Programming for Big Data Analytics

Homework 3

Submitted By: Harsh Yadav

Net Id: hy1217

Question1:

Spark Code:

%spark.pyspark

import pyspark

%pyspark

df=spark.read.json("/shared/d/business.json")

#First method, by not exploding the categories

df1 = df.groupBy("city","categories").agg(avg("review_count"), avg("stars"))

df1.show()

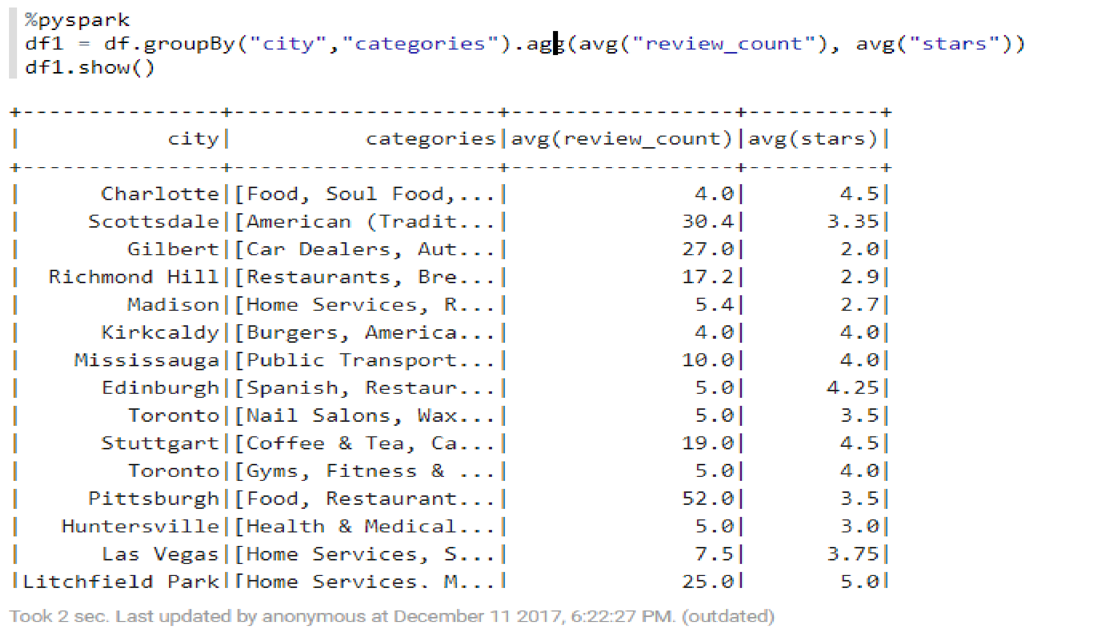
#Second method, by exploding categories and then grouping

a = df.withColumn("category", explode(col("categories")))

a.groupBy("city","category").agg(avg("review_count"), avg("stars")).show()

Spark Output:

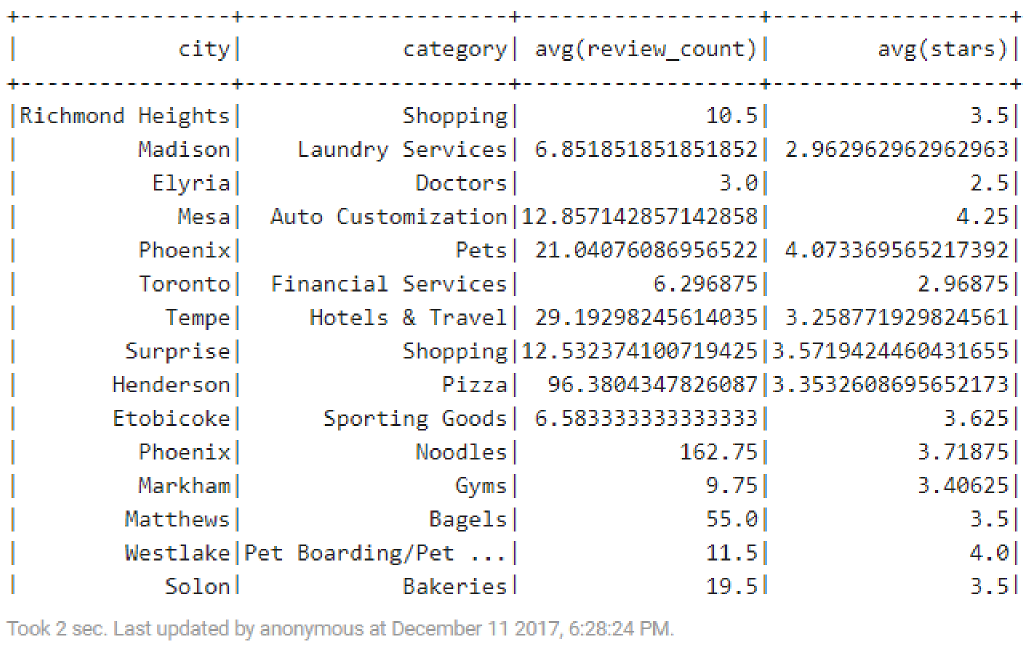
Without Exploding the categories:



```
%pyspark
df1 = df.groupBy("city","categories").agg(avg("review_count"), avg("stars"))
df1.show()

+---------------+--------------------+-----------------+----------+
|           city|          categories|avg(review_count)|avg(stars)|
+---------------+--------------------+-----------------+----------+
|      Charlotte|[Food, Soul Food,...|              4.0|       4.5|
|     Scottsdale|[American (Tradit...|             30.4|      3.35|
|        Gilbert|[Car Dealers, Aut...|             27.0|       2.0|
|  Richmond Hill|[Restaurants, Bre...|             17.2|       2.9|
|        Madison|[Home Services, R...|              5.4|       2.7|
|      Kirkcaldy|[Burgers, America...|              4.0|       4.0|
|    Mississauga|[Public Transport...|             10.0|       4.0|
|      Edinburgh|[Spanish, Restaur...|              5.0|      4.25|
|        Toronto|[Nail Salons, Wax...|              5.0|       3.5|
|      Stuttgart|[Coffee & Tea, Ca...|             19.0|       4.5|
|        Toronto|[Gyms, Fitness & ...|              5.0|       4.0|
|     Pittsburgh|[Food, Restaurant...|             52.0|       3.5|
|   Huntersville|[Health & Medical...|              5.0|       3.0|
|      Las Vegas|[Home Services, S...|              7.5|      3.75|
|Litchfield Park|[Home Services, M...|             25.0|       5.0|
```

Took 2 sec. Last updated by anonymous at December 11 2017, 6:22:27 PM. (outdated)

