Analyzing Airbnb Accommodation Dataset

컴퓨터공학과

2017103948 고다경

1. Dataframe: listing each room_id, host_id with total score in two sorting ways

1. 1) index = (**room_id**, **host_id)**
2. 2) column = **total_score**: **o verall_satisfaction** + **reviews * 0.378**
3. 3) output = **1**. sorted total_score in ascending **2**. sorted total_score in descending

   = sorted_total_score_ascend.csv, sorted_total_score_descend.csv

In [2]:
```
file_name = "./airbnb.csv"
airbnb_rdd = sc.textFile(file_name)

header = airbnb_rdd.first() #extract header
airbnb_rdd = airbnb_rdd.filter(lambda row: row != header)

airbnb_rdd = airbnb_rdd.map(lambda x: x.split(','))
airbnb_rows = airbnb_rdd.map(lambda x: Row(room_id=x[0], host_id=x[1], room_type=x[2], borough=x[3], neighborhood=x[4],
                                           reviews=x[5], overall_satisfaction=x[6], accommodates=x[7], bedrooms=x[8],
                                           price=x[9], minstay=x[10], latitude=x[11], longitude=x[12], last_modified=x[13]))
airbnb_df = sqlContext.createDataFrame(airbnb_rows) #rdd > DF

#overall_satisfaction 빈 값 채우기
updated = udf(lambda x: '3' if x=='' else x, StringType())
airbnb_df = airbnb_df.withColumn('overall_satisfaction', updated(airbnb_df.overall_satisfaction))
#airbnb_df.show()
```

In [3]:
```
#total_score
airbnb_df_1 = airbnb_df.withColumn('total_score',airbnb_df['overall_satisfaction']+airbnb_df['reviews']*0.378)

df1 = airbnb_df_1.select(airbnb_df_1.room_id, airbnb_df_1.host_id, airbnb_df_1.total_score)
df1.registerTempTable('airbnb_df_1')
query1 = 'SELECT room_id, host_id, total_score FROM airbnb_df_1 ORDER BY total_score'
result1 = sqlContext.sql(query1)
result1.show()
result1.coalesce(1).write.format("com.databricks.spark.csv").option("header", "true").save("sorted_total_score_ascend.c

query2 = 'SELECT room_id, host_id, total_score FROM airbnb_df_1 ORDER BY total_score desc'
result2 = sqlContext.sql(query2)
result2.show()
result2.coalesce(1).write.format("com.databricks.spark.csv").option("header", "true").save("sorted_total_score_descend.
```

```
+--------+--------+-----------+
| room_id| host_id|total_score|
+--------+--------+-----------+
|11757251|26873897|      2.134|
|13066225|42820495|        3.0|
| 2513870|12867663|        3.0|
| 6551454|29224952|        3.0|
|10153739|30283594|        3.0|
|12808014|28197086|        3.0|
|12897590|70637430|        3.0|
| 5824015|30228015|        3.0|
|12563549|34226261|        3.0|
|13007354|20696611|        3.0|
|12890140|54064917|        3.0|
|10118379|30283594|        3.0|
|11432300|60063253|        3.0|
|13062783|44620221|        3.0|
|10053369|30283594|        3.0|
|11256475|54064917|        3.0|
| 7052847|10336060|        3.0|
| 9992532|30283594|        3.0|
| 2443944| 4442258|        3.0|
|12254824| 2086352|        3.0|
+--------+--------+-----------+
only showing top 20 rows
```

```
+-------+-------+-----------+
|room_id|host_id|total_score|
+-------+-------+-----------+
|  66288| 324630|     149.53|
| 414419|2027295|    111.474|
|1497879|2776892|    110.718|
|  31796| 119019|    103.536|
| 916123|3637081|     98.622|
| 815639|3637081|     98.622|
|1136972|1480518|     97.232|
|  47521| 119019|      89.55|
|1695275| 324630|     88.294|
|2776143| 324630|     84.892|
| 197972| 965697|     84.002|
| 766700|3637081|     83.502|
|1615033|3637081|     82.746|
|1141522|1407005|     81.234|
|  22354|  85770|     81.234|
|1472520|6608084|     80.222|
|1472481|6608084|     79.844|
| 708802|3648427|     79.844|
|1147871|1407005|     78.966|
|1544702|6608084|      78.71|
+-------+-------+-----------+
only showing top 20 rows
```

