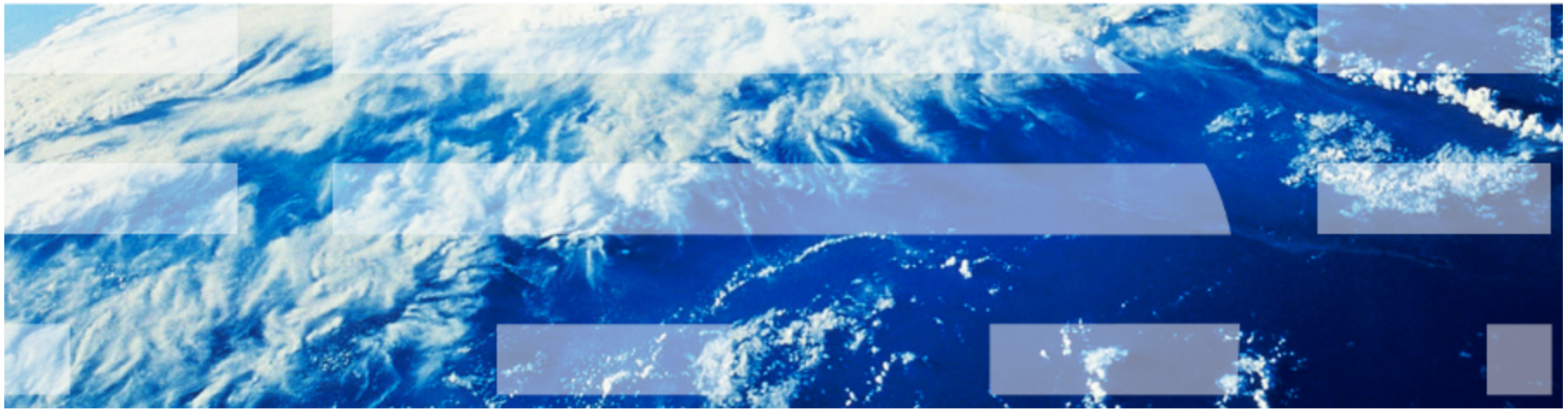
E6893 Big Data Analytics:

Yelp-er: Analyzing Yelp Data

Team Members: Naman Jain, Natasha Kenkre, Rhea Goel, Sanket Jain



December 11, 2014

Overview

Columbia University

I. Query-based HeatMap

II. Semantic Analysis & Topic Modeling

III. Gamification

I. User Perspective: Query-based HeatMap

Columbia University

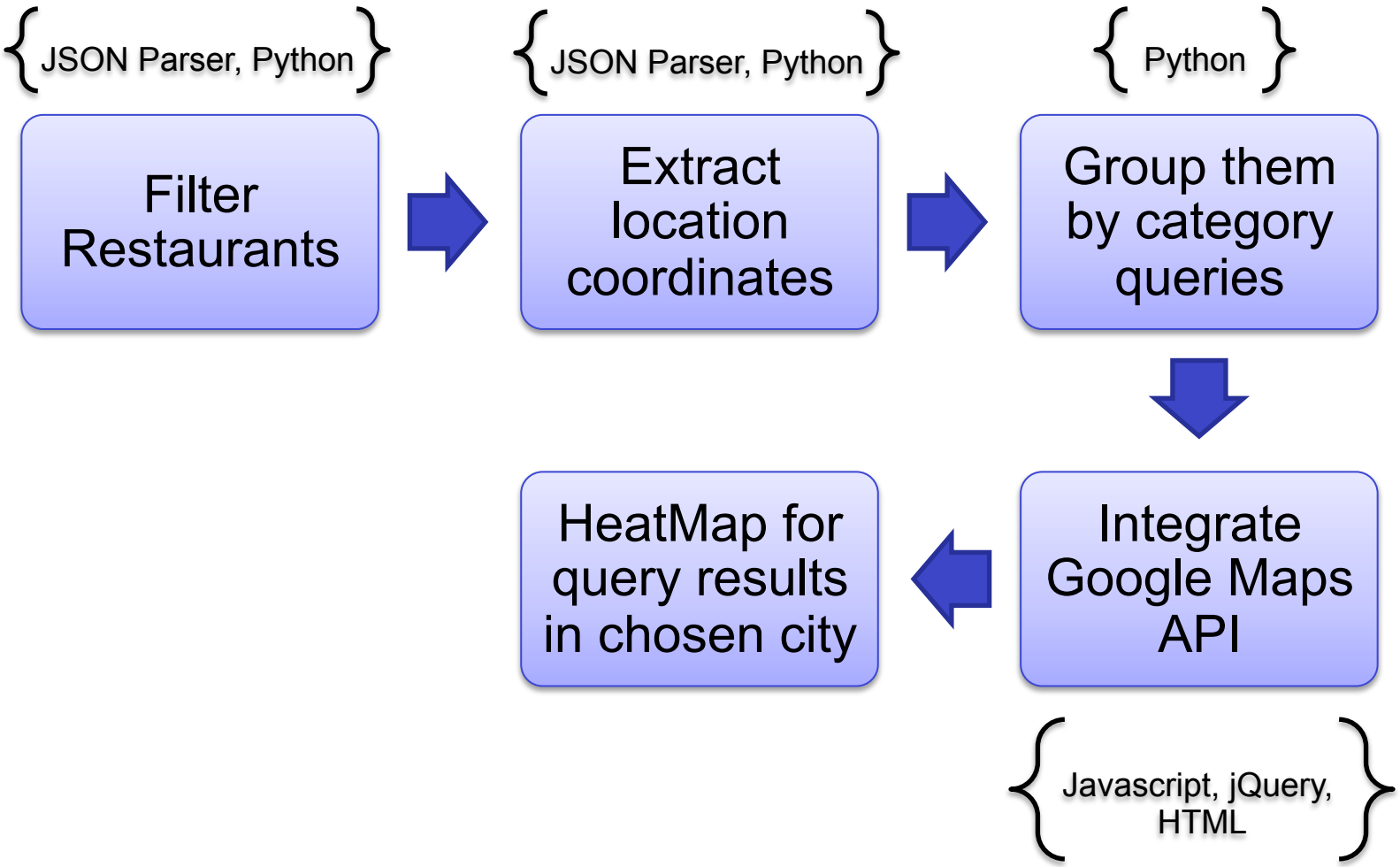
What?

- Display heatmap for a search query on google maps
- Data visualization to make it more user friendly