Exploding categories into distinct values:



```
+----------------+--------------------+------------------+------------------+
|            city|            category| avg(review_count)|        avg(stars)|
+----------------+--------------------+------------------+------------------+
|Richmond Heights|            Shopping|              10.5|               3.5|
|         Madison|    Laundry Services| 6.851851851851852| 2.962962962962963|
|          Elyria|             Doctors|               3.0|               2.5|
|            Mesa|  Auto Customization|12.857142857142858|              4.25|
|         Phoenix|                Pets| 21.04076086956522| 4.073369565217392|
|         Toronto|  Financial Services|          6.296875|           2.96875|
|           Tempe|      Hotels & Travel| 29.19298245614035| 3.258771929824561|
|        Surprise|            Shopping|12.532374100719425|3.5719424460431655|
|       Henderson|               Pizza|  96.3804347826087|3.3532608695652173|
|       Etobicoke|      Sporting Goods| 6.583333333333333|             3.625|
|         Phoenix|             Noodles|            162.75|           3.71875|
|         Markham|                Gyms|              9.75|           3.40625|
|        Matthews|              Bagels|              55.0|               3.5|
|        Westlake|Pet Boarding/Pet ...|              11.5|               4.0|
|           Solon|            Bakeries|              19.5|               3.5|
```

Took 2 sec. Last updated by anonymous at December 11 2017, 6:28:24 PM.

Pig Code:

register elephant-bird-hadoop-compat-4.1.jar

register elephant-bird-pig-4.1.jar

register json-simple-1.1.1.jar

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

B = FOREACH A GENERATE (int)json#'review_count' AS review_count, (float)json#'stars' AS stars, json#'city' as city, FLATTEN(json#'categories') AS categories;

C = GROUP B BY (city,categories);

D = FOREACH C GENERATE group.city as city, group.categories as category,AVG(B.review_count) AS reviewCount, AVG(B.stars) AS stars;

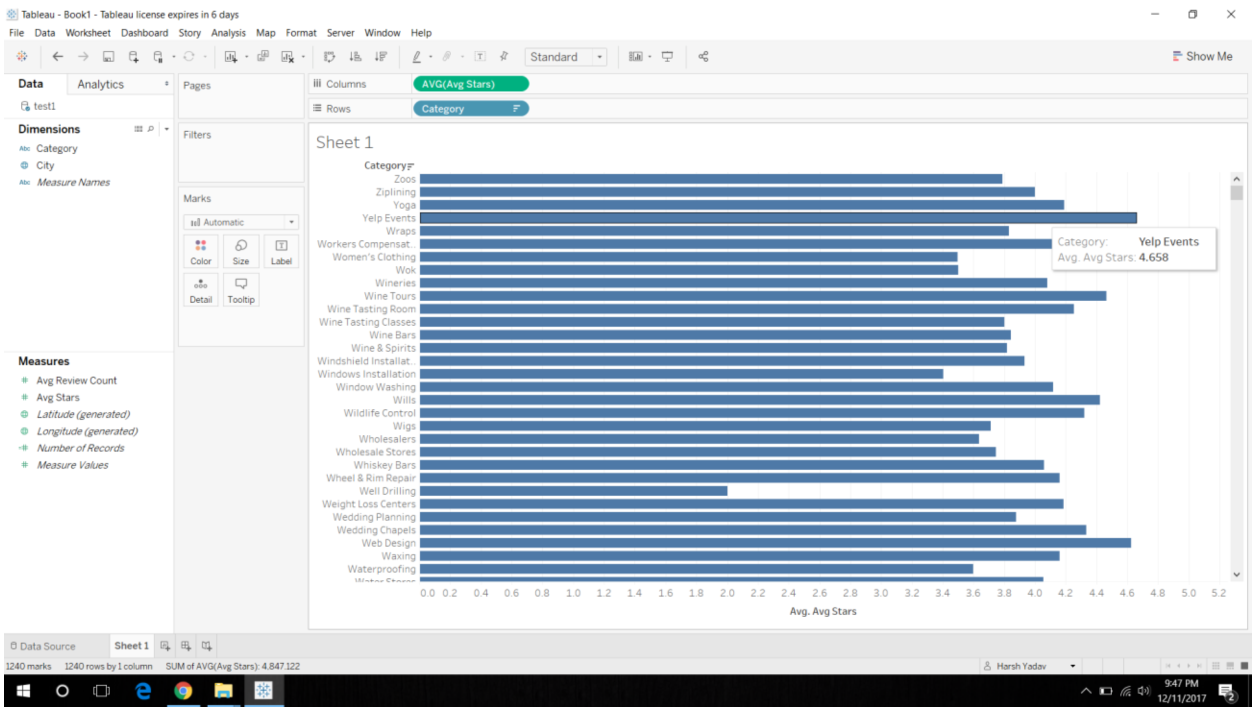
STORE D INTO './answer1.out';

Pig Output:

Output file has been attached in the zip folder named Pig Output.

Visualization:

Categories by average number of stars:

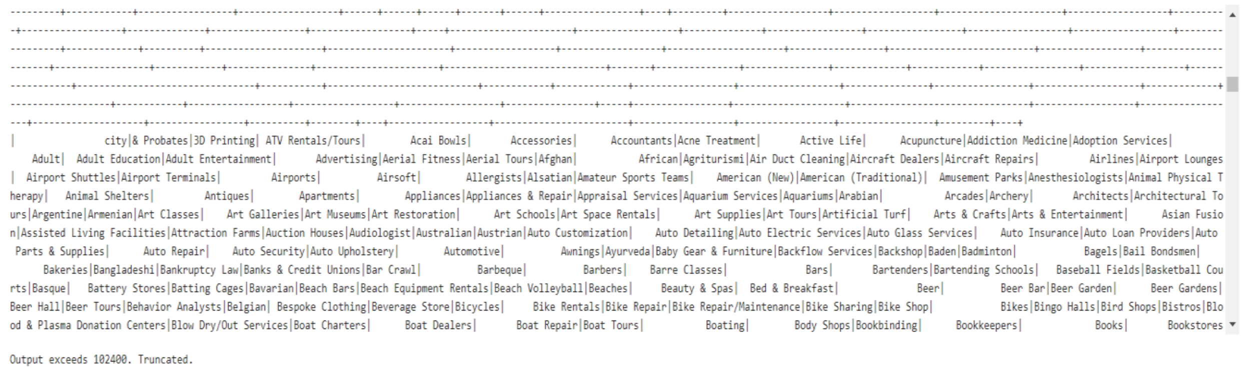


Question2:

Spark Code:

```
%spark.pyspark

import pyspark

%pyspark

df=spark.read.json("/shared/d/business.json")

a = df.withColumn("category", explode(col("categories")))

a.groupBy("city").pivot("category").avg("stars").show()
```

