Data Scientist Role Play: Profiling and Analyzing the Yelp Dataset Coursera Worksheet

This is a 2-part assignment. In the first part, you are asked a series of questions that will help you profile and understand the data just like a data scientist would. For this first part of the assignment, you will be assessed both on the correctness of your findings, as well as the code you used to arrive at your answer. You will be graded on how easy your code is to read, so remember to use proper formatting and comments where necessary.

In the second part of the assignment, you are asked to come up with your own inferences and analysis of the data for a particular research question you want to answer. You will be required to prepare the dataset for the analysis you choose to do. As with the first part, you will be graded, in part, on how easy your code is to read, so use proper formatting and comments to illustrate and communicate your intent as required.

For both parts of this assignment, use this "worksheet." It provides all the questions you are being asked, and your job will be to transfer your answers and SQL coding where indicated into this worksheet so that your peers can review your work. You should be able to use any Text Editor (Windows Notepad, Apple TextEdit, Notepad ++, Sublime Text, etc.) to copy and paste your answers. If you are going to use Word or some other page layout application, just be careful to make sure your answers and code are lined appropriately. In this case, you may want to save as a PDF to ensure your formatting remains intact for you reviewer.

Part 1: Yelp Dataset Profiling and Understanding

1. Profile the data by finding the total number of records for each of the tables below:

i. Attribute table = 10000
SQL code used:

```
Select count(*) as total_records
From Attribute;
```

ii. Business table = 10000
SQL code used:

```
Select count(*) as total_records
From Business;
```

iii. Category table = 10000
SQL code used:

```
Select count(*) as total_records
From Category;
```

iv. Checkin table = 10000
SQL code used:

```
Select count(*) as total_records
From Checkin;
```

v. elite_years table = 10000
SQL code used:

```
Select count(*) as total_records
From elite_years;
```

vi. friend table = 10000
SQL code used:

```
Select count(*) as total_records
From Friend;
```

vii. hours table = 10000
SQL code used:

```
Select count(*) as total_records
From hours;
```

viii. photo table = 10000
SQL code used:

```
Select count(*) as total_records
From Photo;
```

ix. review table = 10000
SQL code used:

```
Select count(*) as total_records
From review;
```

x. tip table = 10000
SQL code used:

```
Select count(*) as total_records
From tip;
```

xi. user table = 10000
SQL code used:

```
Select count(*) as total_records
From User;
```

2. Find the total distinct records by either the foreign key or primary key for each table. If two foreign keys are listed in the table, please specify which foreign key.

i. Business = 10000
SQL code used:

```
Select count(distinct id) as total_records /*id is the Primary key*/
From Business;
```

ii. Hours = 1562
SQL code used:

```
Select count(distinct business_id) as total_records /*business_id is the Foreignk
ey*/
From hours;
```

iii. Category = 2643
SQL code used:

```
Select count(distinct business_id) as total_records /*business_id is the Foreignk
ey*/
From category;
```

iv. Attribute = 1115
SQL code used:

```
Select count(distinct business_id) as total_records /*business_id is the Foreignk
ey*/
From Attribute;
```

v. Review = 10000
SQL code used:

```
Select count(distinct id) as total_records /*id is the Primary key*/
From Review;
```

vi. Checkin = 493
SQL code used:

```
Select count(distinct business_id) as total_records /*business_id is the Foreignk
ey*/
From Checkin;
```

vii. Photo = 10000
SQL code used:

```
Select count(distinct id) as total_records /*id is the Primary key*/
From Photo;
```

viii. Tip = 537
SQL code used:

```
Select count(distinct user_id) as total_records /*business_id is the Foreignkey*/
From Tip;
```

ix. User = 10000
SQL code used:

```
Select count(distinct id) as total_records /*id is the Primary key*/
From User;
```

x. Friend = 11
SQL code used:

```
Select count(distinct user_id) as total_records /*business_id is the Foreignkey*/
From Friend;
```

xi. Elite_years = 2780
SQL code used:

```
Select count(distinct user_id) as total_records /*business_id is the Foreignkey*/
From Elite_years;
```

Note: Primary Keys are denoted in the ER-Diagram with a yellow key icon.

