Big Data 220 Final Project

NYC Taxi Data Streaming

Kirk Force

March 17, 2020

The Setup

- I approached this project as if I had been given the task of building a pipeline to ingest the existing stream of taxi cab data and store it in a useful format for future downstream analysis and dissemination.

Data Background

- Instructor provided a Java application that simulated a stream of taxi cab data over a day's period.
- Data is returned as comma separated values sorted by ride dropoff date/time. Date is today, time is in GMT.
- Fields of interest include: Pickup time/location, dropoff time/location, trip distance, trip duration, fare amount and tip amount.

```
bash-4.4# kafka-console-consumer.sh --bootstrap-server UbuntuServer027.westus2.cloudapp.azure.com:9092 --topic FinalProject
2,2020-03-18 03:06:09,2020-03-18 03:15:45,1,1.61,1,N,249,186,1,8.5,0.5,0.5,2.45,0,0.3,12.25
2,2020-03-18 02:56:49,2020-03-18 03:15:46,5,5.42,1,N,231,239,1,19,0.5,0.5,4.06,0,0.3,24.36
1,2020-03-18 03:00:08,2020-03-18 03:15:47,2,3.80,1,N,141,114,1,14,0.5,0.5,1,0,0.3,16.3
1,2020-03-18 03:06:00,2020-03-18 03:15:48,1,2.70,1,N,162,262,1,10,0.5,0.5,2.25,0,0.3,13.55
2,2020-03-18 03:02:22,2020-03-18 03:15:49,3,3.03,1,N,148,186,1,12.5,0.5,0.5,2.76,0,0.3,16.56
1,2020-03-18 03:05:18,2020-03-18 03:15:50,1,1.90,1,N,79,209,1,9,0.5,0.5,2.55,0,0.3,12.85
1,2020-03-18 03:02:46,2020-03-18 03:15:50,1,3.20,1,N,17,61,1,12.5,0.5,0.5,2.75,0,0.3,16.55
2,2020-03-18 02:39:54,2020-03-18 03:15:52,1,18.77,2,N,132,50,1,52,0,0.5,10,5.76,0.3,68.56
1,2020-03-18 02:55:18,2020-03-18 03:15:53,1,8.60,1,N,234,223,1,26.5,0.5,0.5,10.05,5.76,0.3,43.61
2,2020-03-18 03:09:33,2020-03-18 03:15:54,1,1.84,1,N,164,237,1,7.5,0.5,0.5,2,0,0.3,10.8
1,2020-03-18 02:51:31,2020-03-18 03:15:55,1,10.80,1,N,138,230,1,32.5,0.5,0.5,7.9,5.76,0.3,47.46
2,2020-03-18 02:51:21,2020-03-18 03:15:56,1,16.28,1,N,132,255,1,44.5,0.5,0.5,9.16,0,0.3,54.96
2,2020-03-18 02:36:00,2020-03-18 03:15:56,1,8.91,1,N,246,36,2,32,0.5,0.5,0,0,0.3,33.3
2,2020-03-18 03:08:00,2020-03-18 03:15:58,4,1.47,1,N,158,231,1,8,0.5,0.5,1,0,0.3,10.3
1,2020-03-18 03:02:19,2020-03-18 03:15:58,1,8.90,1,N,48,127,1,25,0.5,0.5,5,0,0.3,31.3
2,2020-03-18 03:10:34,2020-03-18 03:16:00,1,2.00,1,N,48,238,1,7.5,0.5,0.5,2.2,0,0.3,11
2,2020-03-18 03:11:33,2020-03-18 03:16:00,1,.72,1,N,232,144,1,5,0.5,0.5,1,0,0.3,7.3
1,2020-03-18 03:10:56,2020-03-18 03:16:01,1,1.00,1,N,164,161,1,5.5,0.5,0.5,1.35,0,0.3,8.15
```

Stream Setup

- Similar to the homework, I set up a Kafka topic for the stream and ran the stream-producing application through that topic.
- From a Databricks notebook, I connected to the stream and was able to parse the data into a more useful DataFrame.

```
1 display(taxiData)
```

