Yelp User Dataset Analysis

Sanity Check SQL Queries:

Before performing any analysis, the data must be tested to ensure it satisfies basic sanity tests. Three basic sanity checks were performed using SQL queries on the reduced Yelp dataset:

1) Sanity Check 1 - verify that years elite is consistent with other time values:

```
select user_id
from elite_years join user on elite_years.user_id = user.id
where year(user.yelping_since) > elite_years.year or elite_years.year <
2004 or elite_years.year > 2017;
```

Result: Empty set (2.67 sec)

2) Sanity Check 2 - verify that the number of reviews written by a user in the User table is not smaller than the number of reviews by that user in the Review table:

```
select u.id, u.review_count, count(r.id) as review_num
from user as u join review as r on u.id = r.id
group by u.id, u.review_count
having u.review_count < review_num;
```

Result: 8 rows in set (4.00 sec)

3) Sanity Check 3 - verify that the number of reviews received by a business in the Business table is not smaller than the number of reviews of that business in the Review table :

```
select b.id, b.review_count, count(r.business_id) as review_num
from business as b join review as r on b.id = r.business_id
group by b.id, b.review_count
having b.review_count < review_num;
```

Result: Empty set (2.48 sec)

SQL Deletion Queries:

Rather than attempting to adjust the data, the rows returned from the sanity check queries was removed before performing data analysis. The SQL query used to remove the offending rows from the user, business and review tables is shown below:

```
drop table if exists usersToRemove;
create temporary table usersToRemove( id varchar(22) );

insert into usersToRemove
select distinct(u.id)
from user as u join review as r on u.id = r.id
group by u.id, u.review_count
having u.review_count < count(r.id);

insert into usersToRemove
select distinct(u.id)
from elite_years e join user u on e.user_id = u.id
where year(u.yelping_since) > e.year or e.year < 2004 or e.year > 2017;

drop table if exists businessesToRemove;
create temporary table businessesToRemove( id varchar(22) );

insert into businessesToRemove
select distinct(b.id)
from business as b join review as r on b.id = r.business_id
group by b.id, b.review_count
having b.review_count < count(r.business_id);

SET SQL_SAFE_UPDATES = 0;
SET FOREIGN_KEY_CHECKS=0;

delete u, r
from user as u join review as r on u.id = r.id
where u.id in (select * from usersToRemove);

delete b, r
from business as b join review as r on b.id = r.business_id
where b.id in (select * from businessesToRemove);

SET FOREIGN_KEY_CHECKS=1;
SET SQL_SAFE_UPDATES = 1;
```

Representative Sample

The following two queries are used to select the non-representative samples of users and businesses to extract the skewed portion of the sample.

1) Representative Sample of Users:

```
SELECT u.id, u.average_stars AS User_AvgStars, r_avgStars as
      Review_AvgStars, ABS(u.average_stars - r_avgStars) as RatingDiff
FROM user u INNER JOIN
     (SELECT user_id, avg(stars) AS r_avgStars
      FROM review
      GROUP BY user_id) AS r_avgTable
ON u.id=r_avgTable.user_id
WHERE ABS(u.average_stars - r_avgStars) > 0.5
ORDER BY RatingDiff DESC;
```

Size of non-representative users in reduced yelp database: 229244

2) Representative Sample of Businesses:

```
SELECT b.id, b.stars AS 'Business Avg Stars', r_avgStars as 'Review Avg
      Stars', ABS(b.stars - r_avgStars) as RatingDifference
FROM business b INNER JOIN
     (SELECT business_id, avg(stars) AS r_avgStars
      FROM review
      GROUP BY business_id) AS r_avgTable
ON b.id=r_avgTable.business_id
WHERE ABS(b.stars - r_avgStars) > 0.5
ORDER BY RatingDifference DESC;
```

Size of non-representative businesses in reduced yelp database: 38122

Analysis

Technique 1: Random Forest Classification

Technique Overview

Random forest is one of the most popular classification algorithms as it can be used for both classification and regression type of problems. This algorithm was chosen due to the unique nature of the problem which is predicting the ratings for any arbitrary user for an arbitrary business. The predicted rating falls into a target class i.e 1, 2, 3, 4 or 5 stars. Since classification algorithms are designed to predict the target class by analyzing the training dataset, a classification algorithm was chosen.