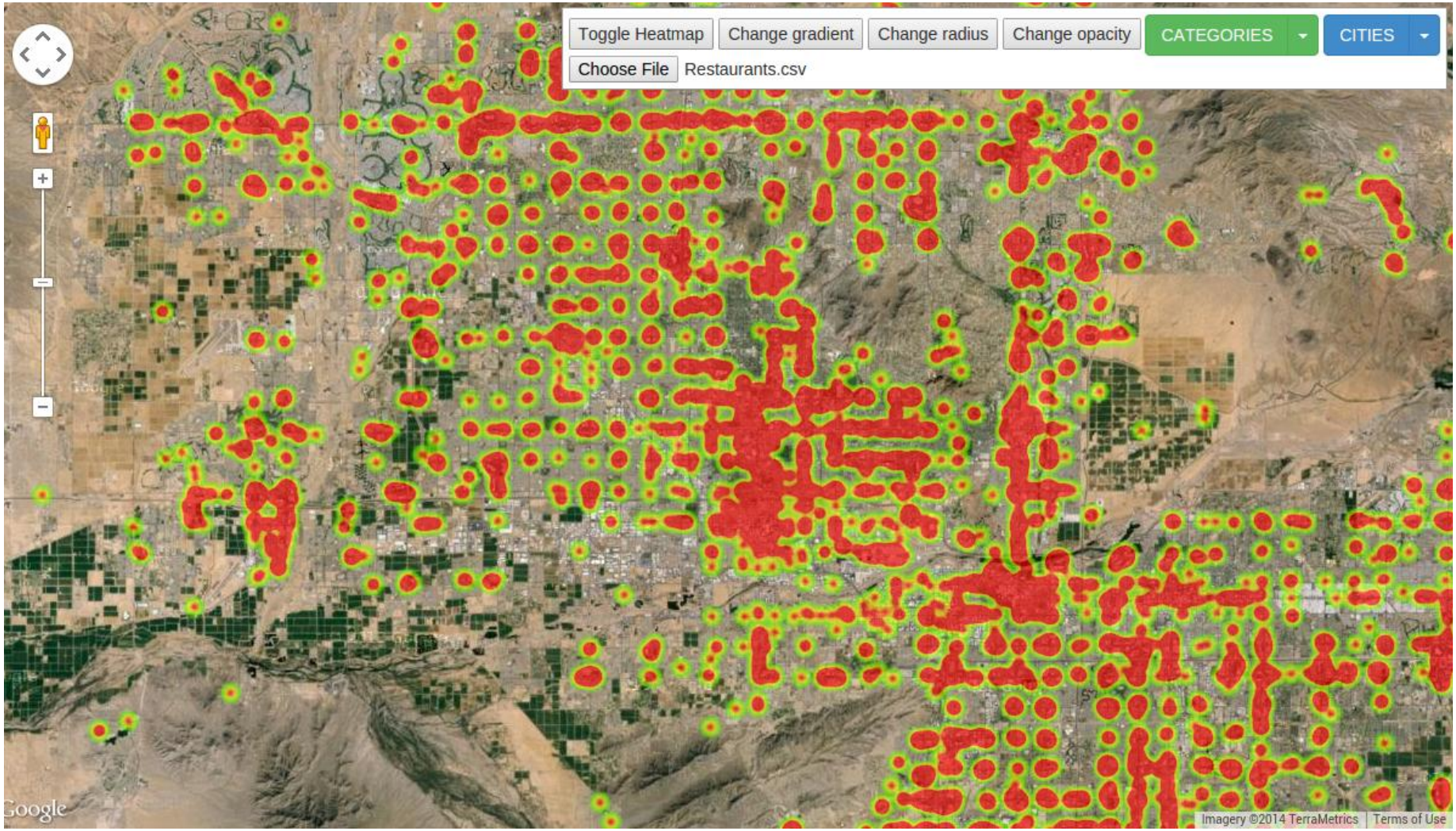
Commercial Value

- Helps user identify hubs for his/her interests
- Improves the usability, and the look-and-feel of the interface