Cancel

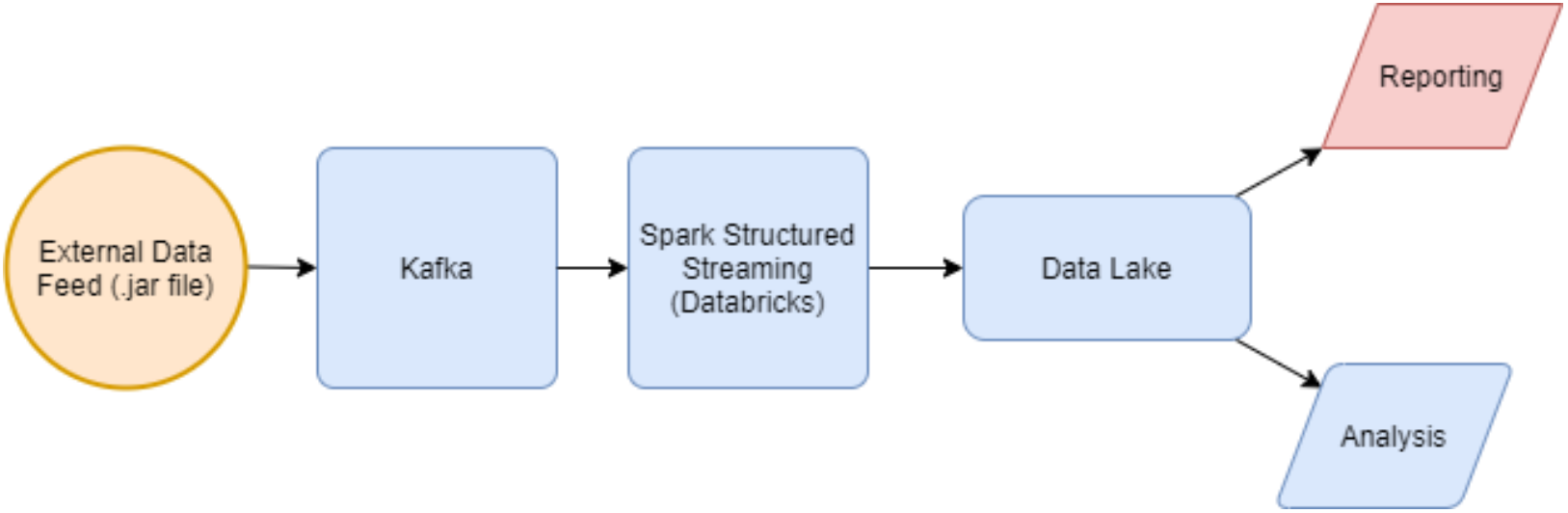
(1) Spark Jobs

display_query_3 (id: 200c751d-159c-43d6-8134-6b651639b4fb) *Last updated: 5 seconds ago*

VendorID	tpep_pickup_datetime	tpep_dropoff_datetime	passenger_count	trip_distance	RatecodeID	store_and_fwd_flag	PULocationID	DOLocationID	payment_type	fare_amount	extra	mta_tax	tip_amo
2	2020-03-09 03:40:41	2020-03-09 03:40:41	2	2.50	1	N	230	50	2	18	0.5	0.5	0
1	2020-03-09 03:34:48	2020-03-09 03:40:43	3	1.30	1	N	162	186	2	6.5	0.5	0.5	0
2	2020-03-09 03:24:39	2020-03-09 03:40:44	1	2.74	1	N	262	24	1	13	0.5	0.5	3.58
1	2020-03-09 03:17:39	2020-03-09 03:40:46	2	6.20	1	N	50	256	1	22	0.5	0.5	5.8
2	2020-03-09 03:19:52	2020-03-09 03:40:47	1	3.86	1	N	144	163	2	16	0.5	0.5	0
1	2020-03-09 03:33:34	2020-03-09 03:40:48	1	1.10	1	N	170	79	1	6.5	0.5	0.5	1.55
2	2020-03-09 03:30:09	2020-03-09 03:40:48	1	1.58	1	N	161	230	1	8.5	0.5	0.5	1.96
2	2020-03-09 03:26:13	2020-03-09 03:40:49	2	2.98	1	N	48	79	1	12.5	0.5	0.5	2.76

- I ran the streaming application for several hours over a couple days and saved to Parquet in Azure Data Lake.