2. Dataframe: listing average of factors by grouped neighborhood

1. 1) index = (**neighborhood)**
2. 2) column = **avg of reviews | avg of overall_satisfaction | avg of price | max of reviews | min of reviews | max of price | min of price**
3. 3) output = **1**. sorted neighborhood in ascending

   = sorted_neighborhood_factors.csv

```
In [22]: airbnb_df_2 = airbnb_df.select('reviews','overall_satisfaction','price','neighborhood')
         airbnb_df_2 = airbnb_df_2.groupBy('neighborhood').agg(F.mean('reviews'), F.mean('overall_satisfaction'),
                                                               F.mean('price'), F.max('reviews'), F.min('reviews'),
                                                               F.max('price'), F.min('price'))
         airbnb_df_2 = (airbnb_df_2.withColumnRenamed('avg(reviews)', 'avg of reviews')
                        .withColumnRenamed('avg(overall_satisfaction)', 'avg of overall_satisfaction')
                        .withColumnRenamed('avg(price)', 'avg of price')
                        .withColumnRenamed('max(reviews)', 'max of reviews')
                        .withColumnRenamed('min(reviews)', 'min of reviews')
                        .withColumnRenamed('max(price)', 'max of price')
                        .withColumnRenamed('min(price)', 'min of price'))

         airbnb_df_2 = airbnb_df_2.sort('neighborhood')

         #airbnb_df_2.show()
         airbnb_df_2.coalesce(1).write.format("com.databricks.spark.csv").option("header", "true").save("sorted_neighborhood_fac
```

```
+--------------------+------------------+--------------------------+------------------+--------------+--------------
+------------+------------+
|        neighborhood|    avg of reviews|avg of overall_satisfaction|      avg of price|max of reviews|min of reviews
|max of price|min of price|
+--------------------+------------------+--------------------------+------------------+--------------+--------------
+------------+------------+
|             Allston| 10.31400966183575|          3.710144927536232|100.01932367149759|            92|             0
|        99.0|       100.0|
|            Back Bay|11.296819787985866|         3.9558303886925796|237.51590106007066|             9|             0
|        99.0|       100.0|
|         Bay Village|10.947368421052632|         3.9473684210526314| 254.1578947368421|             8|             0
|        99.0|       120.0|
|         Beacon Hill|              17.8|          4.087804878048781|215.90731707317073|            91|             0
|        99.0|       100.0|
|            Brighton|             13.25|                      3.85|113.95555555555555|            90|             0
|       999.0|        10.0|
|         Charlestown| 24.43076923076923|                       4.2|228.53846153846155|            95|             0
|        99.0|       100.0|
|           Chinatown|11.909090909090908|         3.8545454545454545|254.92727272727274|             9|             0
|        92.0|       100.0|
|          Dorchester|              25.0|                   4.03125|            91.575|            99|             0
|        99.0|       100.0|
|            Downtown|14.671052631578947|         3.9210526315789473|241.18421052631578|            95|             0
|        99.0|       100.0|
|         East Boston| 28.53174603174603|          4.182539682539683|115.48412698412699|            98|             0
|        99.0|       100.0|
|              Fenway| 8.094594594594595|         3.6621621621621623|218.77027027027026|             9|             0
|        99.0|       100.0|
|           Hyde Park| 9.615384615384615|          3.923076923076923| 92.57692307692308|             7|             0
|        87.0|       150.0|
|       Jamaica Plain|22.406349206349205|          4.177777777777778|140.41587301587302|            99|             0
|        99.0|       100.0|
|    Leather District|             9.125|                       4.0|            342.75|             3|             0
|       571.0|       159.0|
|Longwood Medical ...|52.333333333333336|          4.083333333333333| 99.33333333333333|            94|             0
|        84.0|       100.0|
|            Mattapan|              14.4|                     3.675|             78.65|            69|             0
|        95.0|       100.0|
|        Mission Hill|11.346534653465346|          3.633663366336634|132.16831683168317|             9|             0
|        99.0|       100.0|
|           North End|25.401515151515152|          4.140151515151516|183.13636363636363|            99|             0
|        99.0|       100.0|
|          Roslindale|26.696428571428573|          4.491071428571429| 94.60714285714286|             9|             0
|        99.0|       100.0|
|             Roxbury|22.802721088435373|          4.085034013605442| 148.3673469387755|            97|             0
|        99.0|       100.0|
+--------------------+------------------+--------------------------+------------------+--------------+--------------
+------------+------------+
only showing top 20 rows
```

