Homework 2
Project Report
Yelp Data Ananlysis

Fnu Karan
NetID: kx361

March 12, 2017

The dataset is freely available at https://www.yelp.com/dataset_challenge/dataset

We just need to fill out a small form and we can access the data. We get a .tar file and we can extract the json files from it. This homework did not require us to use the images dataset. But it made us use the other five datasets, which consisted of reviews, users, business, tips.

Here I did some bery minimal statistics to get some sense out of the data. Mainly it was implemented in PIG. And the observations were plotted using Tableau.

Before performing any of the operations, I made sure that the HPC is working correctly, and flawlessly. Also, the most challenging task was to load the json dataset into Pig variables. This was resolved using the libraries by Twitter, called elephant-bird.

The report contains each question described in sequence. Each section contains the respective questions and the code for each of them and the plots for them.

Following are the operations that were performed:

1 Summarize the number of reviews by US city, by business category

Here is the Pig Script for this problem,

```
SET elephantbird.jsonloader.nestedLoad 'true';

--LOAD FROM THE JSON
loadJson = LOAD '/user/kx361/business.json' USING com.twitter.elephantbird.pig
    .load.JsonLoader('-nestedLoad') AS (json:map []);

--GENERATE THE REQUIRED DATA
businessData = FOREACH loadJson GENERATE (int)json#'review_count' as
    review_count, json#'city' as city, json#'categories' as categories;

--FILTER THE DATA BY REQUIRED CITIES
us_business = FILTER businessData BY (
(city matches '.*Pittsburgh.*') OR (city matches '.*Charlotte.*') OR (city
    matches '.*Urbana-Champaign.*') OR (city matches '.*Phoenix.*') OR (city
    matches '.*Las Vegas.*') OR (city matches '.*Madison.*') OR (city matches
    '.*Cleveland.*'));
```

```
12
13 --FLATTEN THE DATA OUT
14 flattenedBusinessData = FOREACH us_business GENERATE review_count, city,
       FLATTEN(categories);
15
16 --GROUP THE DATA BY CITY AND CATEGORIES
17 groupedBusinessData = GROUP flattenedBusinessData BY (city,categories);
18
19 --PERFORM THE REQUIRED SUM OPERATION ON REVIEW_COUNT
20 finalData = FOREACH groupedBusinessData GENERATE group.city as city , group.
       categories as category, SUM(flattenedBusinessData.review_count);
21
22 --STORE THE DATA IN THE DIRECTORY
23 store finalData into './p1' using PigStorage(',') ;
```

In the script, elephant-bird was used to load the data in pig variables. The required fields are then extracted from the data with aliases to work on them. Since the data is grouped together, we also flattened out the data to separate out all record to perform the operations that are required.

Once the script is run, and the required output is obtained, then we plot the visualizations in Figure 1,2 and 3.

As we can see from the figures, yelp is mainly being used for restaurant reviews, since it had the maximum number of reviews.

2 Rank all cities by number of stars descending, for each category

Here is the Pig Script for this problem,

```
1  SET elephantbird.jsonloader.nestedLoad 'true';
2
3  --LOAD BUSINESS DATA
4  yelp_business_data = LOAD '/user/kx361/business.json' USING com.twitter.
       elephantbird.pig.load.JsonLoader('-nestedLoad') as (json:map[]);
5
6  --FLATTEN OUT THE CATEGORIES
7  yelp_business_data_category = FOREACH yelp_business_data GENERATE (float)json#
       'stars' AS stars, json#'city' as city, FLATTEN(json#'categories') AS
       categories;
8
9  --GROUP BY CATEGORIES AND CITY
10 yelp_business_group= GROUP yelp_business_data_category BY (categories,city);
11
12 --CALCULATE THE AVERAGE NUMBER OF STARS FOR EACH GROUP
13 yelp_business_group_data= FOREACH yelp_business_group GENERATE group.
       categories as category,group.city AS city, AVG(yelp_business_data_category
       .stars) AS stars;
14
15 --GROUP AND ORDER THIS DATASET BY CATEGORIES
16 yelp_data_order= ORDER yelp_business_group_data BY category ASC;
17 grouped_categories = group yelp_data_order by category;
18
19 --NOW THE EACH CATEGORY CONSISTS CITY AND AVERAGE STARS. SO WE SORT THAT FOR
       EACH SINGLE CATEGORY.
```