Flow Diagram



Data Lake Storage

- Data is saved in a nested directory structure organized by Year/Month/Day.
- Individual years, months, and days can be all loaded into a Spark RDD with a single command.
- A month of data can be easily loaded and resaved as CSV for release to the public.

```
//Extract year, month, and day for archiving to Data Lake
val taxiWithDate = taxiData
                    .withColumn("pickupYear", year(to_date($"tpep_pickup_datetime")))
                    .withColumn("pickupMonth", month(to_date($"tpep_pickup_datetime")))
                    .withColumn("pickupDay", dayofmonth(to_date($"tpep_pickup_datetime")))
```

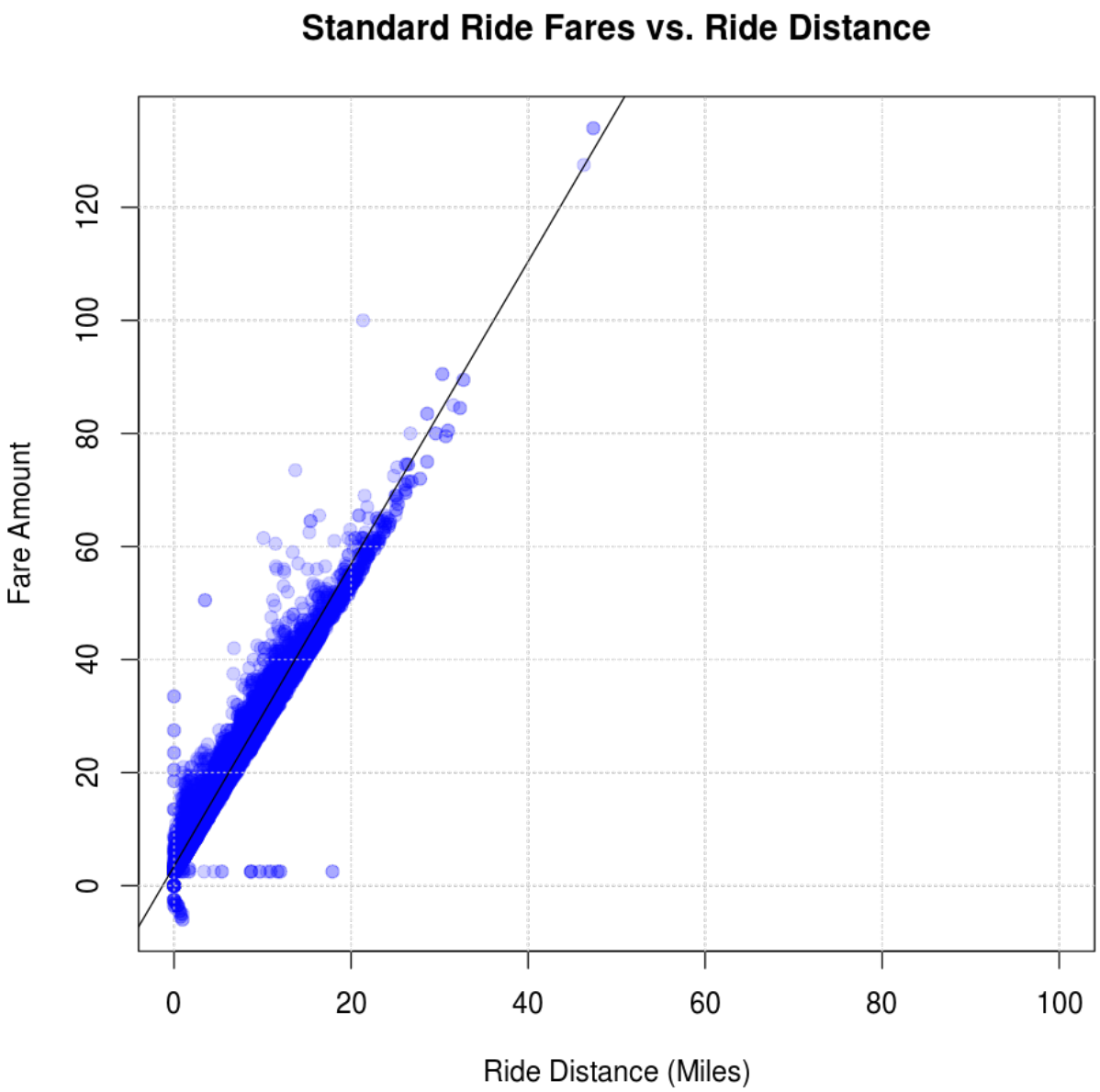
Cmd 6

```
//Save to nested year-month-day data structure
taxiWithDate.writeStream
            .partitionBy("pickupYear", "pickupMonth", "pickupDay")
            .outputMode("append").format("parquet")
            .option("path", "abfss://kforce@uwbigdatatechnologies.dfs.core.windows.net/Final Project/YellowCab")
            .option("checkpointLocation", "abfss://kforce@uwbigdatatechnologies.dfs.core.windows.net/Final Project/YellowCabCheckpoint").start()
```