Furthermore, decision tree algorithms usually have a high probability of overfitting. However, the use of a random forest classifier with a moderate amount of trees in the forest won't overfit the model. Another benefit of the random forest classifier is that it allows the modelling for categorical values which are abundantly present in the yelp database. Finally, the prediction of the ratings is done based on various parameters/features supplied in the training dataset. Thus, we need an algorithm that is able to identify the most important features from the training set and random forest algorithm is highly successful at performing this task.

To summarize our choice of algorithm, the random forest classifier was used to predict which rating any arbitrary Yelp user would give to any arbitrary business. Random forests are a classification method that operate by constructing a number of decision trees during training and outputting the class that is the mode of the classes of the individual trees. The training algorithm for random forests applies a bagging technique known as "feature bagging", where the algorithm will select a random subset of the features at each candidate split in the learning process. Bagging increases the size of the training set, which results in a decrease in the variance of the prediction.

Training Dataset Selection

After removing the results obtained through sanity checks, the following SQL query was used to extract the training and testing dataset:

```
SELECT DISTINCT
u.review_count user_review_count,
u.useful user_useful,
u.funny user_funny,
u.cool user_cool,
u.fans user_fans,
u.average_stars user_average_stars,
b.stars business_stars,
b.review_count business_review_count,
length(r.text),
r.useful review_useful,
r.funny review_funny,
```

```
r.cool review_cool,
r.stars review_stars
FROM review r LEFT JOIN business b ON r.business_id = b.id
LEFT JOIN user u ON r.user_id = u.id
WHERE b.review_count > 2 AND u.review_count > 0;
```

Justification of Dataset Chosen

The random forest algorithm automatically creates decision trees where each decision tree randomly selects “k” features out of total “m” features. By randomly selecting k out of m features, random forest algorithm is able to identify the most important features if supplied a big enough dataset. Since the yelp dataset was large enough, we chose to include as many discrete and continuous columns as possible from review, user and business tables. The columns chosen were:

- User Table: review_count, useful, funny, cool, fans, average_stars
- Business Table: stars, review_count
- Review Table: Length of text, useful, funny, cool
- Target Class: stars

By providing the above 13 features to predict the target class, the random forest algorithm will be able to create enough random trees in the decision tree to identify the most important features.

In addition, users who had a review_count of 0 were excluded from the dataset. We decided to also exclude businesses that had less than 3 reviews in total. The rationale behind this was that the main goal of this model is to predict the ratings for businesses that are actively reviewed on yelp. Thus, the inclusion of businesses with 2 or less reviews in total would only result in a less accurate model while not providing any additional benefits due to the lack of activity of such businesses on yelp.

Results and Validation

Before discussing the accuracy of results obtained from the random forest algorithm, it is important to highlight the training of the classifier. This analysis evaluates the importance of each feature selecting in predicting the target class for the rating of the arbitrary business by an arbitrary user. Figure 1 shows the relative importance of each feature provided in the dataset to the algorithm.

It can be seen that the user’s average stars has the greatest impact on predicting the review a user will give to a business. This logically makes sense as most users tend to be consistent with their previous reviews. A surprising finding was that the length of the review was the second most important feature in the random forest when predicting the target class.