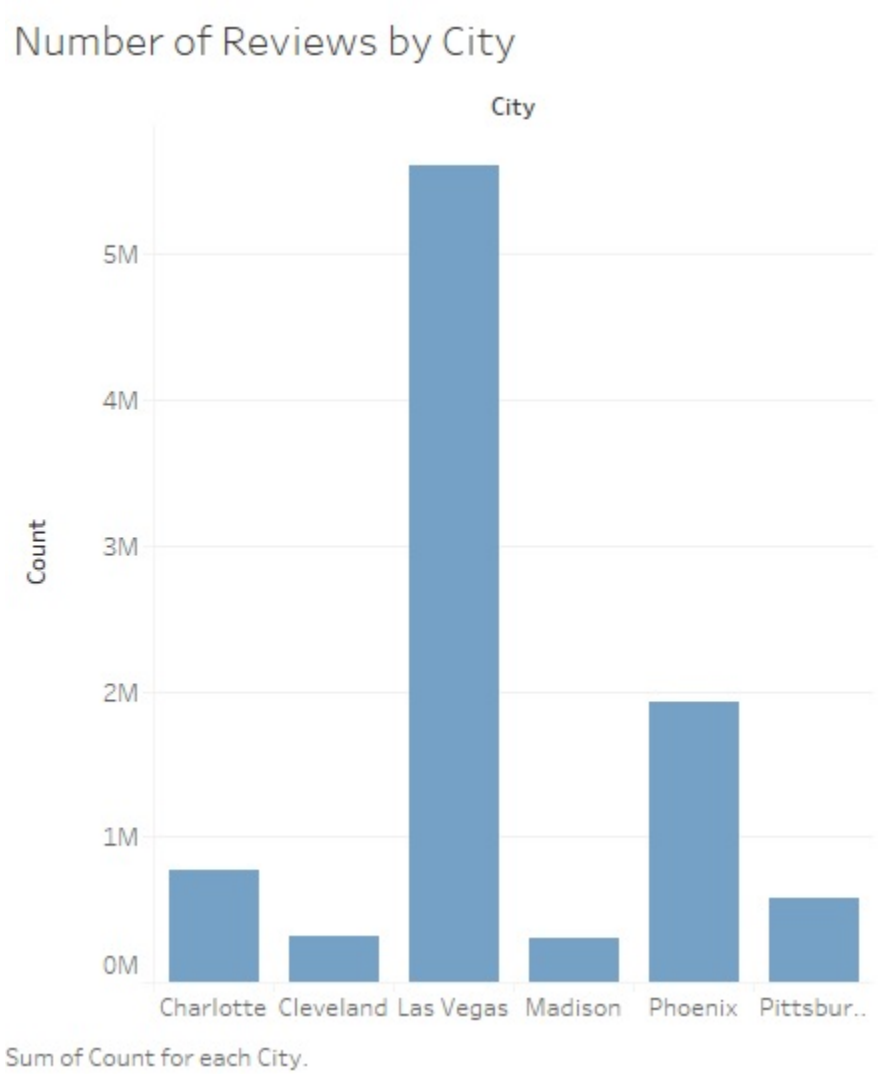

Figure 1: Number of reviews by city.

```
20 sorted_stars_data = FOREACH grouped_categories{
21         sorted= ORDER yelp_data_order by stars desc;
22         GENERATE FLATTEN(sorted);
23 };
24
25 --AND FINALLY STORE THE RESULT.
26 STORE sorted_stars_data INTO './p2' using PigStorage(',') ;
```

The code is self explanatory with all the comments indicating the steps that were taken to work on the problem. Here the catch was to sort the data by two fields. This is indiacted in the line number 20-23. What I did, is that I generated a sorted list of stars for each category and then I flattened it so that each of the records is indicated as individual. I did that for each of the category and hence I was able to sort the cities with number of stars for each category.

Once the script is run, and the required output is obtained, then we plot the visualizations in Figure 4,5, and 6

```
> p1 = read.csv('p1.csv')
> summary(p1)
         City                     Category    Number.Of.Reviews
 Charlotte : 749                        :   6   Min.    :     3
 Cleveland : 583   Accessories          :   6   1st Qu.:    24
 Las Vegas :1002   Active Life          :   6   Median :   123
 Madison   : 628   Acupuncture          :   6   Mean   :  2060
 Phoenix   : 936   Adult                :   6   3rd Qu.:   623
 Pittsburgh: 713   Adult Education      :   6   Max.   :761661
                   (Other)              :4575
```

Figure 2: Summary of the dataset in R.