Spark Output:



```
---------+------------------+--------------------+-----------+-----+--------------------+------------------+------------------+------------------+------------------+------------------+------
-+-----------------+--------------+------------------+------------------+------------------+------------------+---------------------+--------------------+--------------------+----------------
--------+-----------------+---------------+--------------------+--------------------+---------------+---+---------+------------------+---------------+---------------+---------------+------
-------+----------------+-------------+--------------------+--------------------------+----------+-------------+----------------+----------------+-----------------+----------------+-----
----------+----------------------------------+------------+------------------------+-----------+---------+------------+-------------+----------------+----------------+----------------+
----------------+-----------+--------------------+--------------------+----------------+----+----------------+-----------------+----------------+------------------+----------------
---+-------------------+------------------+----------+---------+---+-------------------+------------------+-----------------+------------------+-----------------+-------+---+
|               city|& Probates|3D Printing| ATV Rentals/Tours|        Acai Bowls|       Accessories|       Accountants|Acne Treatment|       Active Life|       Acupuncture|Addiction Medicine|Adoption Services|
   Adult|  Adult Education|Adult Entertainment|       Advertising|Aerial Fitness|Aerial Tours|Afghan|           African|Agriturismi|Air Duct Cleaning|Aircraft Dealers|Aircraft Repairs|         Airlines|Airport Lounges
|  Airport Shuttles|Airport Terminals|         Airports|          Airsoft|        Allergists|Alsatian|Amateur Sports Teams|    American (New)|American (Traditional)|  Amusement Parks|Anesthesiologists|Animal Physical T
herapy|   Animal Shelters|          Antiques|         Apartments|        Appliances|Appliances & Repair|Appraisal Services|Aquarium Services|Aquariums|Arabian|           Arcades|Archery|        Architects|Architectural To
urs|Argentine|Armenian|Art Classes|     Art Galleries|Art Museums|Art Restoration|       Art Schools|Art Space Rentals|       Art Supplies|Art Tours|Artificial Turf|    Arts & Crafts|Arts & Entertainment|       Asian Fusio
n|Assisted Living Facilities|Attraction Farms|Auction Houses|Audiologist|Australian|Austrian|Auto Customization|    Auto Detailing|Auto Electric Services|Auto Glass Services|    Auto Insurance|Auto Loan Providers|Auto
 Parts & Supplies|       Auto Repair|     Auto Security|Auto Upholstery|         Automotive|          Awnings|Ayurveda|Baby Gear & Furniture|Backflow Services|Backshop|Baden|Badminton|          Bagels|Bail Bondsmen|
      Bakeries|Bangladeshi|Bankruptcy Law|Banks & Credit Unions|Bar Crawl|           Barbeque|           Barbers|    Barre Classes|                Bars|         Bartenders|Bartending Schools|   Baseball Fields|Basketball Cou
rts|Basque|   Battery Stores|Batting Cages|Bavarian|Beach Bars|Beach Equipment Rentals|Beach Volleyball|Beaches|      Beauty & Spas|  Bed & Breakfast|              Beer|          Beer Bar|Beer Garden|        Beer Gardens|
Beer Hall|Beer Tours|Behavior Analysts|Belgian| Bespoke Clothing|Beverage Store|Bicycles|       Bike Rentals|Bike Repair|Bike Repair/Maintenance|Bike Sharing|Bike Shop|          Bikes|Bingo Halls|Bird Shops|Bistros|Blo
od & Plasma Donation Centers|Blow Dry/Out Services|Boat Charters|      Boat Dealers|      Boat Repair|Boat Tours|          Boating|        Body Shops|Bookbinding|      Bookkeepers|              Books|        Bookstores

Output exceeds 102400. Truncated.
```

Pig Code:

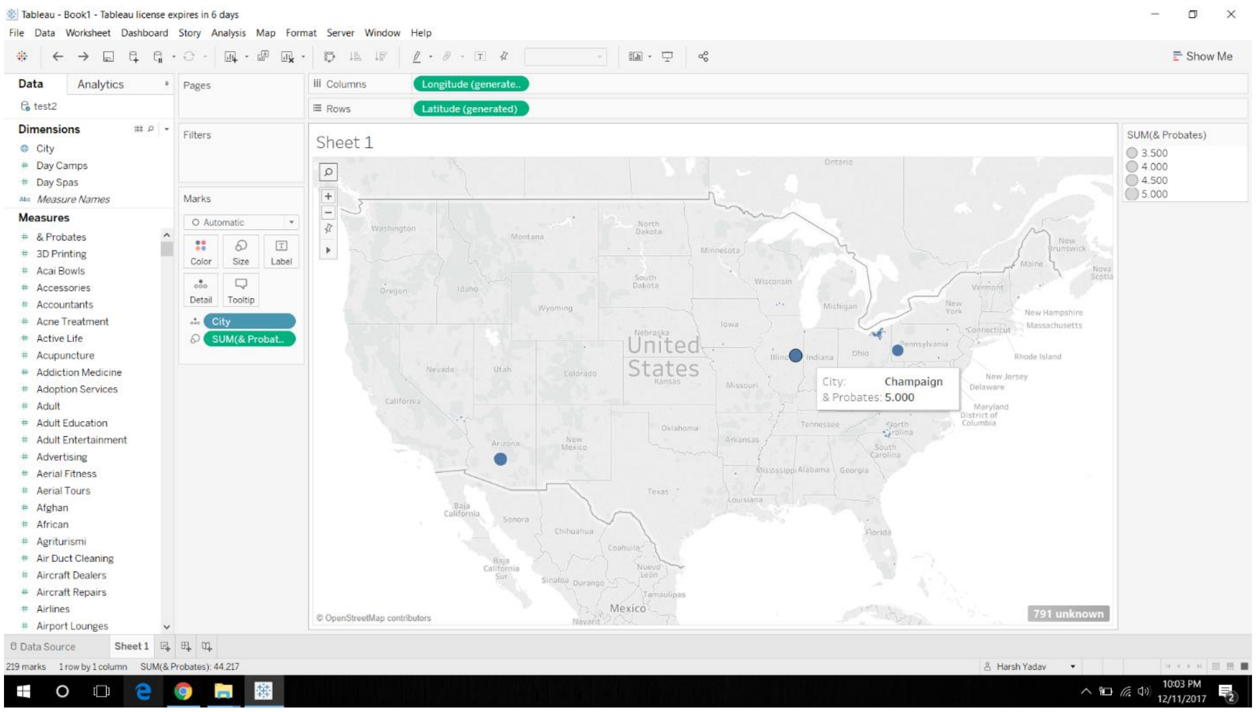
```
A = LOAD 'business.json' USING com.twitter.elephantbird.pig.load.JsonLoader('-nestedLoad') AS
(json:map []);

B = FOREACH A GENERATE (int)json#'review_count' AS review_count, (float)json#'stars' AS stars,
json#'city' as city, FLATTEN(json#'categories') AS categories;

C = GROUP B BY city;

D = FOREACH C GENERATE group, (B.categories) AS categories, AVG(B.stars) AS stars;

STORE D INTO './answer2.out';
```

Pig Output:

Output file is stored in the Zip Folder.