3. Are there any columns with null values in the Users table? Indicate "yes," or "no."

Answer: NO

SQL code used to arrive at answer:

```
select *
    from user
    /*check for each column if there are any null values*/
    where id IS NULL
        OR name IS NULL
        OR review_count IS NULL
        OR yelping_since IS NULL
        OR useful IS NULL
        OR funny IS NULL
        OR cool IS NULL
        OR fans IS NULL
        OR average_stars IS NULL
        OR compliment_hot IS NULL
        OR compliment_more IS NULL
        OR compliment_profile IS NULL
        OR compliment_cute IS NULL
        OR compliment_list IS NULL
        OR compliment_note IS NULL
        OR compliment_plain IS NULL
        OR compliment_cool IS NULL
        OR compliment_funny IS NULL
        OR compliment_writer IS NULL
        OR compliment_photos IS NULL;
```

4. For each table and column listed below, display the smallest (minimum), largest (maximum), and average (mean) value for the following fields:

i. Table: Review, Column: Stars

min:1 max:5 avg:3.7082

SQL code used:

```
select  min(stars) as Smallest,
        max(stars) as Largest,
        avg(stars) as Average
from review;
```

ii. Table: Business, Column: Stars

min:1.0 max:5.0 avg:3.6549

SQL code Used:

```
select  min(stars) as Smallest,
        max(stars) as Largest,
        avg(stars) as Average
from business;
```

iii. Table: Tip, Column: Likes

min:0 max:2 avg:0.0144

SQL code used:

```
select  min(likes) as Smallest,
        max(likes) as Largest,
        avg(likes) as Average
from tip;
```

iv. Table: Checkin, Column: Count

min:1 max:53 avg:1.9414

SQL code used:

```
select  min(count) as Smallest,
        max(count) as Largest,
        avg(count) as Average
from checkin;
```

v. Table: User, Column: Review_count

min:0 max:2000 avg:24.2995

SQL code used:

```
select  min(review_count) as Smallest,
        max(review_count) as Largest,
        avg(review_count) as Average
from user;
```

5. List the cities with the most reviews in descending order:

SQL code used to arrive at answer:

```
SELECT city,
sum(review_count) AS total_review /*total number of reviews for each city*/
FROM business
GROUP BY city
ORDER BY total_review DESC;
```

Copy and Paste the Result Below:

city	total_review
Las Vegas	82854
Phoenix	34503
Toronto	24113
Scottsdale	20614
Charlotte	12523
Henderson	10871
Tempe	10504
Pittsburgh	9798
Montréal	9448
Chandler	8112
Mesa	6875
Gilbert	6380
Cleveland	5593
Madison	5265
Glendale	4406
Mississauga	3814
Edinburgh	2792
Peoria	2624
North Las Vegas	2438
Markham	2352
Champaign	2029
Stuttgart	1849
Surprise	1520
Lakewood	1465
Goodyear	1155

(Output limit exceeded, 25 of 362 total rows shown)

6. Find the distribution of star ratings to the business in the following cities:

i. Avon

SQL code used to arrive at answer:

```
select stars,
count(stars) as count
from business
where city='Avon'
group by stars;
```

Copy and Paste the Resulting Table Below (2 columns â€" star rating and count):

stars	count
1.5	1
2.5	2
3.5	3
4.0	2
4.5	1
5.0	1

ii. Beachwood

SQL code used to arrive at answer:

```
select stars,
count(stars) as count
from business
where city='Beachwood'
group by stars;
```

Copy and Paste the Resulting Table Below (2 columns â€" star rating and count):

stars	count
2.0	1
2.5	1
3.0	2
3.5	2
4.0	1
4.5	2
5.0	5

7. Find the top 3 users based on their total number of reviews:

SQL code used to arrive at answer:

```
SELECT id,
name,
review_count
FROM user
ORDER BY review_count DESC /*only top three*/
limit 3;
```