3 Average number of stars for businesses within 10 miles of the University of Wisconsin, Madison, by business category

Here is the Pig Script for this problem,

```
SET elephantbird.jsonloader.nestedLoad 'true';

--LOAD BUSINESS DATA
buinessData = LOAD '/user/kx361/business.json' USING com.twitter.elephantbird.pig.load.JsonLoader('-nestedLoad') as (json:map[]);

--GET THE REQUIRED DATA
requiredData =  FOREACH buinessData GENERATE FLATTEN(json#'categories') AS type_of_business, (float)json#'stars' AS stars, (float)json#'latitude' as latitude, (float)json#'longitude' as longitude;

--FILTER THE DATA BY LINGITUDE AND LATITUDE
wisconsinBusiness =     FILTER requiredData BY (latitude>42.9083) AND (latitude<43.2417) AND (longitude>-89.5839) AND (longitude<-89.2506);

-- GROUP THE DATA BY BUSINESS TYPE
groupedByCategory = GROUP wisconsinBusiness BY type_of_business;

-- FOR EACH GROUP, GENERATE THE AVG NUMBER OF STARS
finalData = FOREACH groupedByCategory GENERATE group as categories, AVG(wisconsinBusiness.stars) AS stars;

--FINALLY SORT THE DATA BY CATEGORIES/TYPE
orderedData = ORDER finalData BY categories;

--STORE IT AS CSV
STORE orderedData INTO './p3' using PigStorage(',');
```

The code is self explanatory with all the comments indicating the steps that were taken to work on the problem.

Once the script is run, and the required output is obtained, then we plot the visualizations in Figure 7 and 8.