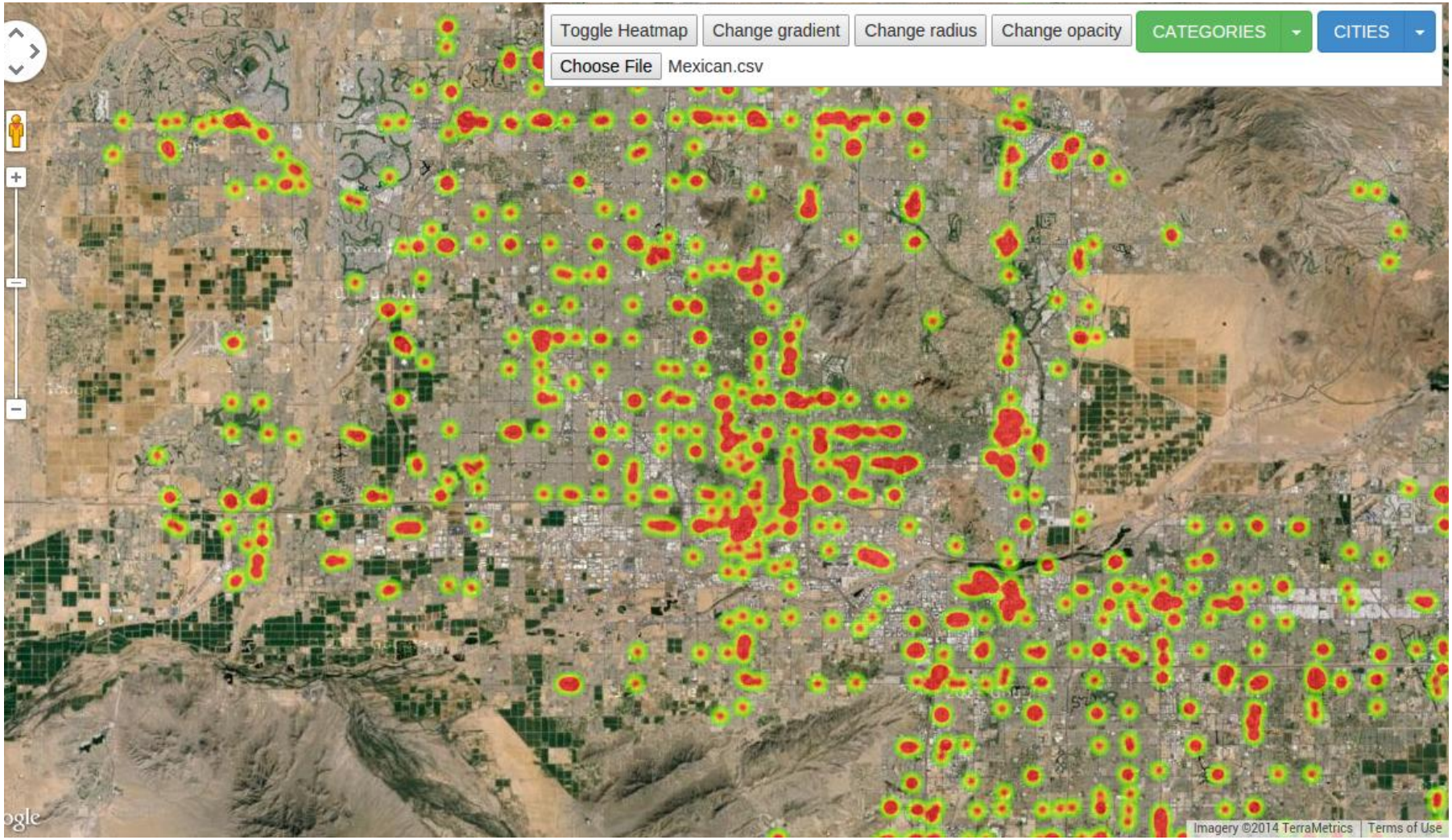
I. Query-based HeatMap: Process Flow



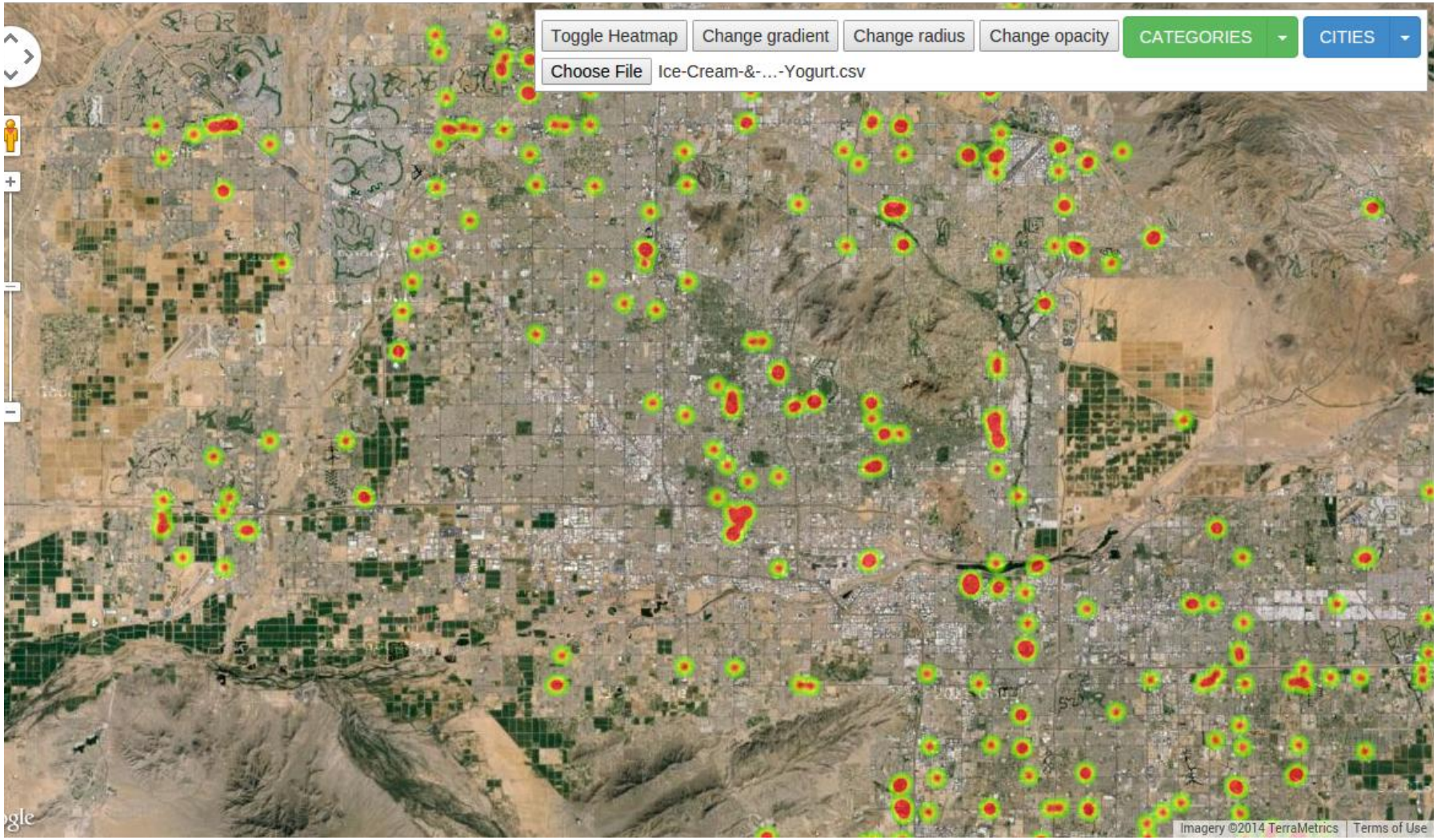
HeatMap for “Restaurants” in Phoenix



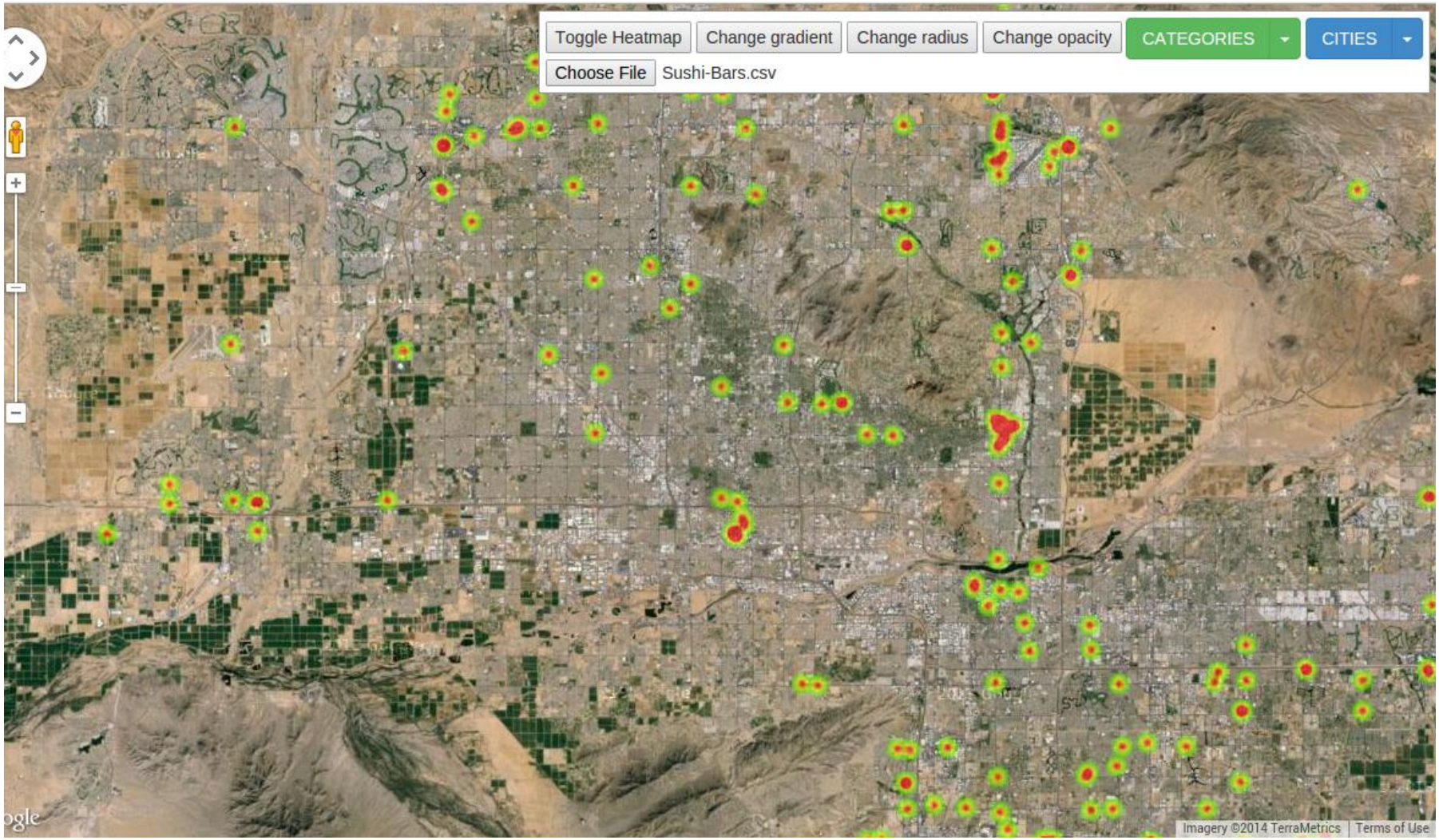
HeatMap for “Mexican” in Phoenix



HeatMap for “Ice Cream” in Phoenix



HeatMap for “Sushi” in Phoenix



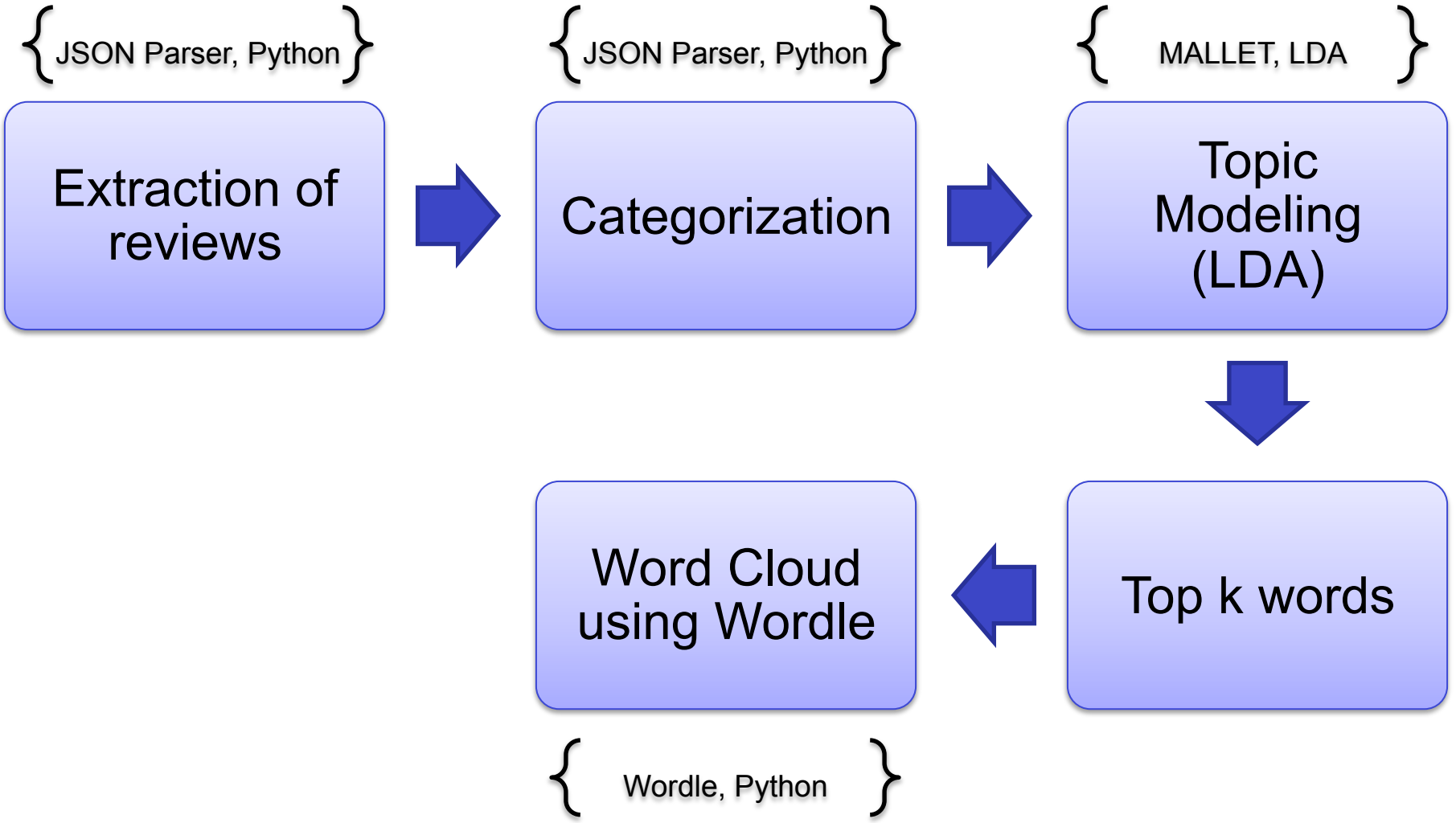
What?

- Identifying topics in review text
- Find most-talked-about topics for a business

Commercial Value

- Can help businesses figure out their strengths and weaknesses
- Monetization based on insights about what attracts the users

II. Semantic Analysis



Latent Dirichlet Allocation (LDA)

Columbia University

Algorithm 1 Batch variational Bayes for LDA

Initialize $\boldsymbol{\lambda}$ randomly.
while relative improvement in $\mathcal{L}(\boldsymbol{w}, \boldsymbol{\phi}, \boldsymbol{\gamma}, \boldsymbol{\lambda}) > 0.00001$ **do**
  E step:
  for $d = 1$ to D **do**
    Initialize $\gamma_{dk} = 1$. (The constant 1 is arbitrary.)
    repeat
      Set $\phi_{dwk} \propto \exp\{\mathbb{E}_q[\log \theta_{dk}] + \mathbb{E}_q[\log \beta_{kw}]\}$
      Set $\gamma_{dk} = \alpha + \sum_w \phi_{dwk} n_{dw}$
    until $\frac{1}{K} \sum_k |\text{change in} \gamma_{dk}| < 0.00001$
  end for
  M step:
  Set $\lambda_{kw} = \eta + \sum_d n_{dw} \phi_{dwk}$
end while