Copy and Paste the Result Below:

```
+------------------------+--------+--------------+
| id                     | name   | review_count |
+------------------------+--------+--------------+
| -G7Zkl1wIWBBmD0KRy_sCw | Gerald |         2000 |
| -3s52C4zL_DHRK0ULG6qtg | Sara   |         1629 |
| -8lbUNlXVSoXqaRRiHiSNg | Yuri   |         1339 |
+------------------------+--------+--------------+
```

8. Does posing more reviews correlate with more fans?

Please explain your findings and interpretation of the results:

When we include number of fans in the data set, it still show no correlation between review count and number of fans. As seen from table below.

```
+-----------+------------------------+--------------+------+
| name      | id                     | review_count | fans |
+-----------+------------------------+--------------+------+
| Gerald    | -G7Zkl1wIWBBmD0KRy_sCw |         2000 |  253 |
| Sara      | -3s52C4zL_DHRK0ULG6qtg |         1629 |   50 |
| Yuri      | -8lbUNlXVSoXqaRRiHiSNg |         1339 |   76 |
| .Hon      | -K2Tcgh2EKX6e6HqqIrBIQ |         1246 |  101 |
| William   | -FZBTkAZEXoP7CYvRV2ZwQ |         1215 |  126 |
| Harald    | --2vR0DIsmQ6WfcSzKWigw |         1153 |  311 |
| eric      | -gokwePdbXjfS0iF7NsUGA |         1116 |   16 |
| Roanna    | -DFCC64NXgqrxlO8aLU5rg |         1039 |  104 |
| Mimi      | -8EnCioUmDygAbsYZmTeRQ |          968 |  497 |
| Christine | -0IiMAZI2SsQ7VmyzJjokQ |          930 |  173 |
| Ed        | -fUARDNuXAfrOn4WLSZLgA |          904 |   38 |
| Nicole    | -hKniZN2OdshWLHYuj21jQ |          864 |   43 |
```

(Output limit exceeded, 12 of 10000 total rows shown)

9. Are there more reviews with the word "love" or with the word "hate" in them?

Answer:
As per my finding, reviews with the word 'love' in text column are more in comparison to review with the word 'hate' in them.
Total reviews containing 'love' word = 1780
Total reviews containing 'hate' word = 232

SQL code used to arrive at answer:

```
/*calculating reviews containing the word love*/
select count(id) as Total_reviews_containing_word_love
from review
where text like '%love%';

/*calculating reviews containing the word hate*/
select count(id) as Total_reviews_containing_word_hate
from review
where text like '%hate%';
```

10. Find the top 10 users with the most fans:

SQL code used to arrive at answer:

```
SELECT id,
name,
fans
FROM user
ORDER BY fans DESC /*finding top 10*/
limit 10;
```

Copy and Paste the Result Below:

id	name	fans
-9I98YbNQnLdAmcYfb324Q	Amy	503
-8EnCioUmDygAbsYZmTeRQ	Mimi	497
--2vR0DIsmQ6WfcSzKWigw	Harald	311
-G7Zkl1wIWBBmD0KRy_sCw	Gerald	253
-0IiMAZI2SsQ7VmyzJjokQ	Christine	173
-g3XIcCb2b-BD0QBCcq2Sw	Lisa	159
-9bbDysuiWeo2VShFJJtcw	Cat	133
-FZBTkAZEXoP7CYvRV2ZwQ	William	126
-9da1xk7zgnnfO1uTVYGkA	Fran	124
-lh59ko3dxChBSZ9U7LfUw	Lissa	120

Part 2: Inferences and Analysis

1. Pick one city and category of your choice and group the businesses in that city or category by their overall star rating. Compare the businesses with 2-3 stars to the businesses with 4-5 stars and answer the following questions. Include your code.

I chose city as Las Vegas and chose categories as shopping as they has 82854 reviews and 4 business subsequently.

i. Do the two groups you chose to analyze have a different distribution of hours?

For shopping category, only one business qualified for the 2-3 group i.e., Walgreens and two for the 4-5 group i.e., Desert Medical Equipment and Red Rock Canyon Visitor Centre. On comparison, Walgreens is open every day from 8am to 10pm whereas the ones in 4-5 rating are open from 8 am to 4:30pm and 5 pm respectively. So, there is a stark difference in the distribution of hours.

ii. Do the two groups you chose to analyze have a different number of reviews?