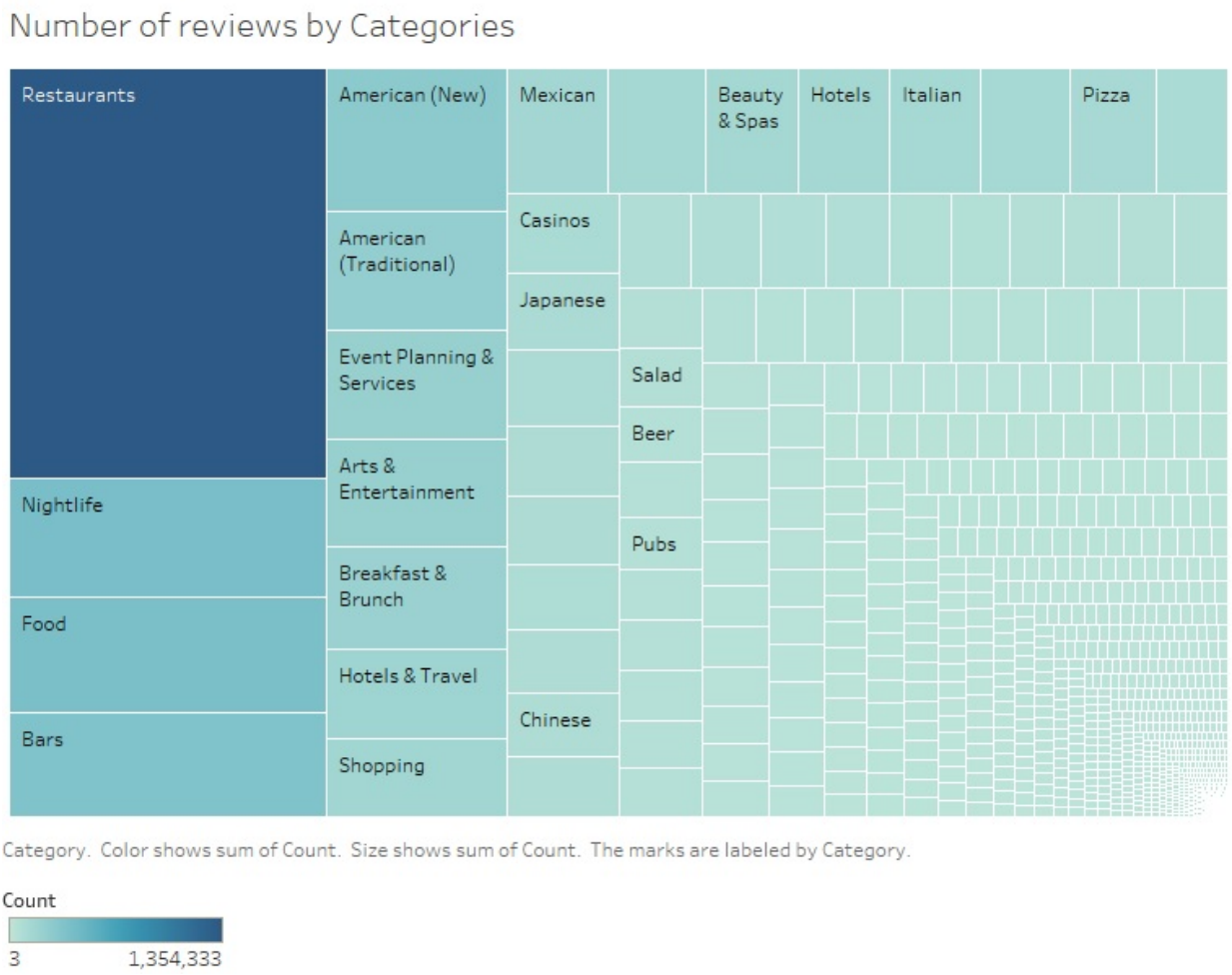

Figure 3: Number of reviews by categories.

4 Average number of stars for the top 10 reviewers

Here is the Pig Script for this problem,

```
1 SET elephantbird.jsonloader.nestedLoad 'true';
2
3 --Fetch business data
4 businessData = LOAD './business.json' USING com.twitter.elephantbird.pig.load.
    JsonLoader('-nestedLoad=true') AS (yelp_business: map[]);
5 business = FOREACH businessData GENERATE yelp_business#'business_id' as
    business_id, yelp_business#'categories' as categories;
6
7 --Fetch review data
8 reviewData = LOAD './yelp_academic_dataset_review.json' USING com.twitter.
    elephantbird.pig.load.JsonLoader('-nestedLoad=true') AS (yelp_review: map
    []);
9 review = FOREACH reviewData GENERATE yelp_review#'user_id' as user_id1,
    yelp_review#'business_id' as business_id, (int)yelp_review#'stars' as
```

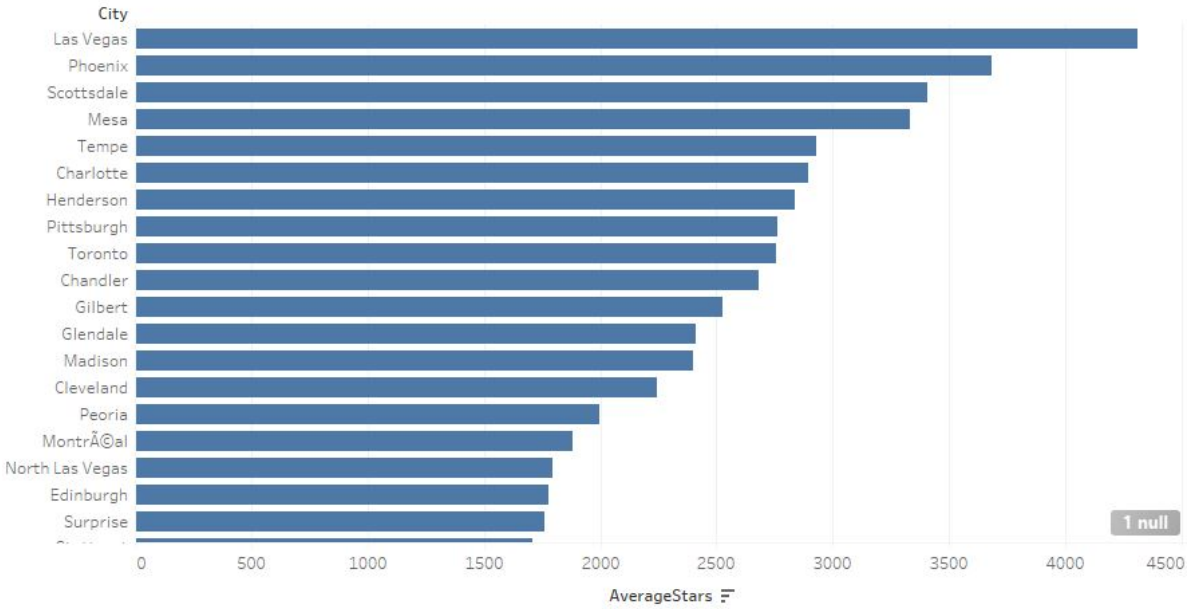

Figure 4: Total of Average stars for each category by city.

```
    stars;

--Fetch user data
userData = LOAD './yelp_academic_dataset_user.json' USING com.twitter.
    elephantbird.pig.load.JsonLoader('-nestedLoad=true') AS (yelp_user: map[])
    ;
user = FOREACH userData GENERATE yelp_user#'user_id' as user_id, (int)
    yelp_user#'review_count' as review_count;

--Sort the order in descending order
sortedUsers = ORDER user BY review_count DESC;

--Get the top 10
top10Users = LIMIT sortedUsers 10;

--Join the users with reviews on by user id
userReviewJoined = JOIN top10Users BY user_id, review BY user_id1;

--Jooin the joined part with businesses
userReviewBusinessJoined = JOIN business BY business_id, userReviewJoined BY
    business_id;

--Flatten the data
flattenedData = FOREACH userReviewBusinessJoined GENERATE FLATTEN(categories),
     stars, user_id;

--group the data by users and also categories
grouped = GROUP flattenedData BY (user_id, categories);

--calculate the averaage and sort it
```