Visualization:

Representation of one category in different parts of US.



From this visualization we can observe that most of the categories are not present all over the US, but only at some place.

Question 3:

Spark Code:

%spark.pyspark

import pyspark

%pyspark

df=spark.read.json("/shared/d/business.json")

from pyspark.sql.functions import split, explode

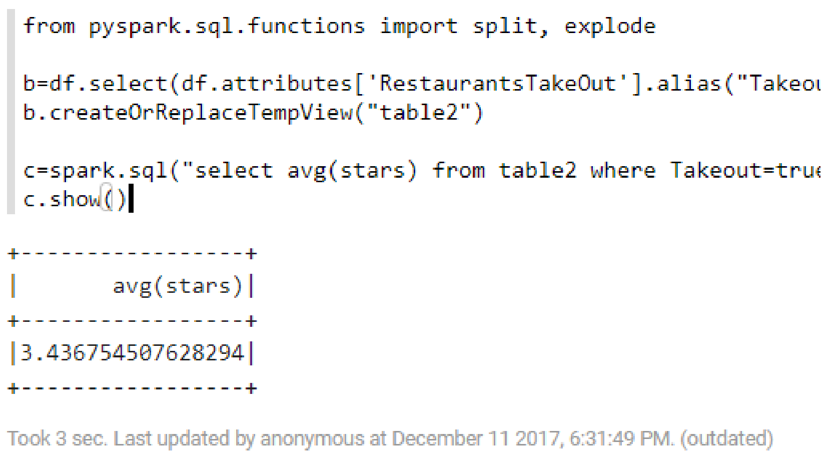
b=df.select(df.attributes['RestaurantsTakeOut'].alias("Takeout"), df.business_id, explode(df.categories).alias("category"), df.stars)

b.createOrReplaceTempView("table2")

c=spark.sql("select avg(stars) from table2 where Takeout=true and category='Mexican'")

c.show()

Spark Output:



```
from pyspark.sql.functions import split, explode

b=df.select(df.attributes['RestaurantsTakeOut'].alias("Takeou
b.createOrReplaceTempView("table2")

c=spark.sql("select avg(stars) from table2 where Takeout=true
c.show()

+-----------------+
|       avg(stars)|
+-----------------+
|3.436754507628294|
+-----------------+
```

Took 3 sec. Last updated by anonymous at December 11 2017, 6:31:49 PM. (outdated)

Pig Code:

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

B = FOREACH A GENERATE FLATTEN(json#'categories') as categories, json#'attributes'#'RestaurantsTakeOut' as attribute, (float)json#'stars' as stars;

C = FILTER B BY attribute matches '.*true.*';

D = FILTER C BY categories matches 'Mexican';

E = GROUP D BY (categories);

F = FOREACH E GENERATE group, AVG(D.stars) as avg_stars;

store F into './answer3.out';

Pig Output:

Output file is present in the Zip folder.

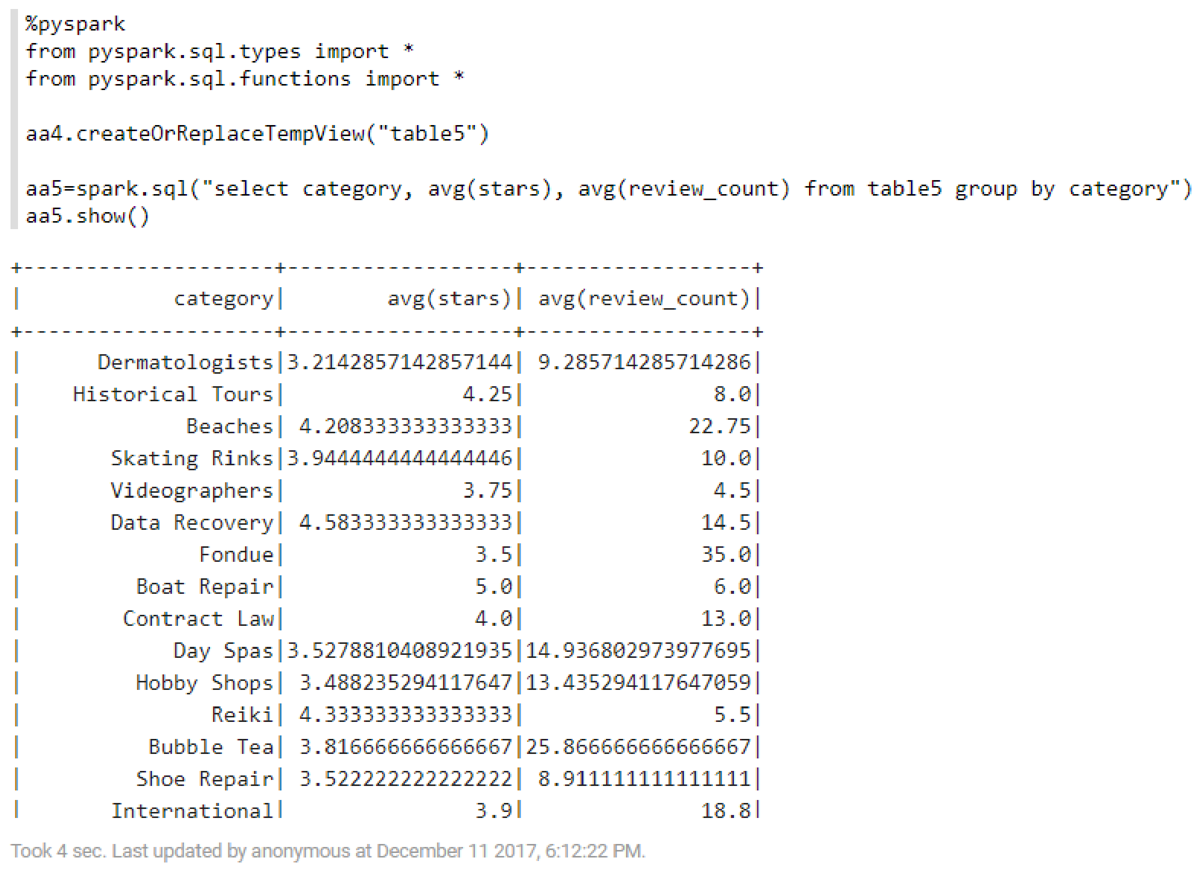
From this part, we can infer that the restaurants who offer Mexican food and offer take out have good star ratings.