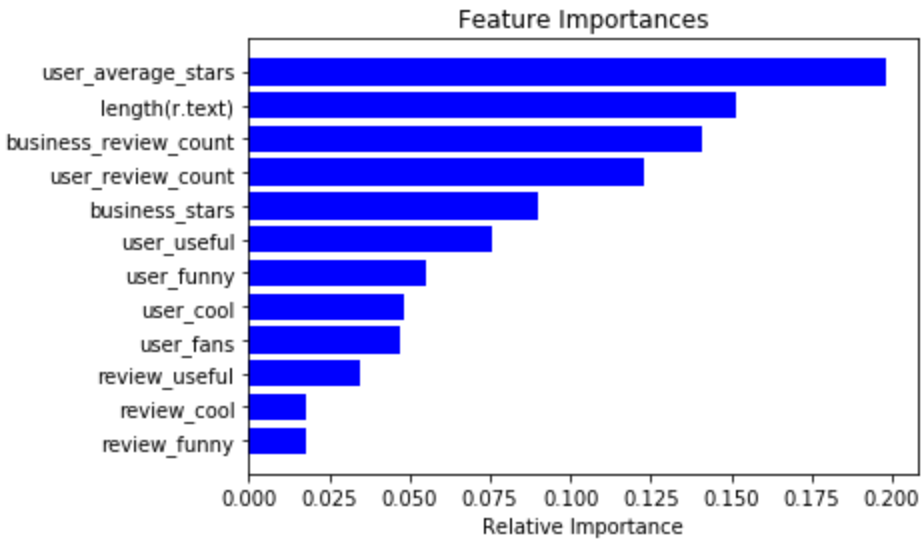

Figure 1 - Features importances in predicting rating

The dataset was randomly split in half into a training set and a testing set, each with 482012 entries. The classifier was then trained which took the specified features and learned how strongly each feature related to the target class (i.e. rating). After training the classifier, it was applied to the test dataset to attempt to predict the rating the user would give to the business. The results of this test were compared against the actual data, allowing for the accuracy of the classifier to be determined. Figure 2 shows the parameters chosen for the random forest algorithm and the obtained accuracy for the model when run on the test dataset.

```
Trained model ::  RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
            max_depth=None, max_features='auto', max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, n_estimators=10, n_jobs=1,
            oob_score=False, random_state=None, verbose=0,
            warm_start=False)
Test Accuracy  ::  0.514505862925
Train Accuracy ::  0.987512759018
```

Figure 2 - Random forest algorithm parameters and test accuracy

Figure 3 - Confusion matrix of random forest classifier

Figure 3 shows a heat map of the confusion matrix for the test data. The columns are the predicted rating, the rows are the actual rating, and therefore, the diagonal is the number of accurate predictions for each class. The lighter coloured cells are along the diagonal, which indicates that the classifier predictions have good accuracy. The accuracy of the method is calculated to be 51.45%.

As the confusion matrix shows, the model is least accurate when predicting ratings that are 2 or 3 stars and has the highest accuracy when predicting 1 or 5 stars.

Technique 2: Logistic Regression

Technique Overview

Another interesting piece of information that could be obtained from the Yelp dataset is whether there are certain attributes of users that make them more likely to be an elite user. Thus, we decided to look at factors that contribute to a certain user achieving Yelp Elite status as this information might be useful in predicting the future creation of elite users. This is important as Yelp Elite are responsible for creating most of the content (reviews) that is read by Yelp users.

The technique that best suits this analysis is logistic regression for a few main reasons. The main reason is that Logistic Regression is best suited to predict the probability of a categorical dependent variable, and the dependent variable in the analysis is a categorical variable where 1 means is_Elite and 0 means is_Not_Elite. Specifically, binary logistic regression is being used for the analysis due to the binary nature of the dependent variable where the factor level 1 of the dependent variable (is_Elite) represents the desired outcome

For Logistic Regression analysis, feature selection was done using a Recursive Feature Elimination (RFE) algorithm. First, the RFE is trained on the initial set of features and the importance of each feature is obtained. Then, the least important features are pruned from current set of features. The goal of this was to perform the analysis only on the features which would have the greatest effect of the value of the dependent variable. The RFE resulted in the removal of the "cool" and "useful" features. The logistic model was then fit to our training dataset. After the model had been trained, its validity was tested on the testing dataset.

Training Dataset Selection

After removing the results obtained through sanity checks, the following SQL query was used to extract the training and testing dataset:

```
SELECT DISTINCT
review_count as review_count,
YEAR(yelping_since) as yelping_since_year,
average_stars,
useful as useful,
funny as funny,
cool as cool,
fans as fans,
IF(year IS NULL, 0, 1) as is_elite,
FROM user LEFT JOIN elite_years ON user.id = elite_years.user_id
WHERE review_count >= 5 AND review_count < 1000;
```

Justification of Dataset Chosen

Since the objective of this analysis was to predict elite user status, it was decided to limit the data used for prediction to only the user table. Since the user table has about 0.42 million entries, we chose to include many columns which were:

- review_count
- yelping_since (only Year)
- average_stars
- useful
- funny
- cool
- fans

In addition, we placed a lower and upper limit of 5 and 1000 respectively on the total reviews of the users that were analyzed. The rationale behind having a lower limit was to eliminate users who aren't active on Yelp and would not provide significant data for the training set. The upper limit was put in place to focus the analysis on users who are in the "transition" phase of being elite as most users with extremely high reviews are already elite.

