Big Data Applications Symposium - Fall 2017

Project Name: Travel Duration Prediction of Taxi and Bike Trips in NYC

Team:

Weiqiang Li; Liheng Gong

Abstract: How would weather condition affect commute time and passenger flow in New York City? To study this problem, we collected the weather data, taxi trip data, and CityBike data in New York City and utilized machine learning algorithms to reveal the relationship between weather condition and commute time in New York City. We use three machine learning algorithms - Linear Regression, Decision Tree and Random Forest to find that weather condition does affect trip time in New York City; the performance of these three algorithms are also compared.

Motivation

Who are the users of this application?

Commuters, city transportation administration and taxi service providers

Who will benefit from this application?

Commuters, city transportation administration and taxi service providers

Why is this application important?

Use big data technology to improve commute experience

Remediation

What actuation(s) or remediation actions are performed by the application?

Predict commute time based on historical data; commuters can make better plan based on our prediction.

Travel Duration Prediction of Taxi and Bike Trips in NYC

Data Sources

Name: New York City weather data

Description: The NOAA Online Weather Data contains the temperature and climate information collected from NYC Central Park station.

Size of data: 1MB

Name: New York City Green Taxi trip data

Description: This dataset contains the detailed records of taxi trips in New York City. The columns include trip pick-up and drop-off dates, times and coordinates, number of passengers, trip distance and taxi fees, etc.

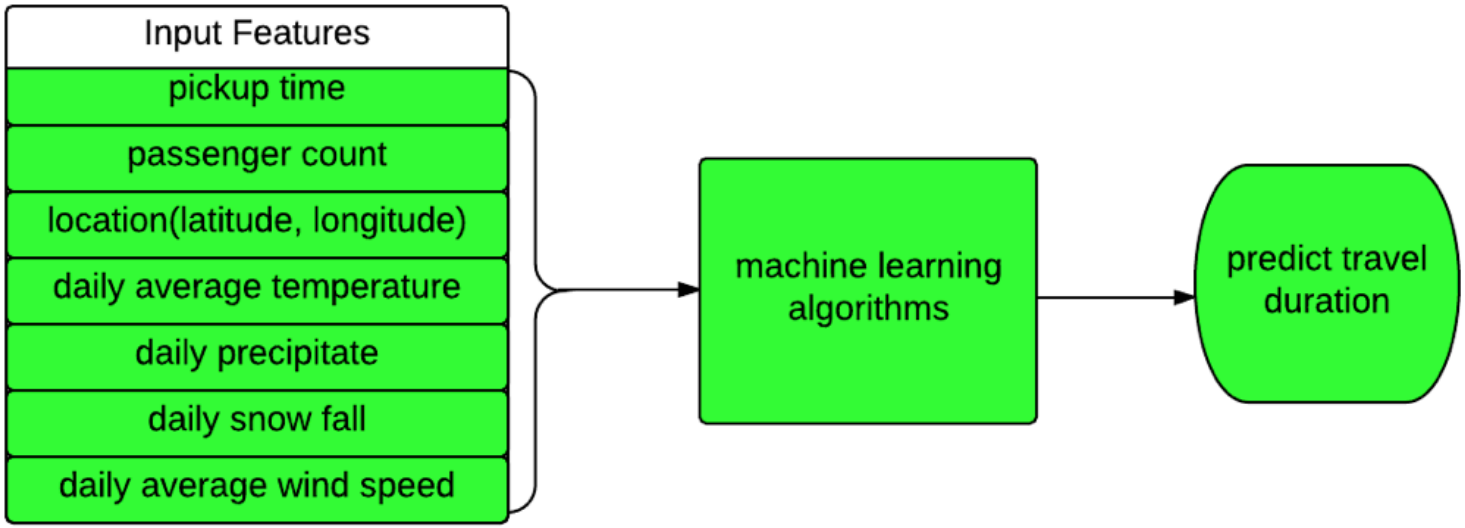
Size of data: 2GB

Name: New York CitiBike data

Description: This dataset contains the detailed records of CitiBike trip histories in New York City. The columns include start and end dates and times, station names, station coordinates and trip durations, etc.

Size of data: 4GB

Design Diagram - input features

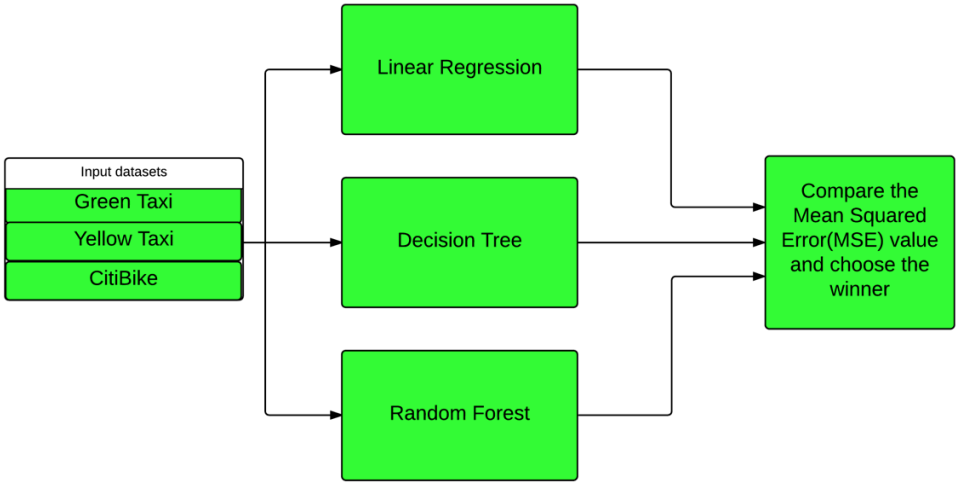


Platform(s) on which the application runs:

NYU HPC cluster

Travel Duration Prediction of Taxi and Bike Trips in NYC

Design Diagram - Machine Learning algorithms



Platform(s) on which the application runs:

NYU HPC cluster

Experiments/Results - Taxi trips

Conclusion: Weather condition(temperature, precipitation, snow fall, wind speed) can affect trip duration. We can use weather condition, date & time and location(latitude, longitude) to predict trip duration.

1. Regression using Linear Regression

the optimized parameters: number of iterations = 25, step size = 1.0E-7,

MSE(Mean Squared Error) = 166183.241785208

2. Using Random Forest

the optimized parameters: max depth = 16, max bins = 256, number of trees = 64,

MSE = 69830.45453755137

3. Using Decision Tree

the optimized parameters: max depth = 8, max bins 48,

MSE = 79335.78372910304

Experiments/Results - CitiBike Trips

Conclusion: Weather condition(temperature, precipitation, snow fall, wind speed) can affect trip duration. We can use weather condition, date & time and location(latitude, longitude) to predict trip duration.

1. Regression using Linear Regression

the optimized parameters: number of iterations = 100, step size = 1.0E-6,

MSE(Mean Squared Error) = 103504.69125622454

2. Using Random Forest

the optimized parameters: max depth = 12, max bins = 128, number of trees = 128,

MSE = 84521.91028820915

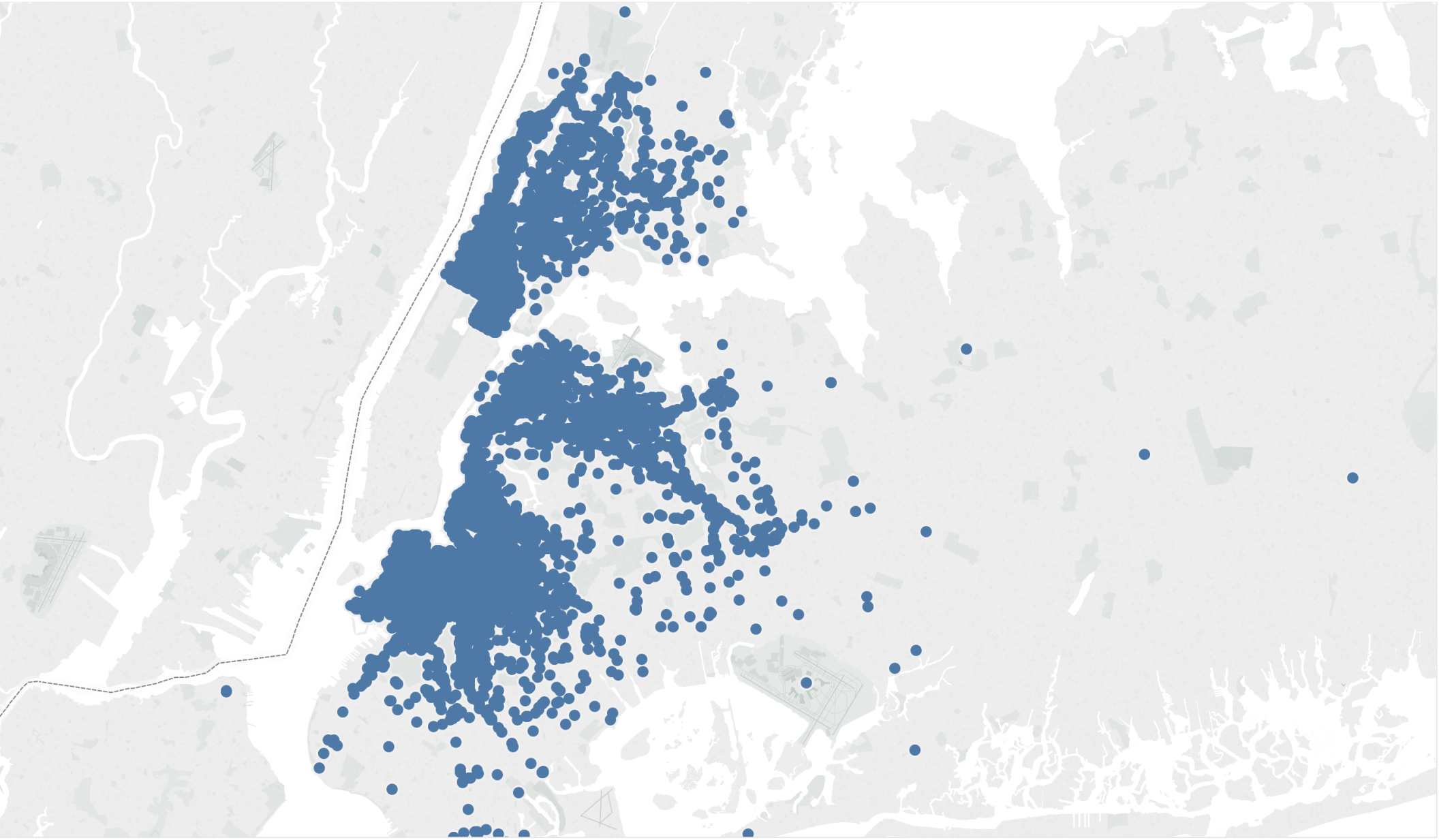
3. Using Decision Tree

the optimized parameters: max depth = 4, max bins 48,

MSE = 95883.47439572362

Green Taxi pickup location

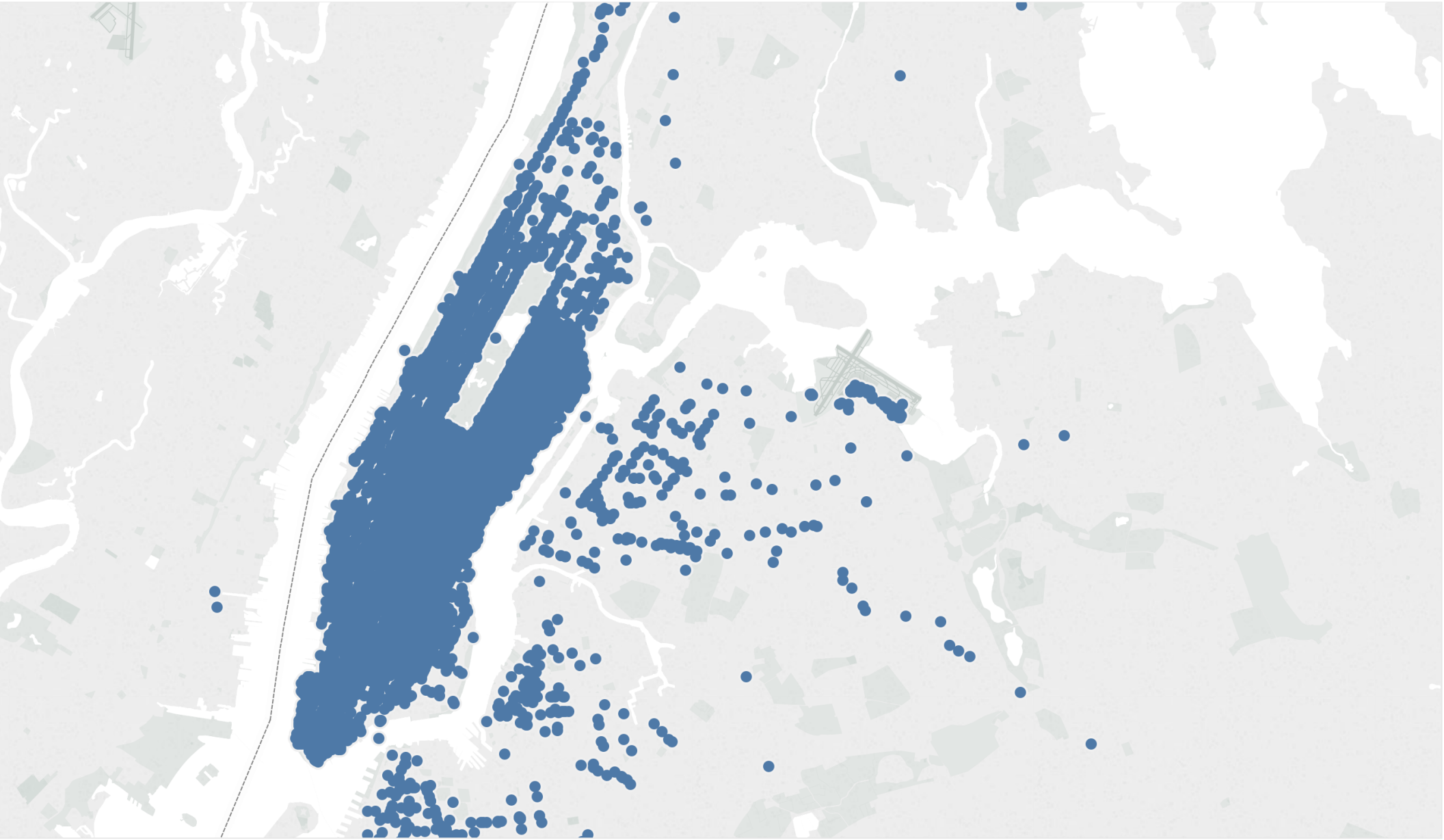
Sheet 1



Map based on long1 and lat1.