Question 4:

Spark Code:

```
%spark.pyspark

import pyspark

%pyspark

df=spark.read.json("/shared/d/business.json")

from pyspark.sql.functions import split, explode, col

d = df.withColumn("category", explode(col("categories")))

d.createOrReplaceTempView("table4")

aa=spark.sql("select business_id,latitude, longitude, category, stars, review_count from table4")

from pyspark.sql.functions import *

import math

from math import radians, cos, sin, asin, sqrt, atan2, pi

from pyspark.sql.types import *

aa2=aa.withColumn('latitude_r', (aa.latitude*pi)/180)

aa3=aa2.withColumn('longitude_r', (aa.longitude*pi)/180)

lat_tor=((43.6532)*pi/180)

lon_tor=((-79.3832)*pi/180)

aa3.createOrReplaceTempView("table111")

aa4=spark.sql("SELECT * FROM table111 WHERE acos(sin(0.7618921) * sin(latitude_r) + cos(0.7618921)
* cos(latitude_r) * cos(longitude_r - (-1.3855))) * 6371 <= 15")

aa4.createOrReplaceTempView("table5")

aa5=spark.sql("select category, avg(stars), avg(review_count) from table5 group by category")

aa5.show()
```

Spark Output:



```
%pyspark
from pyspark.sql.types import *
from pyspark.sql.functions import *

aa4.createOrReplaceTempView("table5")

aa5=spark.sql("select category, avg(stars), avg(review_count) from table5 group by category")
aa5.show()
```

```
+-------------------+------------------+------------------+
|           category|        avg(stars)| avg(review_count)|
+-------------------+------------------+------------------+
|      Dermatologists|3.2142857142857144| 9.285714285714286|
|    Historical Tours|              4.25|               8.0|
|             Beaches| 4.208333333333333|             22.75|
|      Skating Rinks|3.9444444444444446|              10.0|
|        Videographers|              3.75|               4.5|
|       Data Recovery| 4.583333333333333|              14.5|
|              Fondue|               3.5|              35.0|
|         Boat Repair|               5.0|               6.0|
|        Contract Law|               4.0|              13.0|
|            Day Spas|3.527881040892193|14.936802973977695|
|         Hobby Shops| 3.488235294117647|13.435294117647059|
|               Reiki| 4.333333333333333|               5.5|
|          Bubble Tea| 3.816666666666667|25.866666666666667|
|         Shoe Repair| 3.522222222222222| 8.911111111111111|
|       International|               3.9|              18.8|
```

Took 4 sec. Last updated by anonymous at December 11 2017, 6:12:22 PM.

Pig Code:

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

B = FOREACH A GENERATE FLATTEN(json#'categories') as categories, (float)json#'stars' as stars, (int)json#'review_count' as review_count, (float)json#'latitude' as latitude, (float)json#'longitude' as longitude;

C = FILTER B BY latitude<43.7889 AND latitude>43.5182 AND longitude< -79.1971 AND longitude> -79.5694;

category1 = GROUP C BY (categories);

D = FOREACH category1 GENERATE group, AVG(C.review_count), AVG(C.stars);

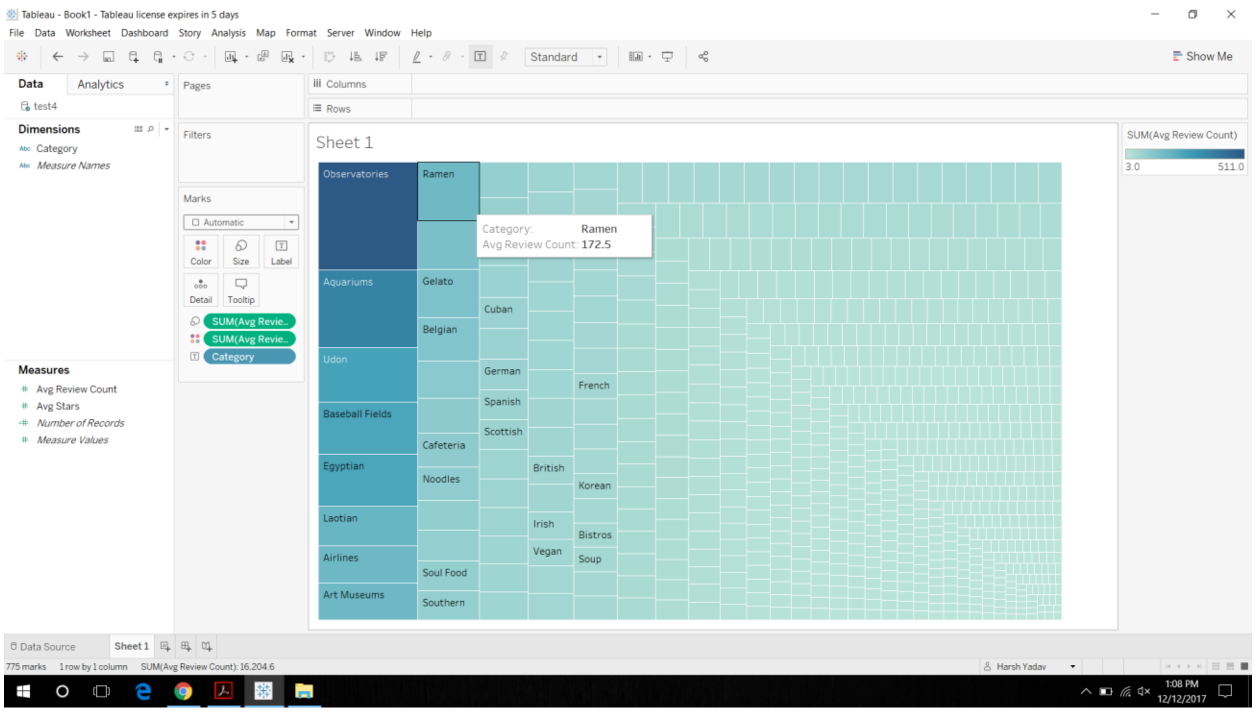
store D into './answer4.out';

Pig Output:

Output file is stored in the zip folder.