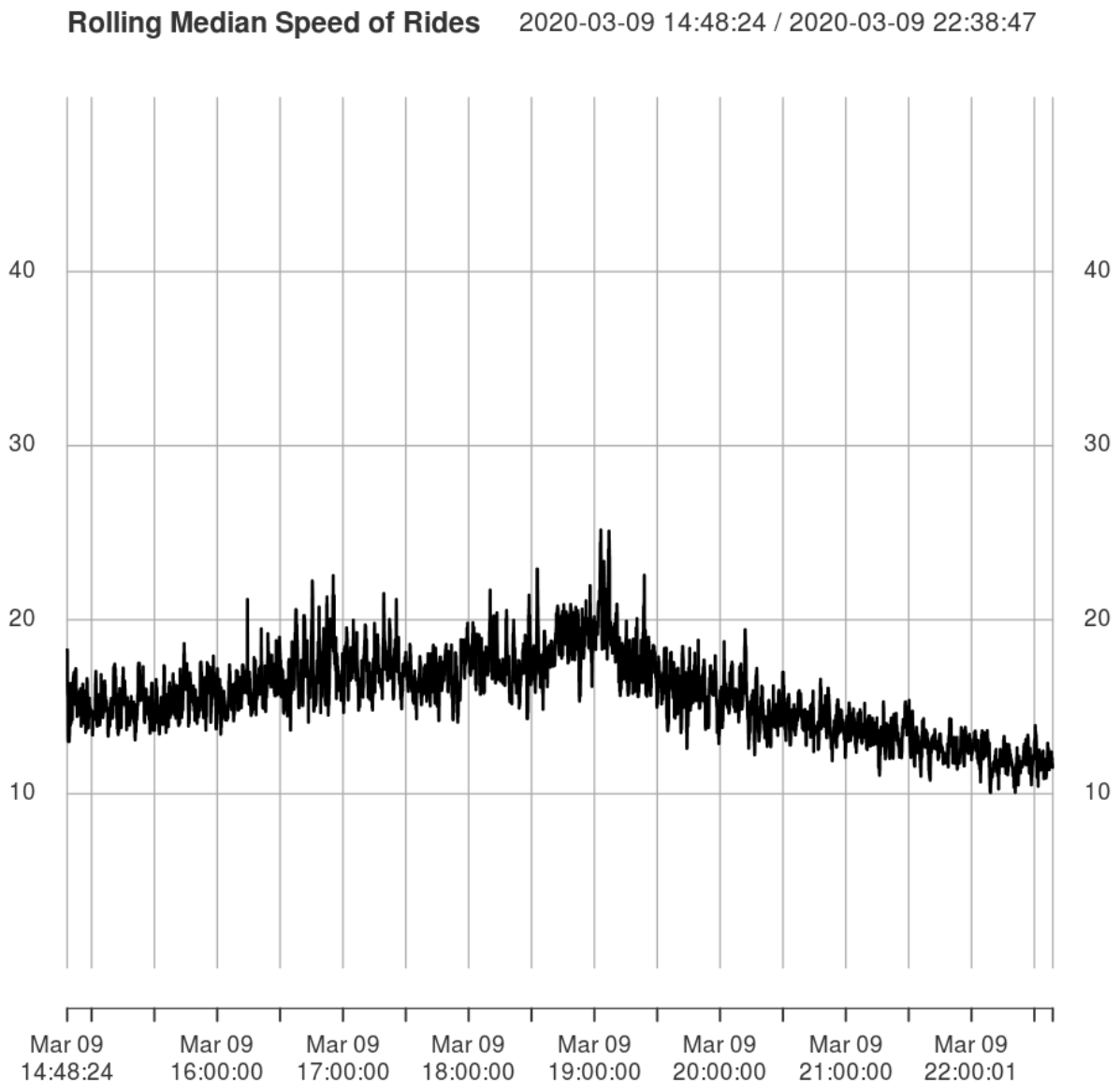
Analysis of Data

- I also did some analysis on the data.
- Streamed for around 8 hours straight on 3/9/2020.
- I loaded this data up into a SparkR notebook to do some analysis.

