CS 839 Spring 2018, Project Stage 4

Team members:
Arpith Neelavara (neelavara@wisc.edu)
Bhargav Tangirala (btangirala@wisc.edu)
Aribhit Mishra (amishra28@wisc.edu)

PART A - Data Merging

1. How did you combine the two tables A and B to obtain E? Did you add any other table? When you did the combination, did you run into any issues? Discuss the combination process in detail, e.g., when you merge tuples, what are the merging functions (such as to merge two age values, always select the age value from the tuple from Table A, unless this value is missing in which case we select the value from the tuple in Table B)

After doing the matching in the previous stage, we got a set of tuple pairs which match (1624 tuple pairs). In this stage we take this result and filter out the tuples that were predicted right (786 tuple pairs) and proceed to do the merging to obtain the final list of restaurants i.e Table E. We used a variety of merging functions depending on the columns which merge. Below are the details:

Name of the restaurant: We picked the name whose length is the longest among the two.

Address: We picked the address whose length is the longest among the two.

Cuisines: We have considered to take the column that lists the highest number of cuisines because we observed that the longer list contain most of the cuisines listed in the smaller one. All of them are separated by a semicolon.

Take Out: If any one if the tuples say that the take is Yes then we have the Take out as Yes in the final lis. Same is the case with No.

Phone: The phone numbers in one of the data set dont have srae code, so we decided to take the data from other set, i.e compared the length of the two and got the one with larger length.

Below is the common thing which was done for the entities **Saturday Opening time, Saturday Closing time, Sunday Opening time, Sunday Closing time**:

If there missing value (nan) in the opening time in one of the tuple pairs we pick the opening time from the other. Same is the case with closing time rule. If both the times have data in them, then we pick the one with shorter length as we found that times with longer length have some inconsistency in them and inorder to get the correct value we picked the smaller ones. (Inconsistency like multiple opening and closing times)

This was all about the rules we used for combining the records. We did not add any other table for combining. We didn't run into any issues as the combining step was relatively simple in our case. Table E had 786 tuples in total.

2. Statistics on Table E: specifically, what is the schema of Table E, how many tuples are in Table E? Give at least four sample tuples from Table E.

Schema of Table E: (Name, Address, Cuisines, Take_Out, Phone, Saturday Opening time, Saturday Closing time, Sunday Opening time, Sunday Closing time)

Table E has 786 tuples in total.

Below are the sample tuples from Table E:

Name	Address	Cuisines	Take_ Out	Phone	Saturday Opening time	Saturday Closing time	Sunday Opening time	Sunday Closing time
Shanghai asian manor	21 Mott St New York City, Ny 10013-5032	Chinese; Vegetarian Friendly; Asian	Yes	+1 212-766-6 311	11:00 AM	10:00 PM	11:00 AM	09:00 PM
Shake shack	366 Columbus Ave New York City, Ny 10024-5109	American; Fast Food	Yes	+1 646-747-8 770	10:30 AM	11:00 PM	10:30 AM	11:00 PM
Blue maiz	606 8Th Ave New York City, Ny 10018-3008	Fast Food; Mexican; Latin	Yes	+1 212-704-2 106	11:00 AM	10:00 PM	11:00 AM	09:00 PM
Barn joo	35 34 W 35Th St New York City, Ny 10001-2256	Korean; Gastropub; Asian	Yes	+1 212-564-4 430	12:00 PM	02:00 AM	12:00 PM	11:00 PM
Southgate bar & restaurant	154 Central Park S New York City, Ny 10019-1510	Gluten Free Options; American	No	+1 212-484-5 120	07:00 AM	10:30 PM	07:00 AM	10:30 PM

3. Code for for merging tables.

As suggested added the code at the end of this document.

PART B - Data Analysis

1. What was the data analysis task that you wanted to do? For that task, describe in detail the data analysis process that you went through.

We tried to analyse using OLAP style analysis. We have examined the data in various ways, below is the list: (Graph plotted and the rows returned after extracting useful insights can be found in the file named **Data Analysis.xlsx**)

- **Analysis 1**: We extracted all the type of food that are available in the restaurants in New York City and the count of number of restaurants that offer this food type. We took out top 10 cuisines that were found from this extraction.
- **Analysis 2**: Also, we got top 10 food types that can be found in restaurants in NYC.

- **Analysis 3**: We have also extracted the number of restaurants that we can find in each zip code. This helps in identifying an area were we can have a lot of options to choose restaurants (Restaurant Density). We have color coded this data over NYC map.
- **Analysis 4**: We have also shown the percentage of restaurants in NYC which provide an option of Take Out and which do not provide Take Out option.
- **Analysis 5**: We also tried to get a rough idea as to how much percentage of the total restaurants are chain restaurants in NYC.

2.Give any accuracy numbers that you have obtained (such as precision and recall for your classification scheme).
We didn't have any such numbers.