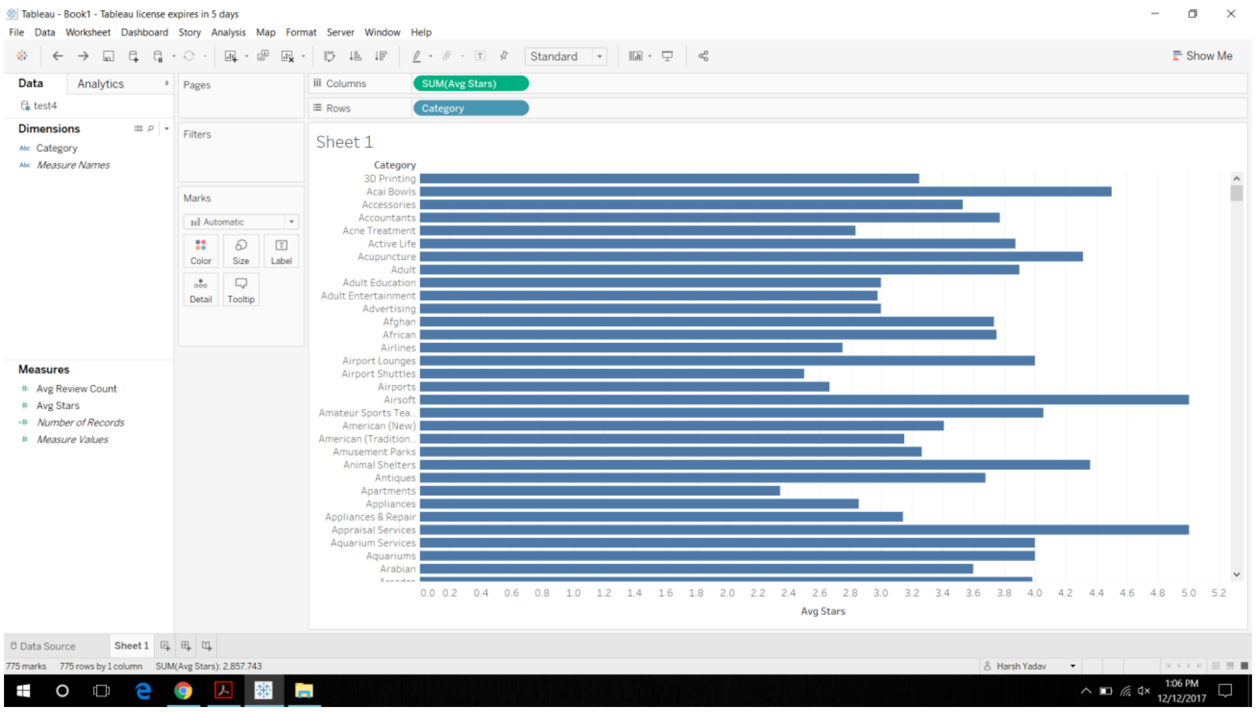
Visualization:

Average Review count by Category:



This shows that Ramen Category has the highest average review count.

Average Stars with category:



Question 5:

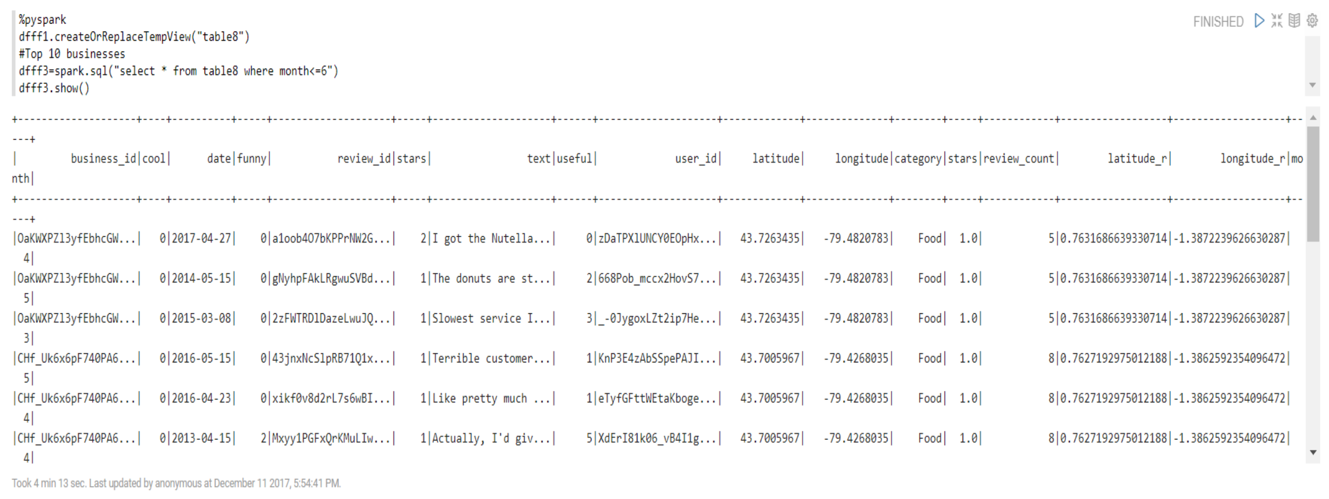
Spark Code:

```
%spark.pyspark

import pyspark

%pyspark

df=spark.read.json("/shared/d/business.json")

from pyspark.sql.functions import split, explode, col

d = df.withColumn("category", explode(col("categories")))

d.createOrReplaceTempView("table4")

aa=spark.sql("select business_id,latitude, longitude, category, stars, review_count from table4")


from pyspark.sql.functions import *

import math

from math import radians, cos, sin, asin, sqrt, atan2, pi

from pyspark.sql.types import *


aa2=aa.withColumn('latitude_r', (aa.latitude*pi)/180)

aa3=aa2.withColumn('longitude_r', (aa.longitude*pi)/180)


lat_tor=((43.6532)*pi/180)

lon_tor=((-79.3832)*pi/180)


aa3.createOrReplaceTempView("table111")

aa4=spark.sql("SELECT * FROM table111 WHERE acos(sin(0.7618921) * sin(latitude_r) + cos(0.7618921)
* cos(latitude_r) * cos(longitude_r - (-1.3855))) * 6371 <= 15")


aa4.createOrReplaceTempView("table5")

aa5=spark.sql("select * from table5 where category=='Food'")


aa5.createOrReplaceTempView("table7")

df4=spark.sql("select * from table7 order by stars asc limit 10")

df5=spark.sql("select * from table7 order by stars desc limit 10")
```

```
df1=spark.read.json("/shared/d/review.json")


df1 = df1.alias('df1')

df2 = df4.alias('df2')

df3 = df5.alias('df3')


dff1=df1.join(df2, "business_id")

dff2=df1.join(df3, "business_id")


dfff1=dff1.withColumn('month', concat(dff1.date.substr(6,2)))

dfff2=dff2.withColumn('month', concat(dff2.date.substr(6,2)))


dfff1=dfff1.withColumn("month", dfff1["month"].cast(IntegerType()))

dfff2=dfff2.withColumn("month", dfff2["month"].cast(IntegerType()))


dfff1.createOrReplaceTempView("table8")

#Top 10 businesses

dfff3=spark.sql("select * from table8 where month<=6")

dfff3.show()


dfff2.createOrReplaceTempView("table9")

dfff4=spark.sql("select * from table9 where month<=6")

#For bottom 10 businesses

dfff4.show()
```

Spark Output:

Top 10:



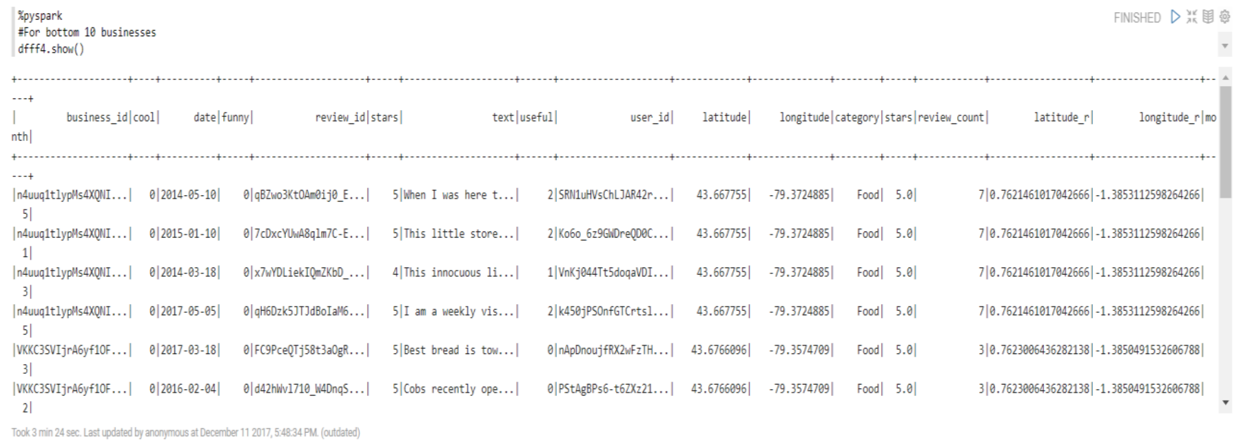
```
%pyspark
dfff1.createOrReplaceTempView("table8")
#Top 10 businesses
dfff3=spark.sql("select * from table8 where month<=6")
dfff3.show()
```

FINISHED

```
+--------------------+----+----------+-----+--------------------+-----+--------------------+------+--------------------+----------+-----------+--------+-----+------------+------------------+------------------+--
---+
|         business_id|cool|      date|funny|           review_id|stars|                text|useful|             user_id|  latitude|  longitude|category|stars|review_count|        latitude_r|       longitude_r|mo
nth|
+--------------------+----+----------+-----+--------------------+-----+--------------------+------+--------------------+----------+-----------+--------+-----+------------+------------------+------------------+--
---+
|OaKWXPZl3yfEbhcGW...|   0|2017-04-27|    0|a1oob4O7bKPPrNW2G...|    2|I got the Nutella...|     0|zDaTPX1UNCY0EOpHx...|43.7263435|-79.4820783|    Food|  1.0|           5|0.7631686639330714|-1.387223962663028 7|
    4|
|OaKWXPZl3yfEbhcGW...|   0|2014-05-15|    0|gNyhpFAkLRgwuSVBd...|    1|The donuts are st...|     2|668Pob_mccx2HovS7...|43.7263435|-79.4820783|    Food|  1.0|           5|0.7631686639330714|-1.3872239626630287|
    5|
|OaKWXPZl3yfEbhcGW...|   0|2015-03-08|    0|2zFWTRD1DazeLwuJQ...|    1|Slowest service I...|     3|_-0JygoxLZt2ip7He...|43.7263435|-79.4820783|    Food|  1.0|           5|0.7631686639330714|-1.3872239626630287|
    3|
|CHf_Uk6x6pF740PA6...|   0|2016-05-15|    0|43jnxNcSlpRB71Q1x...|    1|Terrible customer...|     1|KnP3E4zAbSSpePAJI...|43.7005967|-79.4268035|    Food|  1.0|           8|0.7627192975012188|-1.3862592354096472|
    5|
|CHf_Uk6x6pF740PA6...|   0|2016-04-23|    0|xikf0v8d2rL7s6wBI...|    1|Like pretty much ...|     1|eTyfGFttWEtaKboge...|43.7005967|-79.4268035|    Food|  1.0|           8|0.7627192975012188|-1.3862592354096472|
    4|
|CHf_Uk6x6pF740PA6...|   0|2013-04-15|    2|Mxyy1PGFxQrKMuLIw...|    1|Actually, I'd giv...|     5|XdErI81k06_vB4I1g...|43.7005967|-79.4268035|    Food|  1.0|           8|0.7627192975012188|-1.3862592354096472|
    4|
```

Took 4 min 13 sec. Last updated by anonymous at December 11 2017, 5:54:41 PM.

Bottom 10:



```
%pyspark
#For bottom 10 businesses
dfff4.show()
```

FINISHED

```
+--------------------+----+----------+-----+--------------------+-----+--------------------+------+--------------------+----------+-----------+--------+-----+------------+------------------+------------------+--
---+
|         business_id|cool|      date|funny|           review_id|stars|                text|useful|             user_id|  latitude|  longitude|category|stars|review_count|        latitude_r|       longitude_r|mo
nth|
+--------------------+----+----------+-----+--------------------+-----+--------------------+------+--------------------+----------+-----------+--------+-----+------------+------------------+------------------+--
---+
|n4uuq1tlypMs4XQNI...|   0|2014-05-10|    0|qBZwo3KtOAm0ij0_E...|    5|When I was here t...|     2|SRN1uHVsChLJAR42r...| 43.667755|-79.3724885|    Food|  5.0|           7|0.7621461017042666|-1.3853112598264266|
    5|
|n4uuq1tlypMs4XQNI...|   0|2015-01-10|    0|7cDxcYUwA8qlm7C-E...|    5|This little store...|     2|Ko6o_6z9GWDreQD0C...| 43.667755|-79.3724885|    Food|  5.0|           7|0.7621461017042666|-1.3853112598264266|
    1|
|n4uuq1tlypMs4XQNI...|   0|2014-03-18|    0|x7wYDLiekIQmZKbD_...|    4|This innocuous li...|     1|VnKj044Tt5doqaVDI...| 43.667755|-79.3724885|    Food|  5.0|           7|0.7621461017042666|-1.3853112598264266|
    3|
|n4uuq1tlypMs4XQNI...|   0|2017-05-05|    0|qH6Dzk5JTJdBoIaM6...|    5|I am a weekly vis...|     2|k450jPSOnfGTCrtsl...| 43.667755|-79.3724885|    Food|  5.0|           7|0.7621461017042666|-1.3853112598264266|
    5|
|VKKC3SVIjrA6yf1OF...|   0|2017-03-18|    0|FC9PceQTj58t3aOgR...|    5|Best bread is tow...|     0|nApDnoujfRX2wFzTH...|43.6766096|-79.3574709|    Food|  5.0|           3|0.7623006436282138|-1.3850491532606788|
    3|
|VKKC3SVIjrA6yf1OF...|   0|2016-02-04|    0|d42hWvl710_W4DnqS...|    5|Cobs recently ope...|     0|PStAgBPs6-t6ZXz21...|43.6766096|-79.3574709|    Food|  5.0|           3|0.7623006436282138|-1.3850491532606788|
    2|
```

