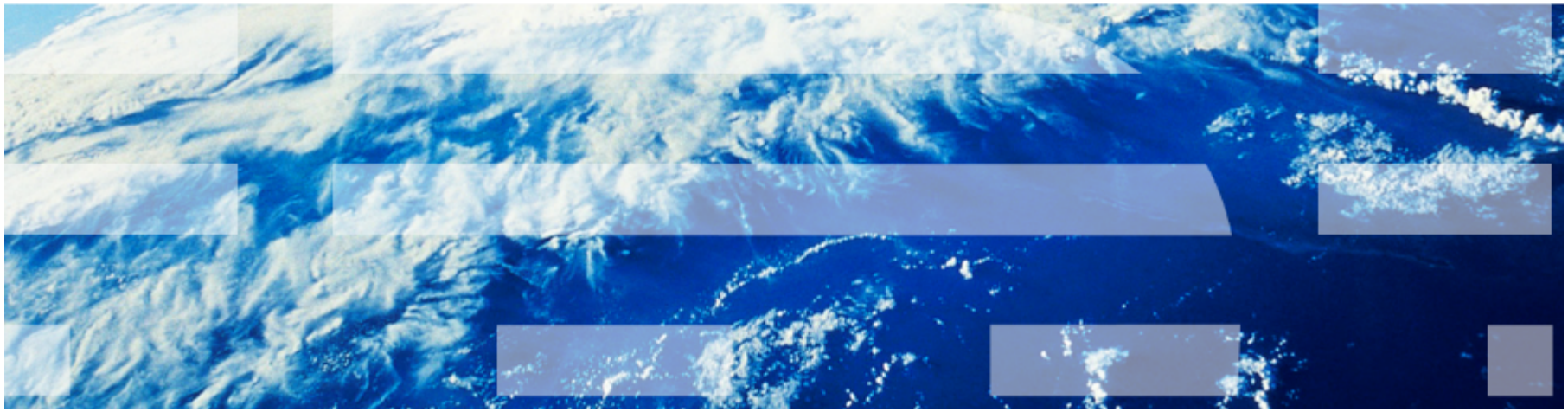
E6893 Big Data Analytics:

Analysis and Visualization of NYC Taxi Trip Data

Xianglu Kong, Junfei Shen, Guochen Jing



Motivation:

- **Visualize NYC taxi trips to create an intuitive picture of pick up and drop off locations on New Year's day.**

- **By doing clustering over the data, we seek to identify most popular pick up places in NYC last Jan. 1st. Hence make recommendations to taxi drivers about where might be the most active places on the coming New Year's day.**

NYC Taxi Trip Data on Jan. 1st 2015

- The data we used is downloaded from NYC taxi and limousine commissioner website. http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml. We only used data of yellow cabs because green cabs are under certain regulation thus do not reflect the demand entirely.

- In order to get a clearer image of demand of taxi at various times we made a histogram indicating number of pick-ups and drop-offs versus time (in one hr increment).