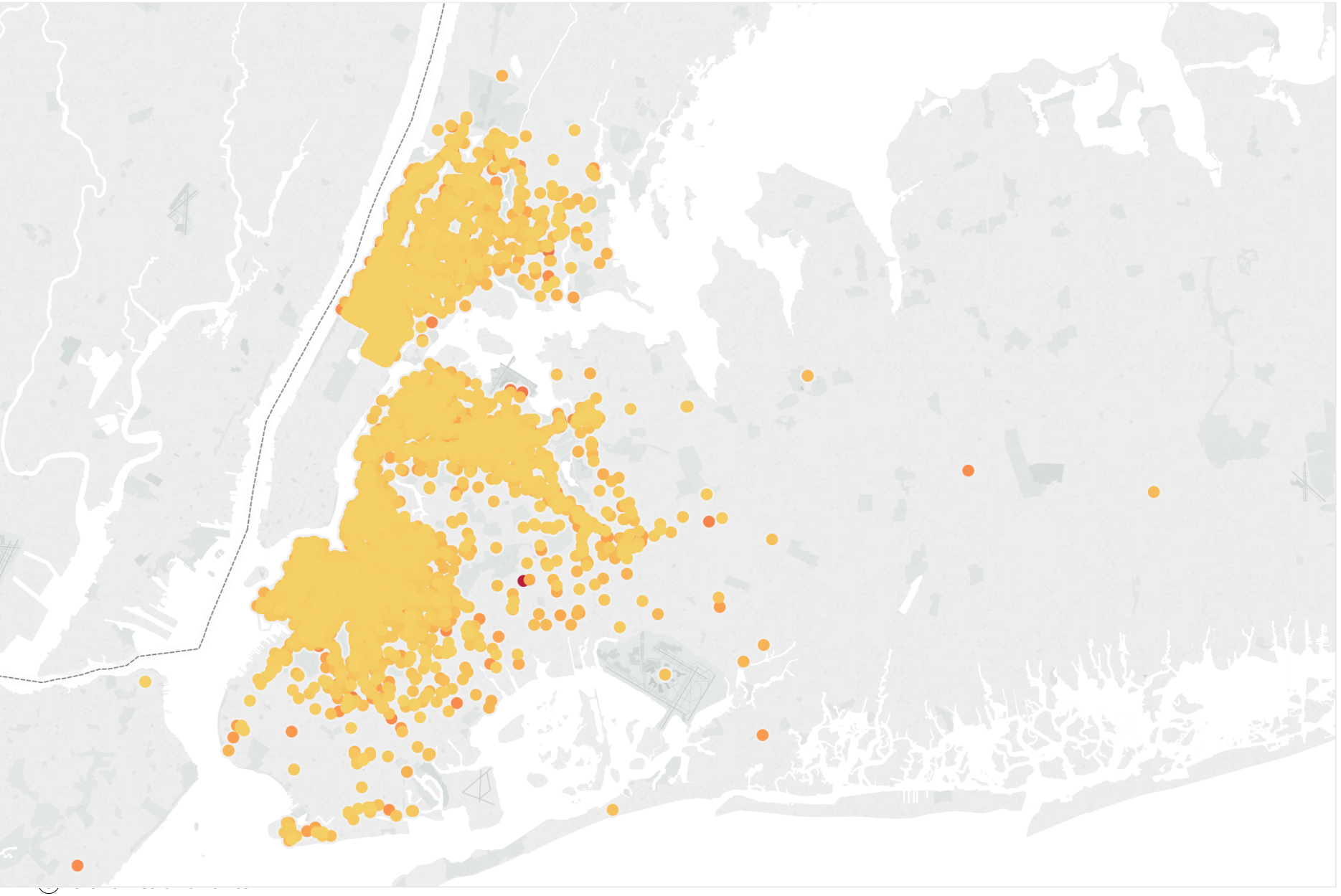
Yellow Taxi - pickup location

Sheet 1

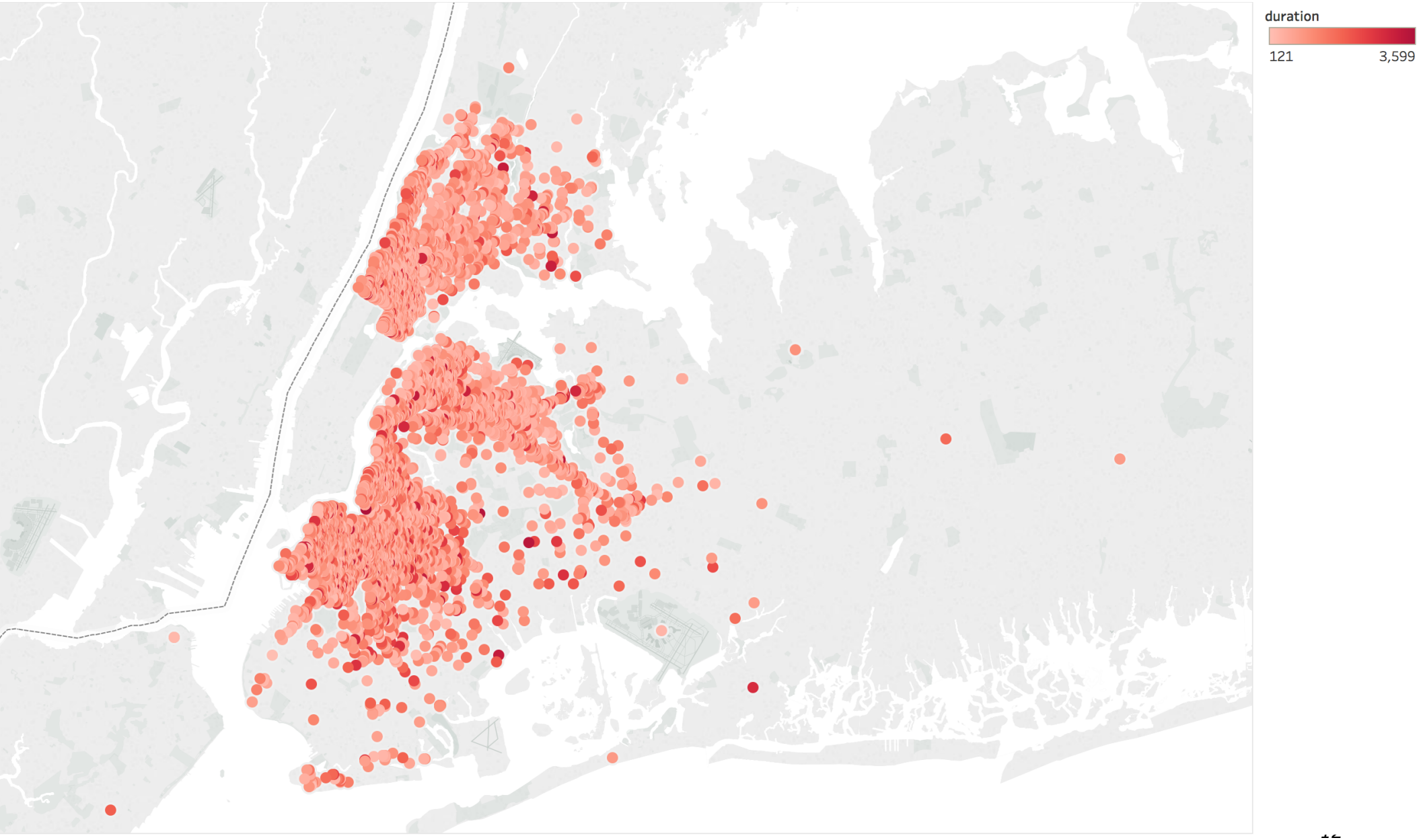


Map based on long1 and lat1.