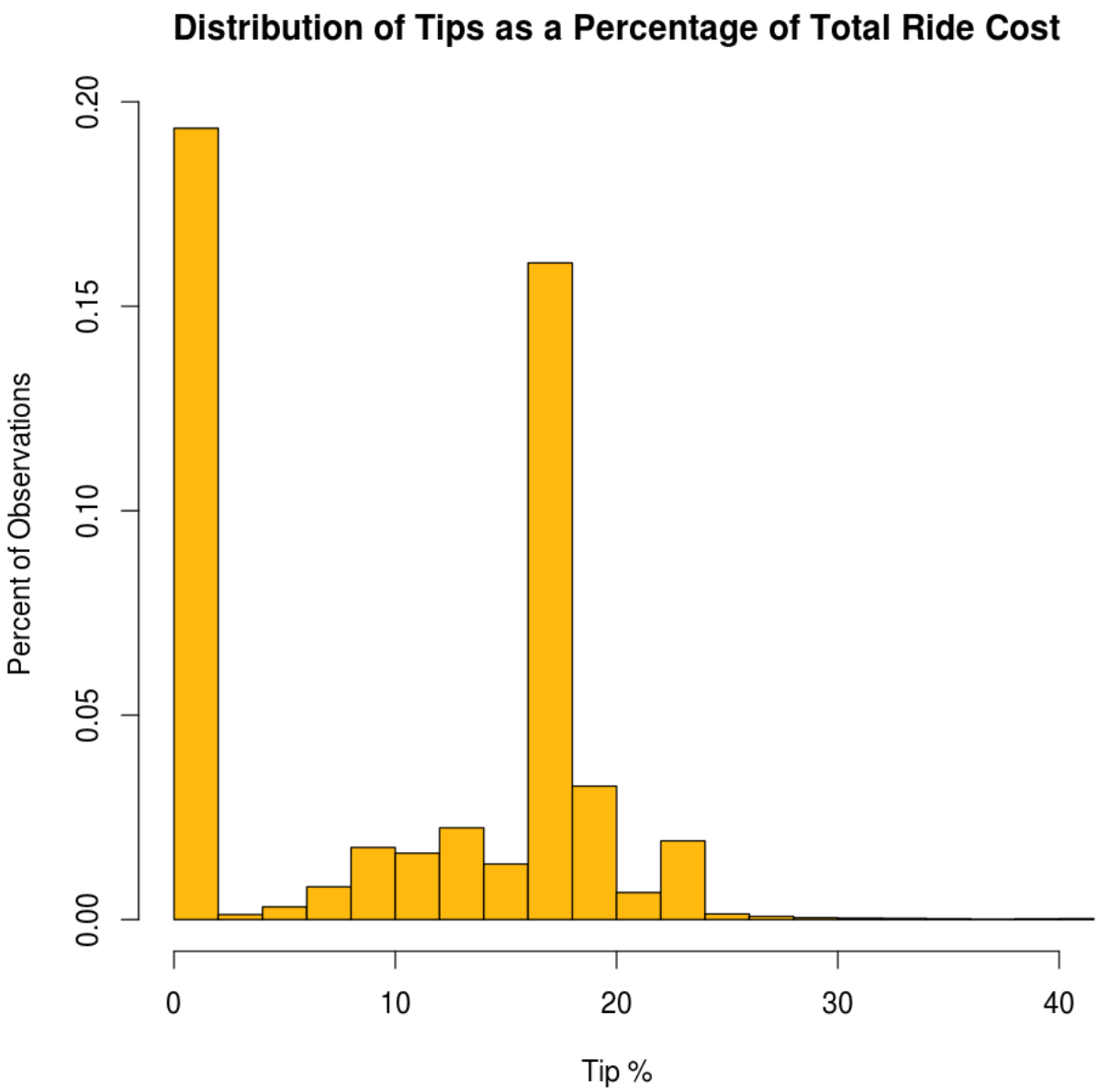
Analysis: Fare by Distance