MALLET

- Java-based package for statistical NLP, document classification, clustering, topic modeling, information extraction, and other machine learning applications to text
- MALLET topic modeling toolkit contains efficient, sampling-based implementations of LDA, Pachinko Allocation and Hierarchical LDA
- It helped us identify topics, along with topic strength – and relevance of each word in all topics

Word Cloud for Topic: Nightlife/Bar

Word Cloud for Topic: Positive Reviews

Word Cloud for Topic: Negative Reviews

Word Cloud for Topic: Dishes/Food items

III. Yelp Perspective: Gamification

Columbia University

What?

- Analyze user's activity (review count, fans, votes, compliments, friends, yelping_since)
- Assign tags like 'Popular', 'Social', 'Newbie', 'Lazybones', 'Super Active', 'Dependable'
- Helps encourage activity and users' self contributions

Commercial Value

- Helps Yelp increase their customer base, market value, brand name, user loyalty
- Drive better customer retention and lifetime value

Code

user

```
{
    'type': 'user',
    'user_id': (encrypted user id),
    'name': (first name),
    'review_count': (review count),
    'average_stars': (floating point average, like 4.31),
    'votes': {(vote type): (count)},
    'friends': [(friend user_ids)],
    'elite': [(years_elite)],
    'yelping_since': (date, formatted like '2012-03'),
    'compliments': {
        (compliment_type): (num_compliments_of_this_type),
        ...
    },
    'fans': (num_fans),
}
```

```
75      yearsItHasBeen = 2014 - year
76
77      #if the user is only 6 months old ->newbie
78      if yearsItHasBeen==0:
79          monthsItHasBeen = 12 - month
80
81          if monthsItHasBeen<6:
82              if 'tags' not in temp:
83                  temp['tags'] = []
84              temp.get('tags').append('Newbie')
85
86
87      #number of reviews is low and yelping since is old -> lazybones
88      if reviews<(yearsItHasBeen):
89          if 'tags' not in temp:
90              temp['tags'] = []
91          temp.get('tags').append('Lazybones')
92
93      #number of reviews is more than 1 per year -> super active
94      if reviews>24*(yearsItHasBeen):
95          if 'tags' not in temp:
96              temp['tags'] = []
97          temp.get('tags').append('Super Active')
98
99      print temp.get('tags')
100
101     #number of votes is high -> dependable
102     if votes>maxVotes/5:
103         if 'tags' not in temp:
104             temp['tags'] = []
105         temp.get('tags').append('Dependable')
106
107     #high number of friends -> social
108     if friends>maxFriends/5:
109         if 'tags' not in temp:
110             temp['tags'] = []
111         temp.get('tags').append('Social')
112
113     allUserInfo[user] = temp
114     print temp.get('tags')
115
116
```

```
None
['Lazybones']
['Lazybones']
None
None
None
None
['Popular', 'Super Active']
['Popular', 'Super Active', 'Dependable', 'Social']
None
None
['Lazybones']
['Lazybones']
['Popular', 'Super Active']
['Popular', 'Super Active']
None
None
['Popular']
['Popular']
['Popular', 'Lazybones']
['Popular', 'Lazybones']
None
None
ne
None
['Lazybones']
['Lazybones']
['Lazybones']
['Lazybones']
['Popular']
['Popular']
['Popular']
['Popular']
None
None
None
None
['Popular', 'Super Active']
['Popular', 'Super Active']
['Lazybones']
['Lazybones']
None
None
['Popular']
['Popular']
None
None
None
```

Thank You