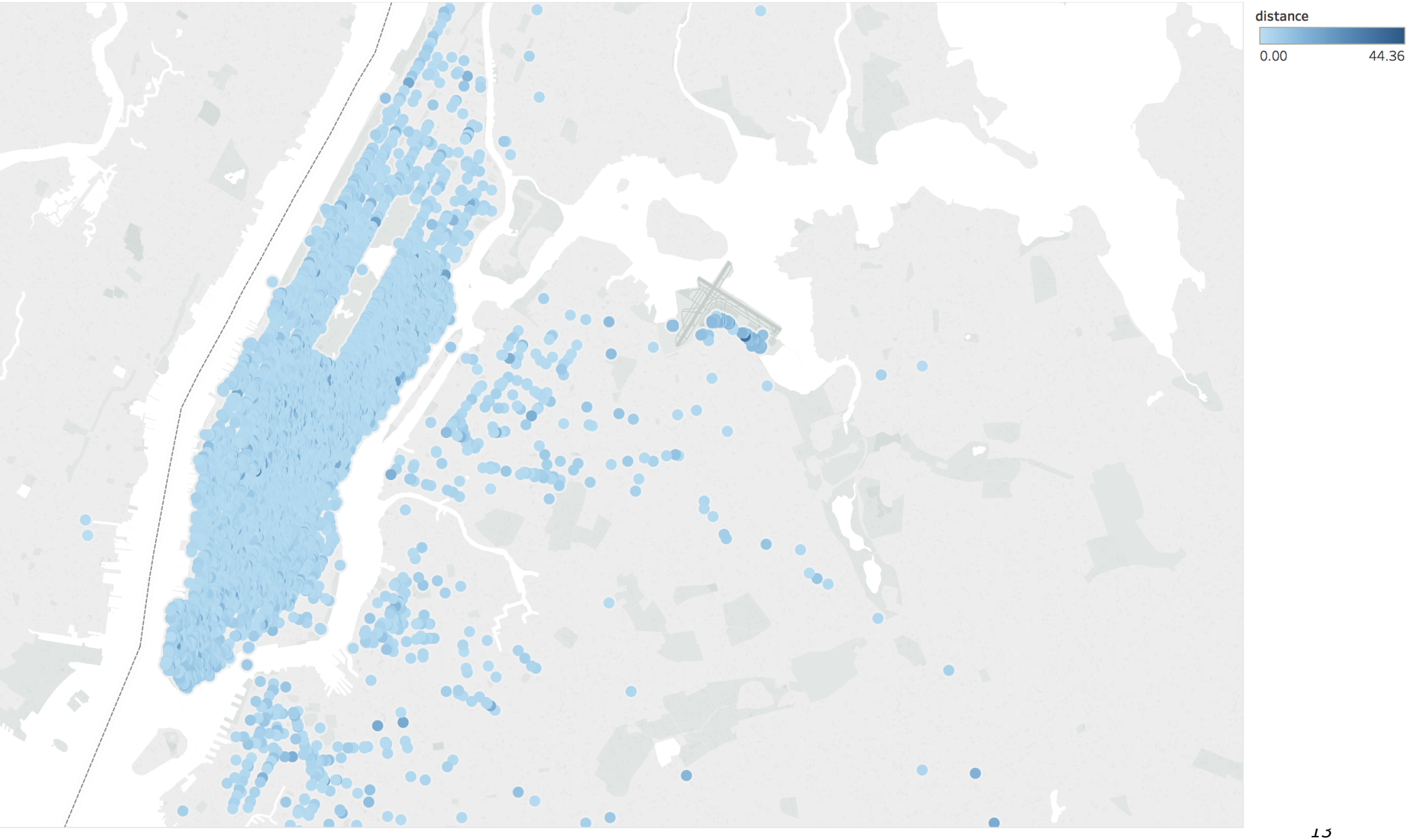
Travel Distance in miles - Green taxi



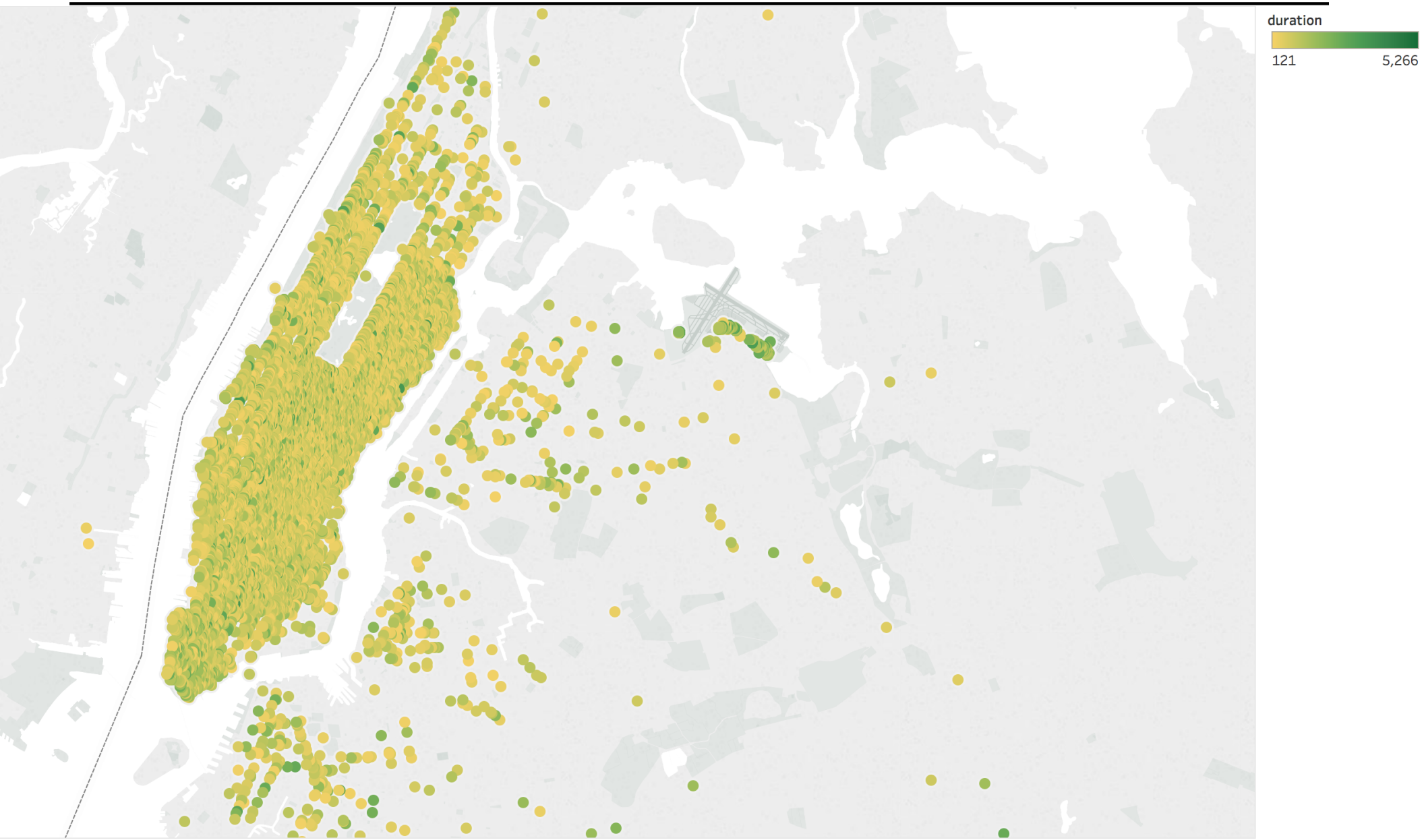
Trip Duration in seconds - Green Taxi



Travel distance in miles - Yellow Taxi



Travel duration in miles - Yellow Taxi



Training data

```
wl1731@login-2-1:~/... ⌘1   ~/code/Spark (zsh) ⌘2
        user: wl1731
17/12/12 14:43:34 INFO cluster.YarnClientSchedulerBackend: Application application_1512409949844_3850 has started running.
17/12/12 14:43:34 INFO util.Utils: Successfully started service 'org.apache.spark.network.netty.NettyBlockTransferService' on po
rt 43612.
17/12/12 14:43:34 INFO netty.NettyBlockTransferService: Server created on 43612
17/12/12 14:43:34 INFO storage.BlockManager: external shuffle service port = 7337
17/12/12 14:43:34 INFO storage.BlockManagerMaster: Trying to register BlockManager
17/12/12 14:43:34 INFO storage.BlockManagerMasterEndpoint: Registering block manager 10.0.255.254:43612 with 530.3 MB RAM, Block
ManagerId(driver, 10.0.255.254, 43612)
17/12/12 14:43:34 INFO storage.BlockManagerMaster: Registered BlockManager
17/12/12 14:43:34 INFO scheduler.EventLoggingListener: Logging events to hdfs://dumbo/user/spark/applicationHistory/application_
1512409949844_3850
17/12/12 14:43:34 INFO spark.SparkContext: Registered listener com.cloudera.spark.lineage.ClouderaNavigatorListener
17/12/12 14:43:35 INFO cluster.YarnClientSchedulerBackend: SchedulerBackend is ready for scheduling beginning after reached minR
egisteredResourcesRatio: 0.8
NUMBER OF CLUSTERS: 6
  Within Set Sum of Squared Errors = 25.697151745129492
NUMBER OF CLUSTERS: 7
  Within Set Sum of Squared Errors = 23.0434743633463
NUMBER OF CLUSTERS: 8
  Within Set Sum of Squared Errors = 18.134322944377843
NUMBER OF CLUSTERS: 9
  Within Set Sum of Squared Errors = 16.46426148622249
NUMBER OF CLUSTERS: 10
  Within Set Sum of Squared Errors = 14.625325164718038
NUMBER OF CLUSTERS: 11
  Within Set Sum of Squared Errors = 13.279636236086706