3.What did you learn/conclude from your data analysis? Were there any problems with the analysis process and with the data?
Below are the conclusions from each analysis done:
Analysis 1: The top 10 cuisines which we discovered were: American, Italian, Asian, Mediterranean, French, Japanese, European, Latin, Spanish and Chinese.
Analysis 2: The top 10 food types which we discovered were: Seafood, Pizza, Fast Food, Sushi, ,Steakhouse, Coffee & Tea, Barbecue, Soups, Sandwiches and Desserts.
Analysis 3: We found out that zip codes 10019, 10003 and 10014 have high density of restaurants.
Analysis 4: We found that around 70% of the restaurants in NYC have Take out option.
Analysis 5: We found that only 3% of the restaurants in NYC are chain restaurants.

We didn't face any problems while analysing the data.

4.If you have more time, what would you propose you can do next?
If we had more time we could probably get more insights as to which type of cuisine is more available in an area so that we can hope to have choices of areas to choose from when we are interested in only a particular type of cuisine.

****

Part A: Append the code of the Python script (that merges the tables) to the end of this pdf file.

Merging Name:
```
def name_merger(row) :
    return row['ltable_Name'][:1].upper()+row['ltable_Name'][1:] if len(row['ltable_Name']) >= len(row['rtable_Name'])\
    else row['rtable_Name'][:1].upper()+row['rtable_Name'][1:]
```

Merging Address:
```
def address_merger(row) :
    return row['ltable_Address'].title() if len(row['ltable_Address']) >= len(row['rtable_Address']) \
    else row['rtable_Address'].title()
```

Merging Cuisines:

```
def cuisine_merger(row):
    result = set()
    union1 = row['ltable_Cuisines'].split(';')
    union1 = [x.lower() for x in union1]
    union2 = row['rtable_Cuisines'].split(';')
    union2 = [x.lower() for x in union2]
    if len(union1)>=len(union2):
        result.update(union1)
    else:
        result.update(union2)
    return "; ".join([x.title() for x in list(result)])
```

Merging Take_Out:

```
def takeout_merger(row) :
    if row['ltable_Take Out'] == 'Yes' or row['rtable_Take Out'] == 'Yes' :
        return 'Yes'
    else :
        return 'No'
```

Merging Phone:

```
def phone_merger(row) :
    return row['ltable_Phone'] if len(row['ltable_Phone']) >= len(row['rtable_Phone']) \
    else row['rtable_Phone']
```

Merging Saturday Opening time:

```
def sat_open_merger(row) :
    if row['ltable_Saturday Opening time'].lower() == 'nan' :
        return row['rtable_Saturday Opening time']
    if row['rtable_Saturday Opening time'].lower() == 'nan' :
        return row['ltable_Saturday Opening time']
    if row['rtable_Saturday Opening time'].lower() == 'closed' or row['ltable_Saturday Opening
time'].lower() == 'closed':
        return 'Closed'
    return row['ltable_Saturday Opening time'] if len(row['ltable_Saturday Opening time']) <=
len(row['rtable_Saturday Opening time']) \
    else row['rtable_Saturday Opening time']
```

Merging Saturday Closing time:

```
def sat_close_merger(row) :
    if row['ltable_Saturday Closing time'].lower() == 'nan' :
        return row['rtable_Saturday Closing time']
    if row['rtable_Saturday Closing time'].lower() == 'nan' :
        return row['ltable_Saturday Closing time']
```

```
    if row['rtable_Saturday Closing time'].lower() == 'closed' or row['ltable_Saturday Closing
time'].lower() == 'closed':
        return 'Closed'
    return row['ltable_Saturday Closing time'] if len(row['ltable_Saturday Closing time']) <=
len(row['rtable_Saturday Closing time']) \
    else row['rtable_Saturday Closing time']
```

Merging Sunday Opening time:

```
def sun_open_merger(row) :
    if row['ltable_Sunday Opening time'].lower() == 'nan' :
        return row['rtable_Sunday Opening time']
    if row['rtable_Sunday Opening time'].lower() == 'nan' :
        return row['ltable_Sunday Opening time']
    if row['rtable_Sunday Opening time'].lower() == 'closed' or row['ltable_Sunday Opening
time'].lower() == 'closed':
        return 'Closed'
    return row['ltable_Sunday Opening time'] if len(row['ltable_Sunday Opening time']) <=
len(row['rtable_Sunday Opening time']) \
    else row['rtable_Sunday Opening time']
```

Merging Sunday Closing time:

```
def sun_close_merger(row):
    if row['ltable_Sunday Closing time'].lower() == 'nan' :
        return row['rtable_Sunday Closing time']
    if row['rtable_Sunday Closing time'].lower() == 'nan' :
        return row['ltable_Sunday Closing time']
    if row['rtable_Sunday Closing time'].lower() == 'closed' or row['ltable_Sunday Closing
time'].lower() == 'closed':
        return 'Closed'
    return row['ltable_Sunday Closing time'] if len(row['ltable_Sunday Closing time']) <=
len(row['rtable_Sunday Closing time']) \
    else row['rtable_Sunday Closing time']
```