Took 3 min 24 sec. Last updated by anonymous at December 11 2017, 5:48:34 PM. (outdated)

Pig Code:

A = LOAD 'business.json' USING com.twitter.elephantbird.pig.load.JsonLoader('-nestedLoad') AS (json1:map []);

B = FOREACH A GENERATE json1#'business_id' as businessid, FLATTEN(json1#'categories') as categories, (float)json1#'stars' as stars, (int)json1#'review_count' as review_count, (float)json1#'latitude' as latitude, (float)json1#'longitude' as longitude;

C = FILTER B BY latitude<43.7889 AND latitude>43.5182 AND longitude< -79.1971 AND longitude> -79.5694;

D = FILTER C BY categories matches 'Food';

E = ORDER D BY stars DESC;

TOP = LIMIT E 10;

F = ORDER D BY stars ASC;

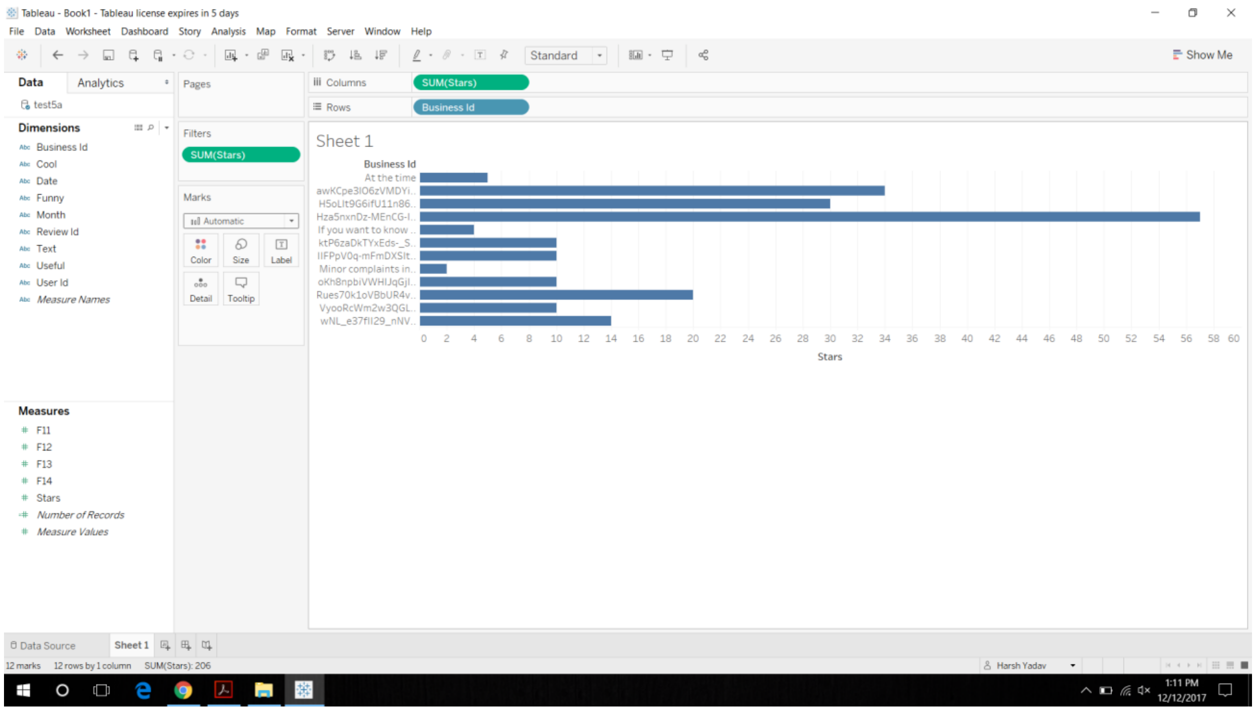
```
BOTTOM = LIMIT F 10;

G = LOAD 'review.json' USING com.twitter.elephantbird.pig.load.JsonLoader('-nestedLoad') AS
(json2:map []);

H = FOREACH G GENERATE json2#'business_id' as businessid, (float)json2#'stars' as stars, json2#'date' as
date;

TOP_REVIEW = JOIN H BY businessid, TOP BY businessid;

data1 = FOREACH TOP_REVIEW GENERATE (INT)SUBSTRING (H::date, 5,7) as dateint, H::businessid as
businessid, H::stars as stars;

TOP_MONTH = FILTER data1 BY dateint >= 1 AND  dateint < 6;

store TOP_MONTH into './answer5_top.out';

BOTTOM_REVIEW = JOIN H BY businessid, BOTTOM BY businessid;

data2 = FOREACH BOTTOM_REVIEW GENERATE (INT)SUBSTRING (H::date, 5,7) as dateint, H::businessid
as businessid, H::stars as stars;

BOTTOM_MONTH = FILTER data2 BY dateint < 6;

store BOTTOM_MONTH into './answer5_bottom.out';
```

Pig Output:

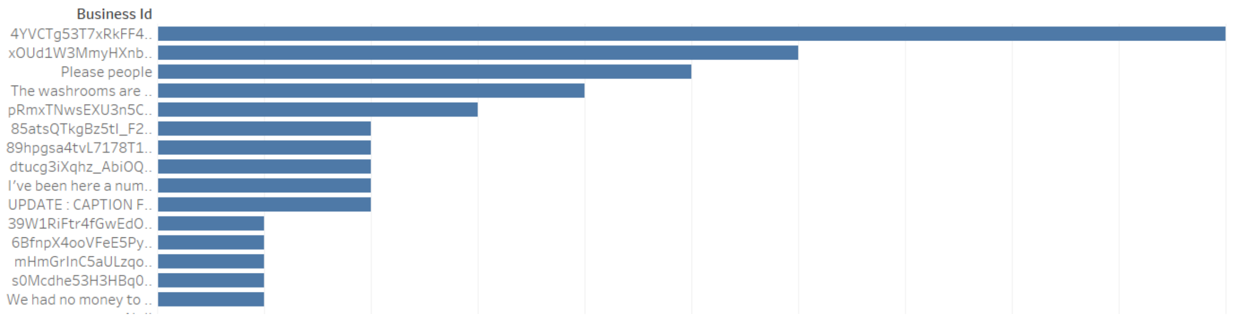
Output file has been stored in the zip folder.

Visualization:

Top 10 Business Categories with Stars.



Bottom 10 business categories with stars:



This shows that the business id 4YVCTg53T7xRkFF4 has the highest star ratings.