Results and Validation

The scikit-learn "train-test-split" function was used to randomly split the data into 2 equal sets - one for the training data, and one for the testing data. Each data set was composed of 422546 entries. The training data was used to train the logistic regression model and the accuracy of the model was measured by using the testing data.

Logit Regression Results

Dep. Variable:	is_elite	No. Observations:	422546
Model:	Logit	Df Residuals:	422539
Method:	MLE	Df Model:	6
Date:	Mon, 04 Dec 2017	Pseudo R-squ.:	0.4351
Time:	20:45:06	Log-Likelihood:	-92337.
converged:	True	LL-Null:	-1.6347e+05
		LLR p-value:	0.000

	coef	std err	z	P>|z|	[0.025	0.975]
review_count	0.0190	0.000	168.249	0.000	0.019	0.019
yelping_since_year	-0.0029	2.54e-05	-115.989	0.000	-0.003	-0.003
average_stars	0.5853	0.013	46.267	0.000	0.561	0.610
useful	-8.179e-05	3.17e-05	-2.580	0.010	-0.000	-1.97e-05
funny	-0.0002	3.12e-05	-6.899	0.000	-0.000	-0.000
cool	8.411e-06	4.13e-05	0.203	0.839	-7.26e-05	8.94e-05
fans	0.0872	0.001	69.593	0.000	0.085	0.090

Accuracy of logistic regression classifier on test set: 0.92

Figure 4: Logistic Regression Results

Figure 4 shows the different parameters for the logistic regression model created and the overall accuracy of the model is 92%. The accuracy is particularly high for the model which can be explained by the dataset chosen. Since all users with reviews < 5 were ignored, the model was able to train itself better yielding a higher accuracy. Finally, the p-values for all variables except for “cool” are very small and are therefore significant to the model. Another interesting finding is that the “average_stars” rating has a strong relationship with a user being elite or not highlighting how Yelp puts emphasis on the ratings a user gives.

As the confusion matrix in figure 5 shows, the model is most accurate when predicting whether a user is not an elite user.

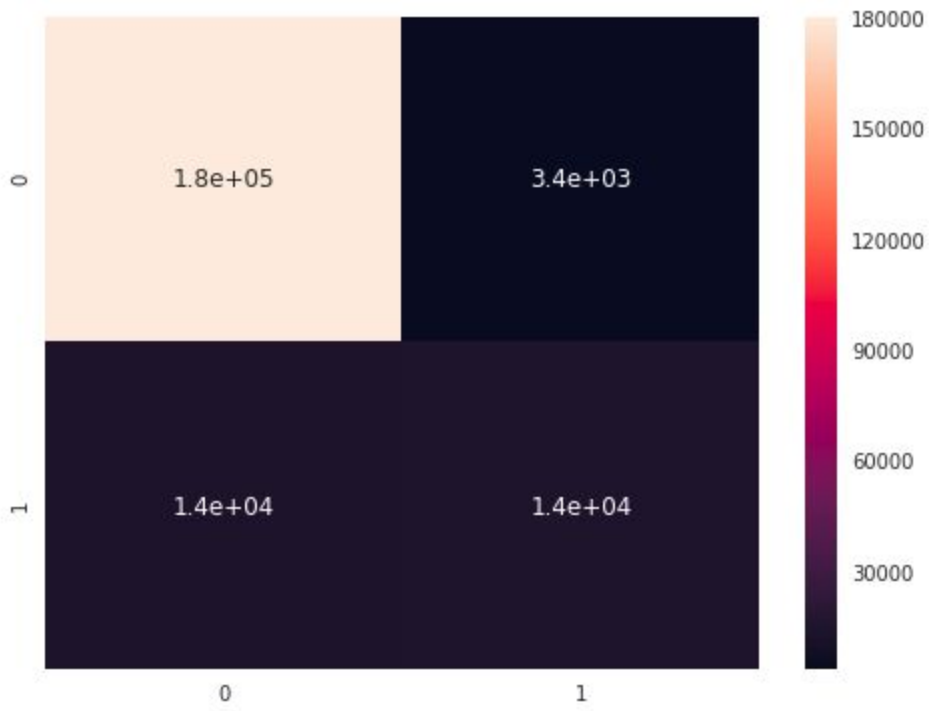

Figure 5 - Confusion matrix of Logistic Regression technique