NUMBER OF CLUSTERS: 12
  Within Set Sum of Squared Errors = 12.42037193044033
-----------------------------------------------------
Best Choice: NUMBER OF CLUSTERS 8
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/opt/cloudera/parcels/CDH-5.11.1-1.cdh5.11.1.p0.4/jars/hive-exec-1.1.0-cdh5.11.1.jar!/shaded/p
arquet/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/opt/cloudera/parcels/CDH-5.11.1-1.cdh5.11.1.p0.4/jars/hive-jdbc-1.1.0-cdh5.11.1-standalone.ja
r!/shaded/parquet/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/opt/cloudera/parcels/CDH-5.11.1-1.cdh5.11.1.p0.4/jars/parquet-format-2.1.0-cdh5.11.1.jar!/sha
ded/parquet/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/opt/cloudera/parcels/CDH-5.11.1-1.cdh5.11.1.p0.4/jars/parquet-hadoop-bundle-1.5.0-cdh5.11.1.j
ar!/shaded/parquet/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/opt/cloudera/parcels/CDH-5.11.1-1.cdh5.11.1.p0.4/jars/parquet-pig-bundle-1.5.0-cdh5.11.1.jar!
/shaded/parquet/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [shaded.parquet.org.slf4j.helpers.NOPLoggerFactory]
-----------------------------------------------------
MAX DEPTH: 2;   MAX BINS: 8
Training Data Mean Squared Error = 116320.95933329541
Test Data Mean Squared Error = 120378.11045430788
MAX DEPTH: 2;   MAX BINS: 16
Training Data Mean Squared Error = 113964.28171883269
Test Data Mean Squared Error = 116868.41102947506
MAX DEPTH: 2;   MAX BINS: 24
Training Data Mean Squared Error = 113855.38685823335
Test Data Mean Squared Error = 117210.63478530858
```

Training data

```
wl1731@login-2-1:~/... ⌘1   ~/code/Spark (zsh) ⌘2
MAX DEPTH: 12;   MAX BINS: 48
Training Data Mean Squared Error = 32970.57794174897
Test Data Mean Squared Error = 115828.2818933513
MAX DEPTH: 12;   MAX BINS: 64
Training Data Mean Squared Error = 34840.77582397362
Test Data Mean Squared Error = 111570.46567913883
MAX DEPTH: 12;   MAX BINS: 96
Training Data Mean Squared Error = 31567.111131065354
Test Data Mean Squared Error = 114366.19981033762