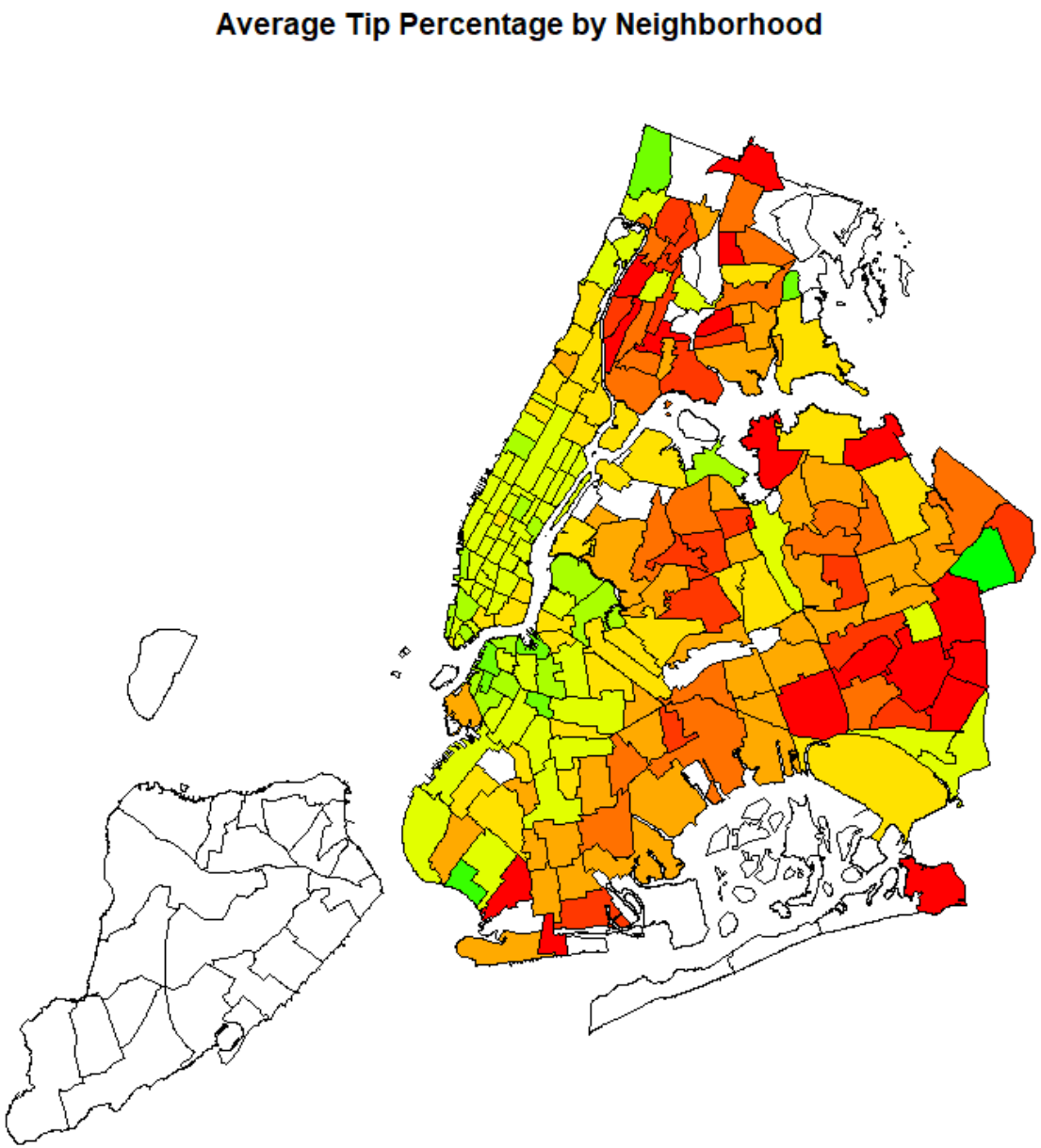
Analysis: Rolling Median Speed of Rides (100 Rides)