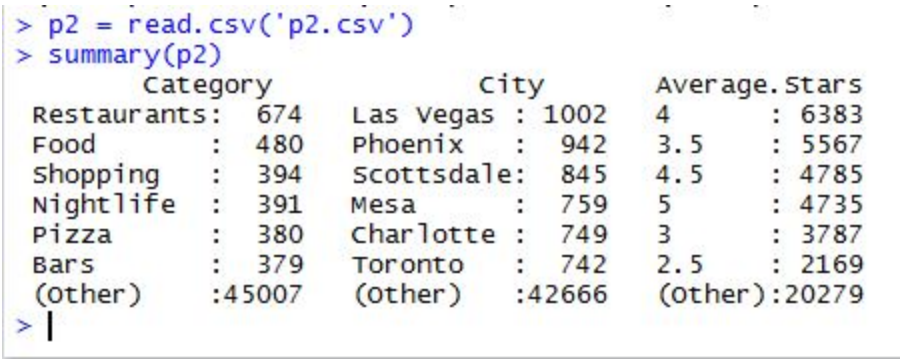

Figure 5: Summary of the dataset in R.

```
grouped_result = FOREACH grouped GENERATE group, AVG(flattenedData.stars) as
    avg_stars;
ordered_result = ORDER grouped_result BY user_id;

--store the result into the directory
STORE grouped_result INTO './p4' using PigStorage(',');
```

The code is self explanatory with all the comments indicating the steps that were taken to work on the problem.

Once the script is run, and the required output is obtained, then we plot the visualizations in Figure 9, 10 and 11.

5 Summarize star rating for reviews in January through May For the top 10 and bottom 10 food business near UWM (in terms of stars)

Here is the Pig Script for this problem,

```
SET elephantbird.jsonloader.nestedLoad 'true';

--
business_data = LOAD './business.json' USING com.twitter.elephantbird.pig.load
    .JsonLoader('-nestedLoad') AS (json:map []);

required_business_data = FOREACH business_data GENERATE json#'business_id' as
    business_id, json#'city' as city, json#'name' as name,(double) json#'
    latitude' as latitude, (float) json#'longitude' as longitude, (double)
    json#'stars' as stars, FLATTEN(json#'categories') as categories;

filtered_data = FILTER required_business_data BY (latitude <= 43.2467) AND (
    latitude >= 42.90833) AND (longitude >= -89.58389) AND (longitude <=
    -89.25056);

filtered_food_business = FILTER filtered_data by categories matches '.*Food.*'
    ;