Comparing the reviews of the businesses, Walgreens has just 6 reviews whereas the 4-5-star businesses have 4 and 32 reviews respectively. It seems the one with 32 reviews i.e., Red Rock Canyon Visitor Centre is quite liked by the residents with 4.5-star rating.

iii. Are you able to infer anything from the location data provided between these two groups? Explain.

The location and neighborhood don't provide sufficient details to reach any major conclusion. All the different businesses have different postal code and neighborhood. So, due to not having sufficient data, no concrete conclusion can be reached on the correlation between star rating and location detail.

SQL code used for analysis:

```
SELECT
CASE
    WHEN stars>=4 THEN '4-5 Stars'
    WHEN (stars>=2 AND stars<=3) THEN '2-3 Stars'
    END as rating,

postal_code,
review_count,
hours.hours,
name,
neighborhood
FROM business INNER JOIN category
     ON business.id=category.business_id
     INNER JOIN hours
     ON business.id=hours.business_id
WHERE city='Las Vegas'
```

```
                AND category = 'Shopping'
                AND (stars>=4 OR (stars <3 and stars>2))
            ORDER BY stars DESC, hours DESC
```

Result:

rating	review_count	hours	name	neighborhood
4-5 Stars	4	Wednesday|8:00-17:00	Desert Medical Equipment	
4-5 Stars	4	Tuesday|8:00-17:00	Desert Medical Equipment	
4-5 Stars	4	Thursday|8:00-17:00	Desert Medical Equipment	
4-5 Stars	4	Monday|8:00-17:00	Desert Medical Equipment	
4-5 Stars	4	Friday|8:00-17:00	Desert Medical Equipment	
4-5 Stars	32	Wednesday|8:00-16:30	Red Rock Canyon Visitor Center	
4-5 Stars	32	Tuesday|8:00-16:30	Red Rock Canyon Visitor Center	
4-5 Stars	32	Thursday|8:00-16:30	Red Rock Canyon Visitor Center	
4-5 Stars	32	Sunday|8:00-16:30	Red Rock Canyon Visitor Center	
4-5 Stars	32	Saturday|8:00-16:30	Red Rock Canyon Visitor Center	
4-5 Stars	32	Monday|8:00-16:30	Red Rock Canyon Visitor Center	
4-5 Stars	32	Friday|8:00-16:30	Red Rock Canyon Visitor Center	
2-3 Stars	6	Wednesday|8:00-22:00	Walgreens	Eastside
2-3 Stars	6	Tuesday|8:00-22:00	Walgreens	Eastside
2-3 Stars	6	Thursday|8:00-22:00	Walgreens	Eastside
2-3 Stars	6	Sunday|8:00-22:00	Walgreens	Eastside
2-3 Stars	6	Saturday|8:00-22:00	Walgreens	Eastside
2-3 Stars	6	Monday|8:00-22:00	Walgreens	Eastside
2-3 Stars	6	Friday|8:00-22:00	Walgreens	Eastside

2. Group business based on the ones that are open and the ones that are closed. What differences can you find between the ones that are still open and the ones that are closed? List at least two differences and the SQL code you used to arrive at your answer.

i. Difference 1:

The businesses that are open tend to have more reviews than ones that are closed on average.
Open: AVG(review_count) = 31.757
Closed: AVG(review_count) = 23.198

ii. Difference 2:

The average star rating is higher for businesses that are open than businesses that are closed.
Open: AVG(stars) = 3.68
Closed: AVG(stars) = 3.52

SQL code used for analysis:

```
SELECT
COUNT(DISTINCT(id)) AS Number_of_business,
ROUND(AVG(review_count),2) AS avg_review,
SUM(review_count) AS total_review,
ROUND(AVG(stars),2) AS avg_rating,
is_open
FROM business
GROUP BY is_open
```

Result:

Number_of_business	avg_review	total_review	avg_rating	is_open
1520	23.2	35261	3.52	0
8480	31.76	269300	3.68	1

3. For this last part of your analysis, you are going to choose the type of analysis you want to conduct on the Yelp dataset and are going to prepare the data for analysis.