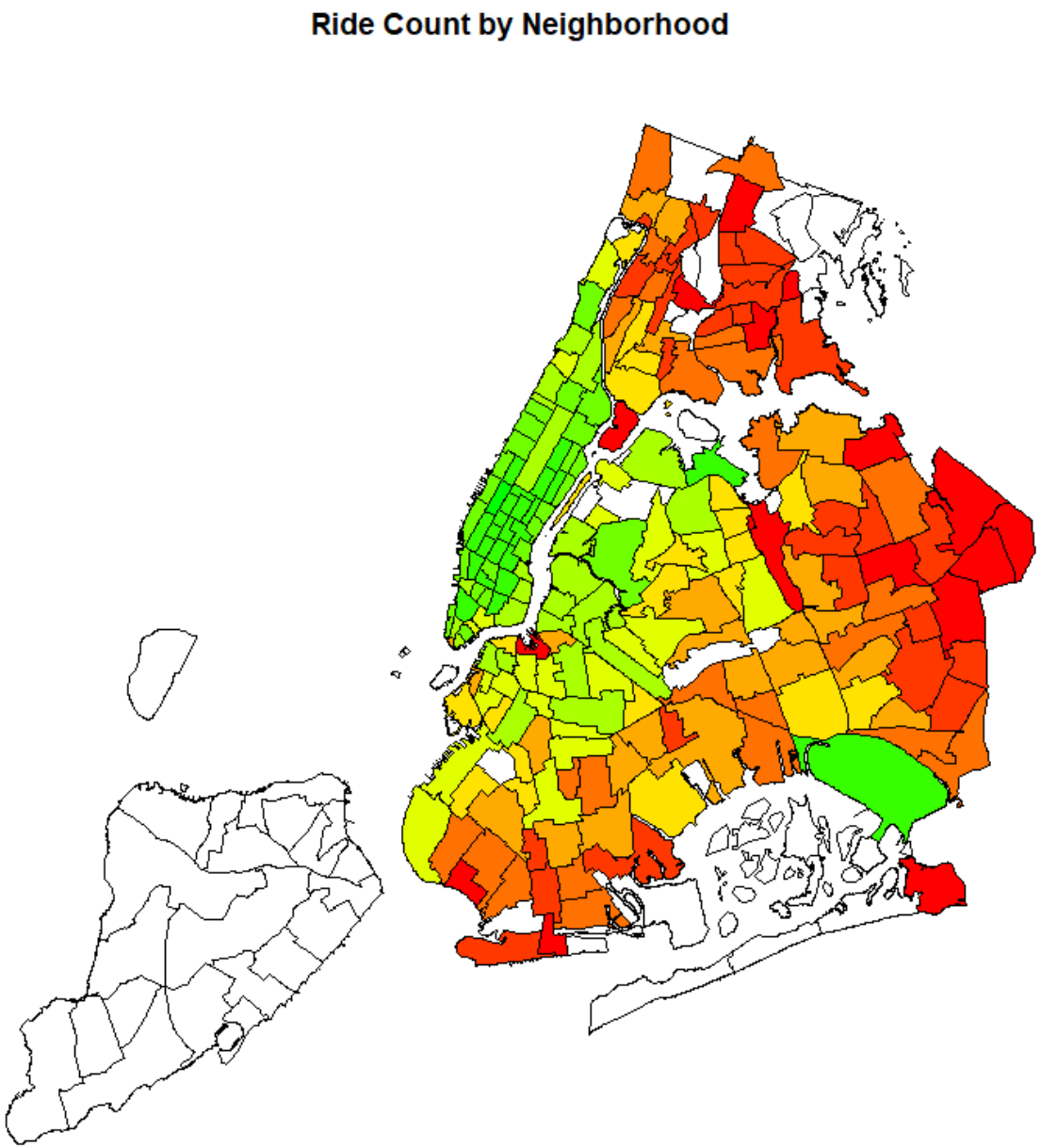
Analysis: Tip Percentage Distribution



Analysis: Tip Percentage by Neighborhood



Analysis: Ride Count by Neighborhood



Potential Improvements

- Data Lake may not be the best location for the data to rest.
- More real time processing with windowed queries.
- Plotting maps within the Databricks environment.
- More stable Kafka environment.