```

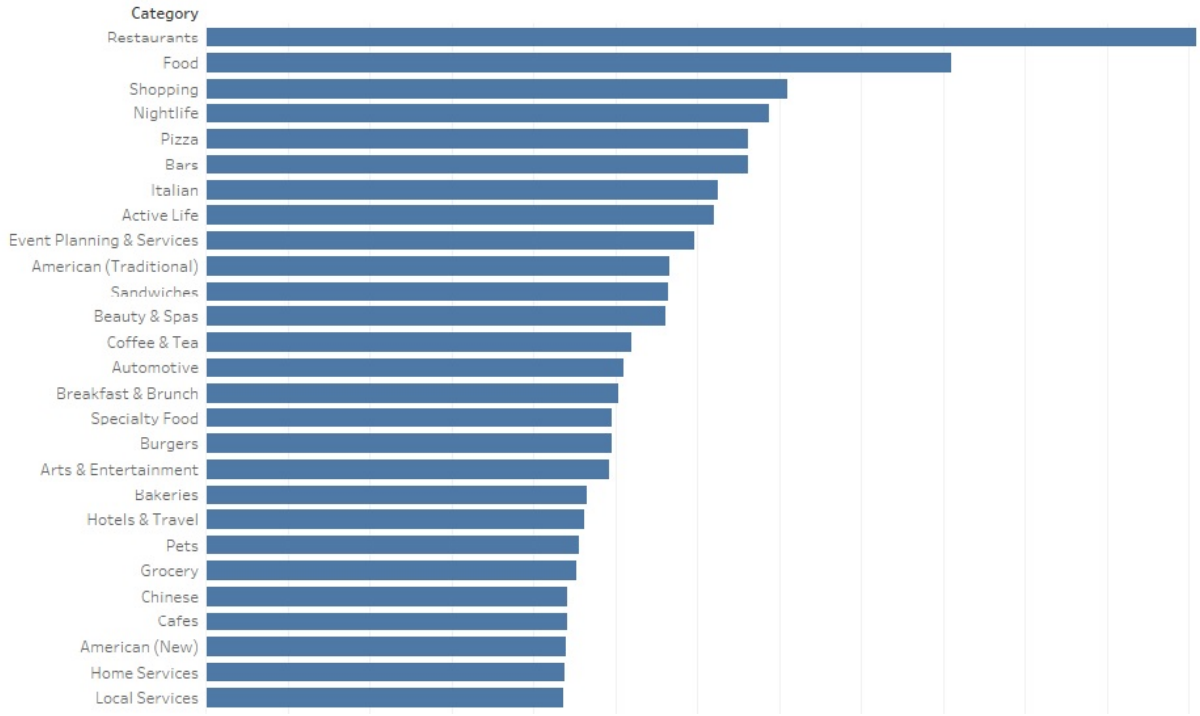

Figure 6: Total of average stars for each city by category.

```
business_id_stars = FOREACH filtered_food_business GENERATE business_id, stars
    ;

ordered_data = ORDER business_id_stars by stars DESC;

top10 = LIMIT ordered_data 10;

Ordered_By_Stars_Asc = ORDER business_id_stars by stars;

bottom10 = LIMIT Ordered_By_Stars_Asc 10;

top10bottom10 = UNION top10, bottom10;

review_data = LOAD './yelp_academic_dataset_review.json' USING com.twitter.
    elephantbird.pig.load.JsonLoader('-nestedLoad') AS (json:map []);

required_review_data = FOREACH review_data GENERATE json#'review_id' as
    review_id, json#'business_id' as business_id,json#'date' as date, json#'
    stars' as stars;

joined_reviews = JOIN required_review_data by business_id, top10bottom10 by
    business_id;

final_required_Data = FOREACH joined_reviews GENERATE top10bottom10::
    business_id as bid, (double) required_review_data::stars as star,
    required_review_data::review_id, SUBSTRING(required_review_data::date,5,7)
     as month;
```