Ideas for analysis include: Parsing out keywords and business attributes for sentiment analysis, clustering businesses to find commonalities or anomalies between them, predicting the overall star rating for a business, predicting the number of fans a user will have, and so on. These are just a few examples to get you started, so feel free to be creative and come up with your own problem you want to solve. Provide answers, in-line, to all of the following:

i. Indicate the type of analysis you chose to do:

Predicting the number of fans a user will have is an interesting problem to consider according to me.

ii. Write 1-2 brief paragraphs on the type of data you will need for your analysis and why you chose that data:

After studying the ER diagram, there are a few points that came to mind that can be useful for predicting the number of fans a user will have like the number of useful reviews, years active on yelp, whether he/she is an elite member and for how long, compliments received from users etc. There could have been a lot of other analysis that could have been done based on other factors like the quality of the review made, review sentiment analysis, rating of the business for which the review was made etc. but I haven't considered all these in the current analysis.

So, I have tried to use the user table and the elite_years table for my analysis. The result of the analysis are as follows:

1) Being an elite member doesn't have much of an effect on the number of fans as most of the high fan user weren't elite members ever.

2) On an average, a user has been on yelp for 7 years.
3) Now for fan prediction, it seems on an average each review can count towards 0.033 fans i.e. on an average one can expect 1 fan for every 30 reviews posted
4) Other observations are, on an average, one can expect 1 fan for every 4 rating for usefulness given by users and on average 1 fan for every 5.5 rating for compliment given by users

iii. Output of your finished dataset:

Answer will be show in the iv part along with the code.

iv. Provide the SQL code you used to create your final dataset:

```
SELECT
name,
DATE('NOW')-yelping_since AS years_act,
(MAX(year)-MIN(year)) AS elite_Y,
ROUND(fans*1.0/review_count,2) as fans_per_rev,
ROUND(fans*1.0/useful,2) as fans_per_useful,
ROUND(fans*1.0/(funny+cool+compliment_hot+ compliment_more+ compliment_profile+ c
ompliment_cute+ compliment_list + compliment_note + compliment_plain + compliment
_cool + compliment_funny + compliment_writer + compliment_photos),2) AS fans_per_
compl
FROM user LEFT JOIN elite_years
ON user.id=elite_years.user_id
GROUP BY name
ORDER BY review_count DESC
LIMIT 10
```

Result:

name	years_act	elite_Y	fans_per_rev	fans_per_useful	fans_per_compl
Gerald	11	None	0.13	0.01	0.01
.Hon	17	None	0.08	0.01	0.01
eric	16	None	0.01	16.0	0.24
Roanna	17	None	0.1	0.03	0.02
Ed	14	7	0.04	0.27	0.13
Dominic	12	6	0.04	0.46	0.13
Lissa	16	8	0.14	0.26	0.04
Alison	16	None	0.08	0.2	0.03
Sui	14	None	0.1	8.67	0.27
Crissy	15	None	0.04	6.25	0.24

```
SELECT
AVG(DATE('NOW')-yelping_since) AS years_active,
AVG(ROUND(fans*1.0/review_count,2)) as fans_per_review,
AVG(ROUND(fans*1.0/useful,2)) as fans_per_useful,
AVG(ROUND(fans*1.0/(funny+cool+compliment_hot+ compliment_more+ compliment_profil
e+ compliment_cute+ compliment_list + compliment_note + compliment_plain + compli
ment_cool + compliment_funny + compliment_writer + compliment_photos),2)) AS fans
_per_compliment
FROM user;
```

Result:

```
+--------------+-----------------+-----------------+----------------------+
| years_active | fans_per_review | fans_per_useful | fans_per_compliment |
+--------------+-----------------+-----------------+----------------------+
|       9.1995 | 0.0329976979281 |  0.240480232298 |      0.179715921136 |
+--------------+-----------------+-----------------+----------------------+
```