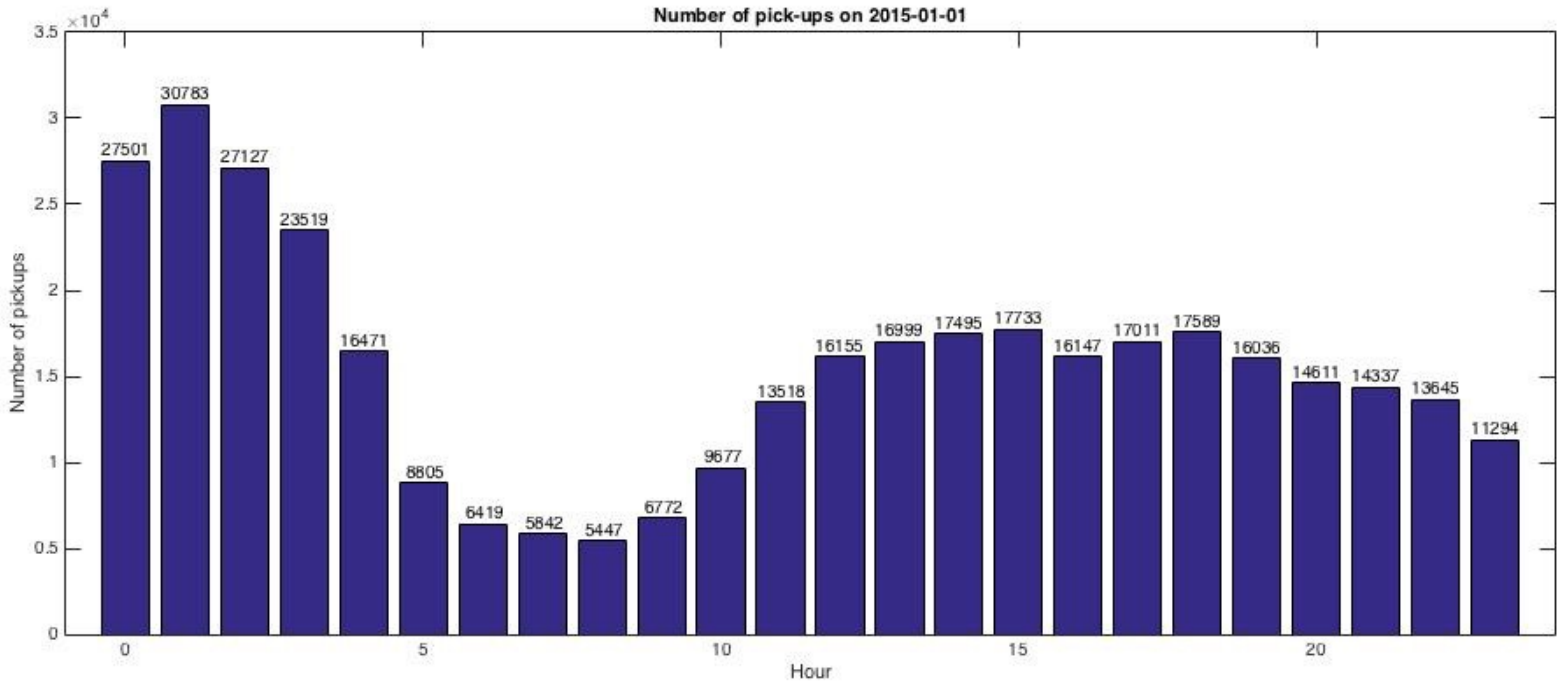
NYC Taxi Trip Data on Jan. 1st 2015

- **To achieve this goal we use pig to pick out the columns we wanted a.k.a pick-up drop-off time from the data and counted the total number of pick-ups and drop-offs at each hour.**

```
# getLocInfo.py
from org.apache.pig.scripting import *


P = Pig.compile("""
tripdata = LOAD '/PigSource/dayRecord/part-r-00000' USING PigStorage('\t') AS (tpep_pickup_datetime,
    tpep_dropoff_datetime,pickup_latitude,pickup_longitude,dropoff_latitude,dropoff_longitude);

hourRecord = FILTER tripdata BY (tpep_dropoff_datetime matches '$time');
hourgroup = GROUP hourRecord ALL;

count = FOREACH hourgroup GENERATE COUNT(hourRecord);
STORE count INTO '$outpath';
""")

params = {}

for i in range(24):
    params["time"] = "2015-01-01 " + ("%02d" % i) + ".*"
    params["outpath"] = "/dropOffCountOutput/" + ("%02d" % i)
    print(params["time"])
    print(params["outpath"])
    bound = P.bind(params)
    stats = bound.runSingle()
    if not stats.isSuccessful():
        raise 'failed'