MAX DEPTH: 12;   MAX BINS: 128
Training Data Mean Squared Error = 29514.713427004484
Test Data Mean Squared Error = 109968.07616081297
MAX DEPTH: 12;   MAX BINS: 256
Training Data Mean Squared Error = 30866.1329388697
Test Data Mean Squared Error = 108398.59751427575
MAX DEPTH: 12;   MAX BINS: 512
Training Data Mean Squared Error = 31254.48290347087
Test Data Mean Squared Error = 105273.67385765968
MAX DEPTH: 16;   MAX BINS: 8
Training Data Mean Squared Error = 11152.426589923443
Test Data Mean Squared Error = 148963.0475104082
MAX DEPTH: 16;   MAX BINS: 16
Training Data Mean Squared Error = 9826.885364105123
Test Data Mean Squared Error = 148867.4011774378
MAX DEPTH: 16;   MAX BINS: 24
Training Data Mean Squared Error = 10767.885980117746
Test Data Mean Squared Error = 134201.50668280295
MAX DEPTH: 16;   MAX BINS: 32
Training Data Mean Squared Error = 9060.737542597868
Test Data Mean Squared Error = 144509.91139170522
MAX DEPTH: 16;   MAX BINS: 48
Training Data Mean Squared Error = 12031.157365511313
Test Data Mean Squared Error = 134818.6547624563
MAX DEPTH: 16;   MAX BINS: 64
Training Data Mean Squared Error = 12925.251058794376
Test Data Mean Squared Error = 129699.9787820123
MAX DEPTH: 16;   MAX BINS: 96
Training Data Mean Squared Error = 12737.418716882908
Test Data Mean Squared Error = 134635.6863115284
MAX DEPTH: 16;   MAX BINS: 128
Training Data Mean Squared Error = 8442.72226595769
Test Data Mean Squared Error = 125824.79551237321
MAX DEPTH: 16;   MAX BINS: 256
Training Data Mean Squared Error = 9408.215075595355
Test Data Mean Squared Error = 127296.08738525983
MAX DEPTH: 16;   MAX BINS: 512
Training Data Mean Squared Error = 11563.891847000807
Test Data Mean Squared Error = 126971.58093055706
-----------------------------------------------------
Best Choice: MAX DEPTH 7, MAX BINS 64
  with MSE 85192.04544289401
Optimization complete. Exit.
-----------------------------------------------------
[wl1731@login-2-1 proj]$
```