3. Dataframe: listing average of factors by grouped ranged prices

index = ranged prices
output = sort_ranged_price.csv

In [40]:
```
airbnb_df_3 = airbnb_df.select('price','accommodates','bedrooms','reviews','neighborhood')

grouped = udf(lambda x:
              '0-100' if x>='0' and x<'100'
              else ('100-200' if x>='100' and x<'200'
                   else('200-300' if x>='200' and x<'300'
                       else('300-400' if x>='300' and x<'400'
                           else('400-500' if x>='400' and x<'500'
                               else('500-1000' if x>='500' and x<'1000'
                                   else('1000-5000' if x>='1000' and x<'5000'
                                       else '')))))),StringType())

update_neighbor = udf(lambda x: ",".join(x))

airbnb_df_3 = airbnb_df_3.withColumn('PRICE',grouped(airbnb_df_3.price))
airbnb_df_3 = airbnb_df_3.groupBy('PRICE').agg(F.mean('accommodates').alias('accommodates average'),
                                               F.percentile_approx('accommodates',0.5).alias('accommodates median'),
                                               F.mean('bedrooms').alias('bedrooms average'),
                                               F.percentile_approx('bedrooms',0.5).alias('bedrooms median'),
                                               F.mean('reviews').alias('reviews average'),
                                               F.percentile_approx('reviews',0.5).alias('reviews median'),
                                               F.collect_list('neighborhood').alias('neighbor_list'),
                                               F.count('accommodates').alias('length'))
airbnb_df_3 = airbnb_df_3.filter(airbnb_df_3.PRICE != '')
airbnb_df_3 = airbnb_df_3.withColumn('neighbor_list',update_neighbor(airbnb_df_3.neighbor_list))
airbnb_df_3 = airbnb_df_3.orderBy(F.split(airbnb_df_3.PRICE, '-').getItem(0).cast(IntegerType()))

airbnb_df_3.show()
airbnb_df_3.coalesce(1).write.format("com.databricks.spark.csv").option("header", "true").save("sorted_ranged_price.csv
```

```
+---------+--------------------+------------------+------------------+--------------+------------------+-----------
---+--------------------+------+
|    PRICE|accommodates average|accommodates median|  bedrooms average|bedrooms median|   reviews average|reviews med
ian|       neighbor_list|length|
+---------+--------------------+------------------+------------------+--------------+------------------+-----------
---+--------------------+------+
|    0-100|                 6.0|               6.0|               3.0|           3.0|               0.0|
0.0|            Brighton|     1|
|  100-200|              2.9832|               2.0| 1.107457898957498|           1.0|           16.9688|
5.0|Jamaica Plain,Bea...|  1250|
|  200-300|   3.792517006802721|               4.0|1.5197934595524958|           1.0|13.615646258503402|
4.0|Back Bay,Jamaica ...|   588|
|  300-400|   4.011627906976744|               4.0|1.7713178294573644|           2.0|10.666666666666666|
3.0|South End,South B...|   258|
|  400-500|  2.8450704225352115|               2.0|1.3380281690140845|           1.0| 7.657276995305164|
1.0|Allston,Allston,A...|   213|
|1000-5000|   5.684210526315789|               5.0|2.5789473684210527|           2.0| 9.578947368421053|
4.0|Jamaica Plain,Bea...|    19|
+---------+--------------------+------------------+------------------+--------------+------------------+-----------
---+--------------------+------+
```