```

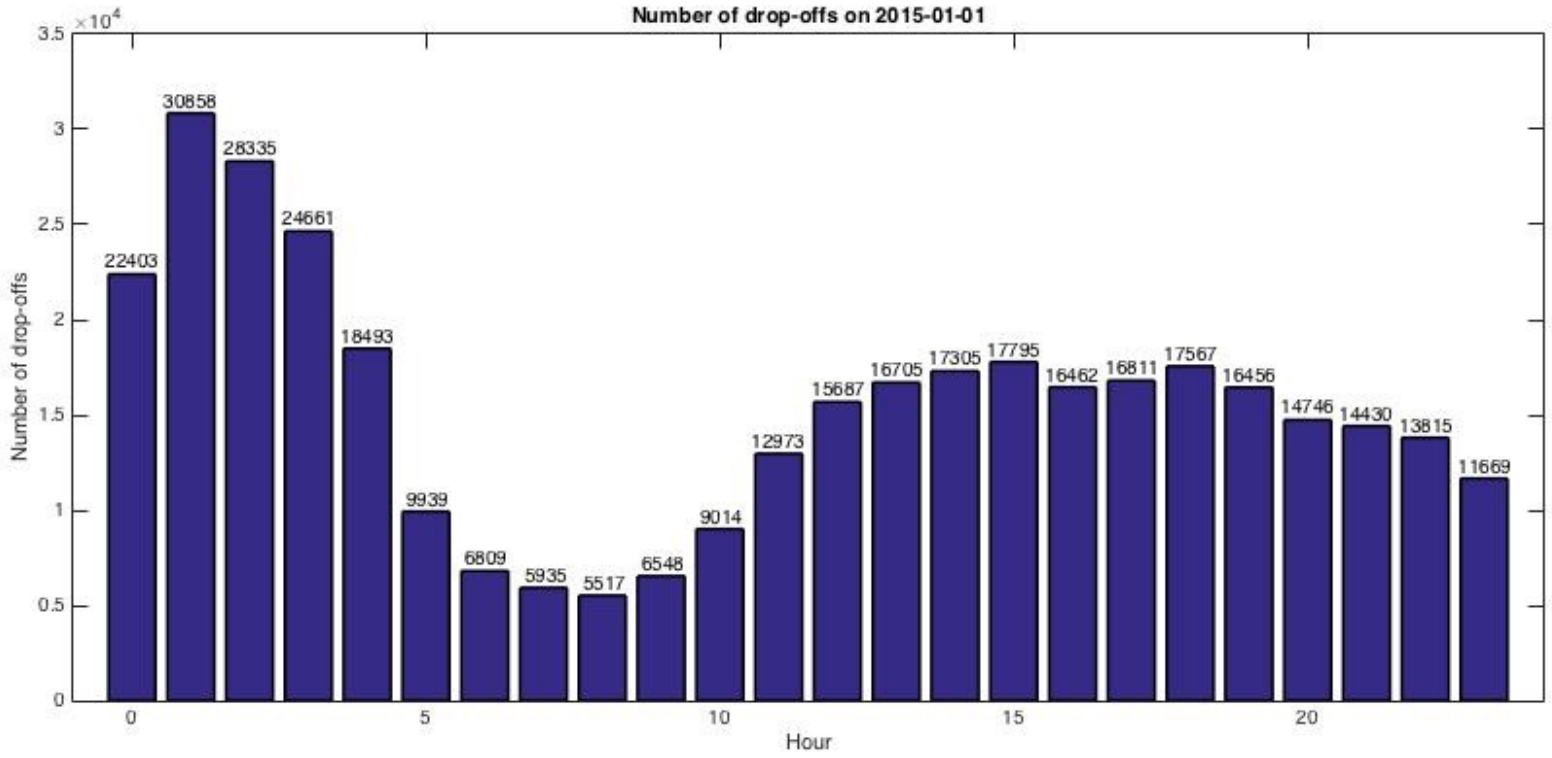
NYC Taxi Trip Data on Jan. 1st 2015

- Number of Pick-ups VS Hour



NYC Taxi Trip Data on Jan. 1st 2015

- Number of Drop-offs VS Hour



Visualization

Visualization

JavaScript API:
mapbox.js
Leaflet.js

var polylineArray = [];

var polylines = L.layerGroup(polylineArray);

polylines.addTo(map);