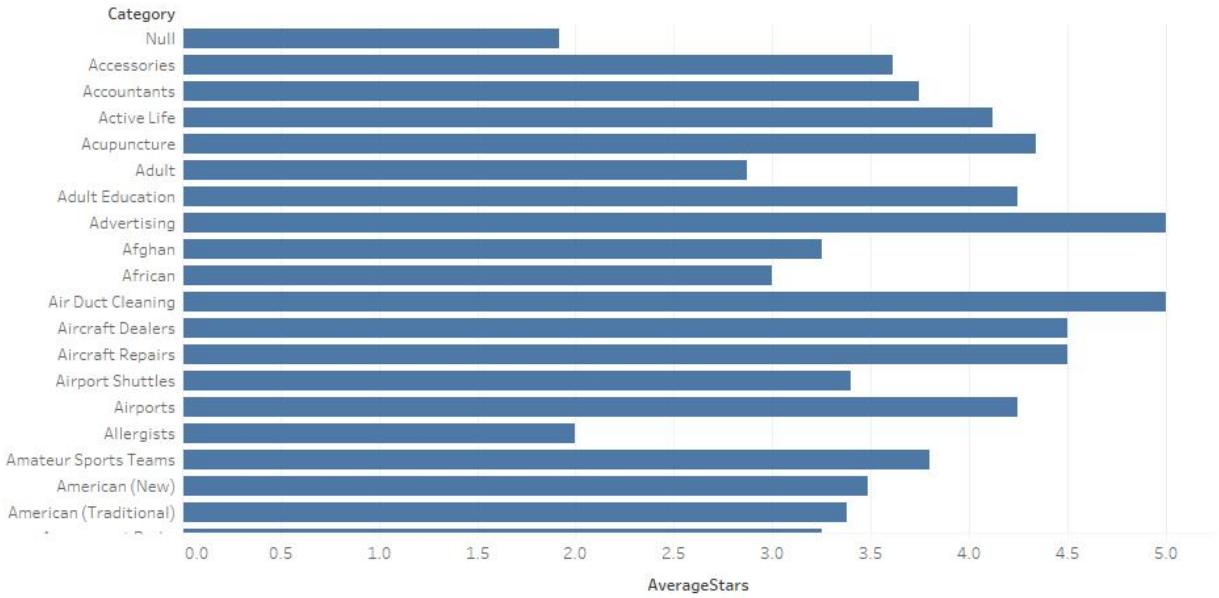

Figure 7: Average number of stars for the categories of businesses near University of Wisconsin.

```
> p3 = read.csv('p3.csv')
> summary(p3)
        Category    AverageStars
             :  1   Min.   :1.000
 Accessories:  1   1st Qu.:3.500
 Accountants:  1   Median :3.833
 Active Life:  1   Mean   :3.817
 Acupuncture:  1   3rd Qu.:4.250
 Adult      :  1   Max.   :5.000
 (Other)    :656
> |
```

Figure 8: Summary of the dataset in R.

```
31
32 filtered_data_by_month = FILTER final_required_Data BY (month matches '
       01|02|03|04|05');
33
34 grouped_data_by_business = GROUP filtered_data_by_month by bid;
35
36 avg_rating = FOREACH grouped_data_by_business GENERATE group, AVG(
       filtered_data_by_month.star) as avg_stars;
37
38 STORE avg_rating into './p5' using PigStorage (',');
```

The code is self explanatory with all the comments indicating the steps that were taken to work on the problem.

Once the script is run, and the required output is obtained, then we plot the visualizations in Figure 12 and 13.

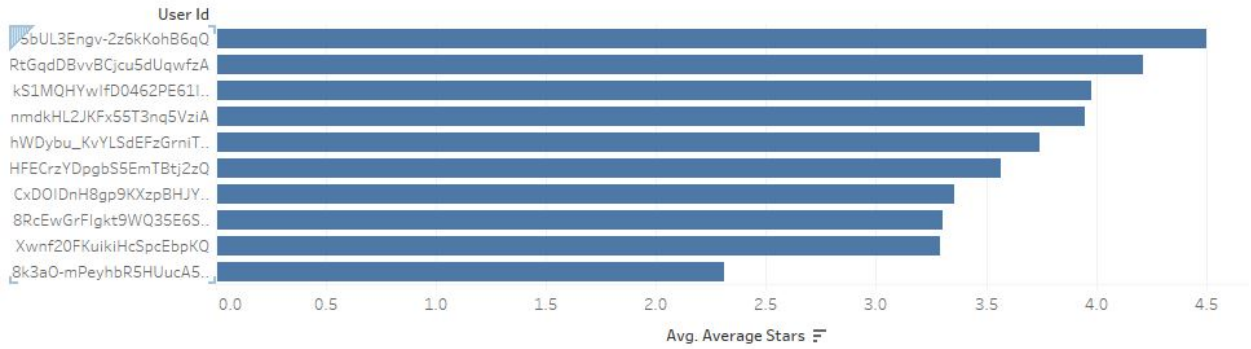

Figure 9: Average number of stars for every user in Top 10.

This might not look much, but it requried a good understanding of PIG, and R. Understanding the data and plotting various plots could not have been done without the understanding of the data. Since, the data was huge, so these manipulations could not have been done any local database like SQL, Oracle or for that matter MongoDB.