Module

```
wl1731@login-2-1:~/... ⌘1   ~/code/Spark (zsh) ⌘2
        Else (feature 8 > 6008.132698490618)
         Predict: 1637.7861635220127
    Else (feature 6 > 11.16)
     If (feature 10 <= 0.0)
      If (feature 12 <= 75.0)
       If (feature 1 <= 0.9481828703703704)
        If (feature 8 <= 2799.232149192425)
         Predict: 3437.3333333333335
        Else (feature 8 > 2799.232149192425)
         Predict: 2240.34126984127
       Else (feature 1 > 0.9481828703703704)
        If (feature 7 <= 4.0)
         Predict: 1599.25
        Else (feature 7 > 4.0)
         Predict: 2062.5
      Else (feature 12 > 75.0)
       If (feature 1 <= 0.23030092592592594)
        If (feature 1 <= 0.06335648148148149)
         Predict: 1332.0
        Else (feature 1 > 0.06335648148148149)
         Predict: 2243.0
       Else (feature 1 > 0.23030092592592594)
        If (feature 3 <= -73.92237091064453)
         Predict: 3185.1
        Else (feature 3 > -73.92237091064453)
         Predict: 2443.3333333333335
    Else (feature 10 > 0.0)
     If (feature 1 <= 0.3160763888888889)
      If (feature 0 <= 0.11748633879781421)
       If (feature 3 <= -73.99595642089844)
        Predict: 2141.0
       Else (feature 3 > -73.99595642089844)
        Predict: 2880.0
      Else (feature 0 > 0.11748633879781421)
       If (feature 12 <= 62.0)
        Predict: 1714.0
       Else (feature 12 > 62.0)
        Predict: 2005.6666666666667
     Else (feature 1 > 0.3160763888888889)
      If (feature 12 <= 48.0)
       If (feature 15 <= 10.4)
        Predict: 2420.235294117647
       Else (feature 15 > 10.4)
        Predict: 3373.0
      Else (feature 12 > 48.0)
       If (feature 2 <= 40.73035430908203)
        Predict: 2723.0
       Else (feature 2 > 40.73035430908203)
        Predict: 3260.625

-------------------------------------------------------
All Data Mean Squared Error = 74067.47528008108
-------------------------------------------------------
[wl1731@login-2-1 proj]$
```

Obstacles

1. Find the right Machine Learning algorithms

As novices to Machine Learning, we at first struggled a lot with the right algorithms. But we finally figured out it is a regression problem and we decided to utilize three regression algorithms: linear regression, decision tree and random forest.

2. Some data sets are incomplete

Originally we planed to use 5 data sets: weather data, taxi trip data, CitiBike data, Uber data and bus data. But we have to drop Uber data and bus data because these two data sets are incomplete.

3. Some data sets need to be cleaned multiple rounds as analysis improves

We profiled and cleaned our datasets from the very beginning, but the problem is that it is difficult to go through each row to make sure all data are correct when there are more than 10 million rows. For example, our algorithms performed poorly at first, and after intense investigation, we found that some taxi trip duration is more than 80,000 seconds(22 hours). And our algorithm's performance improved considerably after these outliers are filtered out.

Summary

- Commute time can be predicted
- Decision tree has the best performance. Clustering can improve the performance of regression
- Data cleaning is an integral part of data exploration

Acknowledgements

Special thanks to Professor McIntosh who is always behind us when we have difficulties in our study. Also thanks to NYU HPC stuff who provided the computing platform and essential help for the coding and debugging environment.

References

1. T. White. Hadoop: The Definitive Guide. O'Reilly Media Inc., Sebastopol, CA, May 2012.
2. Ho. Karau, etc. Learning Spark Lightning-Fast Data Analysis. O'Reilly Media Inc., Sebastopol, CA, February 2015.
3. M. Araki, etc. Impacts of Seasonal Factors on Travel Behavior: Basic Analysis of GPS Trajectory Data for 8 Months. Serviceology for Smart Service System pp 377-384
4. F. Wu, etc. Interpreting traffic dynamics using ubiquitous urban data. GIS '16 Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems Article No. 69

Thank you!