next(); //3s

```
function next() {
    // ......

    map.removeLayer(polylines);
    polylineArray = [];
    pairs = paths[hour];
    for (var i = 0; i < pairs.length; i++) {
        var generator = new arc.GreatCircle(
                obj(pairs[i][0]),
                obj(pairs[i][1])
                );
        var line = generator.Arc(100, { offset: 10 });
        var newLine = L.polyline(line.geometries[0].coords.map(function(c) {
            return c.reverse();
        }), {
            color: '#ffe766',
            weight: 0.5,
            opacity: 0.7
        });
        polylineArray.push(newLine);
    }
    polylines = L.layerGroup(polylineArray);
    polylines.addTo(map);

    for (var i = 0; i < polylineArray.length; i++) {
        var newLine = polylineArray[i];
        var totalLength = newLine._path.getTotalLength();
        newLine._path.classList.add('path-start');
        newLine._path.style.strokeDashoffset = totalLength;
        newLine._path.style.strokeDasharray = totalLength;
        setTimeout((function(path) {
            return function() {
                path.style.strokeDashoffset = 0;
            };
        })(newLine._path), i * 1);
    }
    if (++hour >= paths.length) hour = 0;
}
```

http://htmlpreview.github.io/?https://github.com/Nadia-mint/TaxiTripNYC_Visualization_2015/blob/master/TaxiTripPath.html

K-Means Clustering

- The clustering technique we used is k-means. First, we pin all data onto a map using their corresponding latitude and longitude value. Then, we run k-means algorithm to obtain the cluster centroids. We set k to 10 for convenience.

- We will break up the clustering scheme into 3 steps.
  - Load Data
  - Transform Data
  - Cluster

K-Means Clustering (Load Data)

```
from pyspark.sql import SQLContext
from pyspark.sql.types import *

fileName = "yellow_tripdata_2015-01.csv"

sqlContext = SQLContext(sc)
taxiFile = sc.textFile(fileName)
header = taxiFile.first() # First line: variable names

pickup_fields = [StructField(field_name, StringType(), True) for field_name in header.split(',') if "pickup" in field_name] # find relevant variable names
pickup_fields[0].dataType = TimestampType() # transform data from original strings to proper types
pickup_fields[1].dataType = FloatType()
pickup_fields[2].dataType = FloatType()
pickup_fields[0].name = "pickup_datetime" # rename a variable

pickup_schema = StructType(pickup_fields) # a "template" for data frame

taxiHeader = taxiFile.filter(lambda l: "ID"in l) # find the header row
taxiNoHeader = taxiFile.subtract(taxiHeader) # drop the first row, so we have only data left
```

