - P_i Q_j^T\right)^2 + \lambda\left(\|{P_i}^2\| + \|{Q_j}^2\|\right)$$

Our best results for this method were obtained at rank **k** of 100 and a regularization constant $\boldsymbol{\lambda}$ of 0.3. We could not run more than 25 iterations as that is a limitation on the version of spark we are using. Our SVD with ALS method scored an RMSE of 1.2907.

6. Machine Learning: Random Forest

In addition to the latent-factor model and other usual recommendation related techniques, we plan to make rating predictions using other machine learning methods such as random forest and

SVM. Random forest operates by constructing a multitude of decision trees at training time and outputting the mean prediction of rating (in this case) of the individual trees. Random forests reduce the variance of individual trees. This comes at the expense of a small increase in the bias and some loss of interpretability, but generally greatly boosts the performance of the final model.

In an initial prototype of Random forest, two cases were considered, (a) Known user and Known business; (b) Unknown user but Known business. This was done since most of the features available for building trees are business specific. In addition to business categories, features such as review count, latitude and longitude were considered. Hence, the maximum features considered were ~550-560 (p). For building each tree, a random subset (m) of these predictors was considered. Going by the best practice, m = $\sqrt{p}$ was used. The maximum depth of trees was restricted to 10. This basic random forest regressor resulted in an RMSE of 1.295 which is slightly better than 1.399 achieved through baseline model.

We will work further on improving the performance of random forests. Currently, train and test datasets were considered separately. Combining the users and businesses that are common in train and test will improve the predictions. Random Forest Regressor from the 'mllib' library in Spark is being used to perform the analyses.

7. Next in plan

- Attempt different machine learning techniques on the subsets of data
- Analyze other features such as Check-ins, Coordinates Cluster, etc. that may impact ratings
- Build Models post feature engineering
- To infer gender information from user names and train gender specific models
- Build Models on any other segment of data if found to be behaviorally different

	Known User	Unknown User
Known Business	Neighbor Clustering Collaborative Filtering ALS Machine Learning (RF, SVM) Linear Regression	User Means User - oriented Neighborhood Machine Learning (RF, SVM) Linear Regression
Unknown Business	Business Means Clustering by business Category Means Machine Learning (RF, SVM)	Global Average Predicted User and Business Means Category Means

8. Responsibilities

The current and future distribution of the responsibilities is as given below.
Sakshi: Data Exploration & Analysis, Optimized cold start segment of data, Feature Engineering
Piyush: Random Forest and other Machine Learning techniques, Feature Engineering
Chhavi: Baseline model, ALS model, Implement Collaborative Filtering from Scratch in Spark