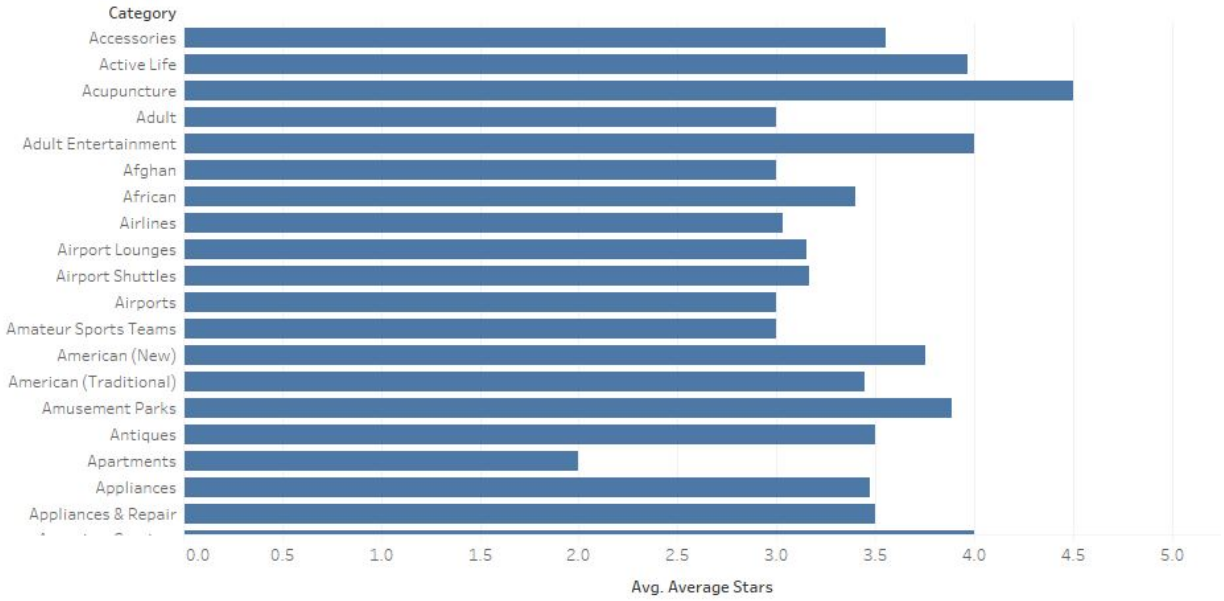

Figure 10: Average number of stars for each category by these top 10 reviewers.

```
> summary(p4)
                 UserId                                Category     AverageStars
 CxDOIDnH8gp9KXzpBHJYXw:385   Arts & Entertainment      :  10   Min.   :1.000
 hWDybu_KvYLSdEFzGrniTw:347   American (New)            :   9   1st Qu.:3.009
 Xwnf20FKuikiHcSpcEbpKQ:154   Bakeries                  :   9   Median :3.500
 nmdkHL2JKFx55T3nq5VziA:114   Bars                      :   9   Mean   :3.585
 HFECrzYDpgbS5EmTBtj2zQ: 95   Event Planning & Services:   9   3rd Qu.:4.000
 8RcEwGrFIgkt9WQ35E6SnQ: 74   Food                      :   9   Max.   :5.000
 (Other)                :141   (Other)                   :1255
>
```

Figure 11: Summary of the dataset in R.

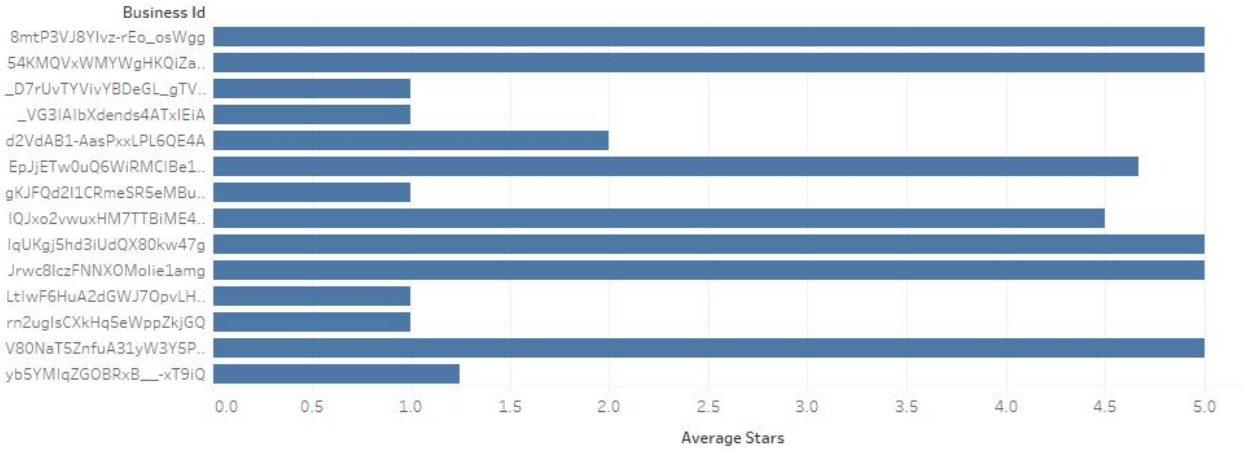

Figure 12: Average number of stars for top 10 and bottom 10 businesses.

```
> p5 = read.csv('p5.csv')
> summary(p5)
                BusinessId  AverageStars
 _D7rUvTYVivYBDeGL_gTVQ:1    Min.   :1.00
 _VG3IAIbXdends4ATxlEiA:1    1st Qu.:1.00
 54KMQVxWMYWgHKQiZa_58Q:1    Median :3.25
 8mtP3VJ8Ylvz-rEo_osWgg:1    Mean   :3.03
 d2VdAB1-AasPxxLPL6QE4A:1    3rd Qu.:5.00
 EpJjETw0uQ6WiRMCIBe10A:1    Max.   :5.00
 (Other)                :8
> 
```

Figure 13: Summary of the dataset in R.