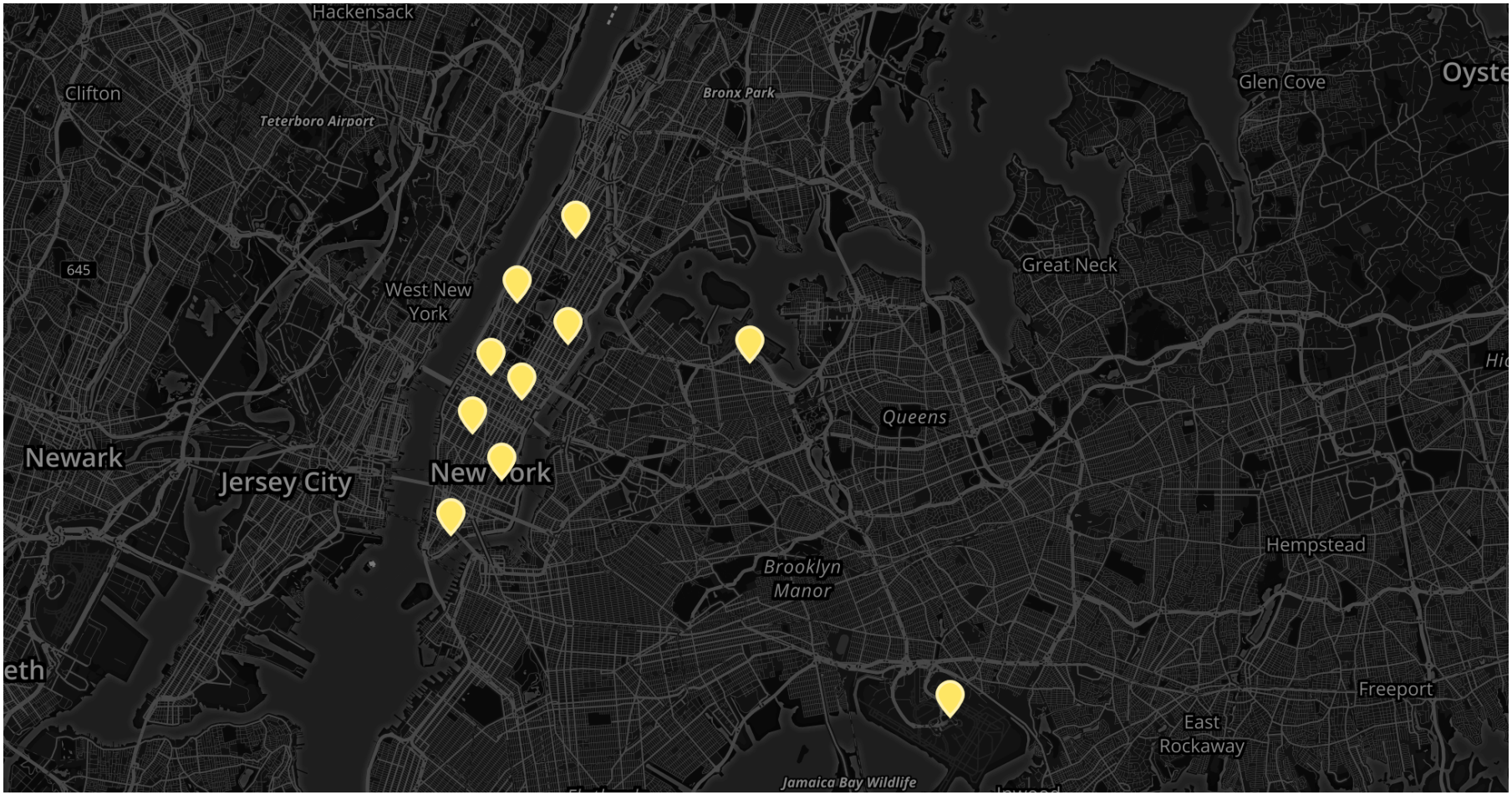
K-Means Clustering (Transform Data)

```
from datetime import *
from dateutil.parser import parse

# I only want data for trips that occured during a given time period
start_datetime = parse("1/1/2015 9:00:00")
end_datetime = parse("1/1/2015 12:00:00") # parse() converts string into datetime objects

pickup_temp = taxiNoHeader.map(lambda k: k.split(",")).map(lambda p: (parse(p[1]), float(p[5]), float(p[6])))

pickup_df = sqlContext.createDataFrame(pickup_temp, pickup_schema) # 12748986 records

pickup_df = pickup_df.filter(pickup_df.pickup_datetime >= start_datetime)
pickup_df = pickup_df.filter(pickup_df.pickup_datetime <= end_datetime)
# there are missing longitude and latitude data filled with 0 and typos
pickup_df = pickup_df.filter(pickup_df.pickup_longitude > -75)
pickup_df = pickup_df.filter(pickup_df.pickup_longitude < -72)
```

K-Means Clustering (Cluster)

```
from numpy import array
from pyspark.mllib.clustering import KMeans, KMeansModel

pickup_array = pickup_df.map(lambda x: array([float(x[1]), float(x[2])]))
pickup_model = KMeans.train(pickup_array, 10, maxIterations=10, runs=4, initializationMode="random")
pickup_centers = pickup_model.clusterCenters
pickup_centers

[array([-73.98227551,  40.72726685]),
 array([-73.94894565,  40.8091506 ]),
 array([-73.97328617,  40.75436116]),
 array([-73.98660181,  40.76263573]),
 array([-73.87124998,  40.76696224]),
 array([-73.97492762,  40.78705872]),
 array([-73.95253233,  40.77328378]),
 array([-73.78222692,  40.64736145]),
 array([-73.99468622,  40.74292497]),
 array([-74.0047403 ,  40.70852686])]
```

K-Means Clustering (Cluster)



http://htmlpreview.github.io/?https://github.com/Nadia-mint/TaxiTripNYC_Visualization_2015/blob/master/pickupCluster.html

Questions?