

A
Project Report
On
“NYC Automated Traffic Volume Counts”

Prepared by

Saurabh Jain | saurabh_jain@csu.fullerton.edu

Pratishtha Soni | pratishthasoni@csu.fullerton.edu

A Report Submitted to
Tseng-Ching James Shen, PhD
Department of Computer Science
College of Engineering and Computer Science (ECS)
California State University, Fullerton (CSUF)
for the Fulfillment of the Requirements for the
CPSC 531-01 Advance Database Management



**California State University, Fullerton
CA - 92831
December, 2022**

TABLE OF CONTENTS

| | |
|-----------------------------------|----|
| INTRODUCTION..... | 3 |
| FUNCTIONALITIES..... | 7 |
| ARCHITECTURE & DESGIN..... | 8 |
| GITHUB LOCATION OF CODE..... | 10 |
| DEPLOYMENT INSTRUCTIONS | 11 |
| STEPS TO RUN THE APPLICATION..... | 12 |
| TEST RESULT | 15 |

INTRODUCTION

Problem Statement

The vehicular traffic is increasing tremendously these days, simultaneously congestion also increases. To prevent congestion, one option is to increase the capacity by increasing the number of the existing transportation system. A second option is to develop alternatives that increase capacity by improving the efficiency of the existing transportation system. The latter focuses on building fewer lane miles.

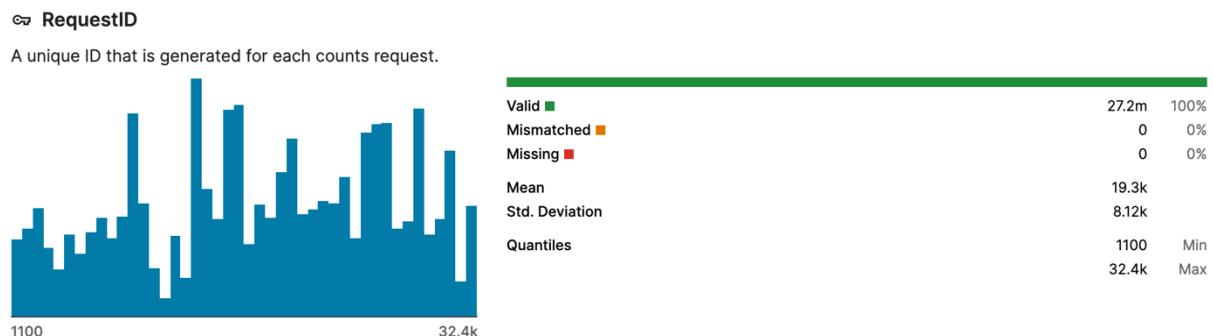
To Overcome these Problems, the New York City Department of Transportation (NYC DOT) uses Automated Traffic Recorders (ATR) to collect traffic sample volume counts at bridge crossings and roadways. These counts do not cover the entire year, and the number of days counted per location may vary from year to year.

Data Description

NYC Automated Traffic Count allows the user to view and access traffic data information of different cities in NYC. The information is displayed in the form of excel. New York City transportation uses automated traffic recorders to collect traffic volume counts at bridge crossings and roadways. The data consists of car volume, averaged every 15 mins for the duration of 2000 to 2020. This traffic data can be used to determine various type of analysis and design patterns in traffic over the year 2000 to 2020.

The information is collected from Kaggle in a form of 1 3GB excel file and consists of 14 rows and 27190511 columns:

RequestID - Unique ID that is generated for each count request.



Boro - Lists which of the five administrative divisions of New York City the location is within, written as a word. Where Brooklyn data is 29% and Queens data is 26%.

A Boro

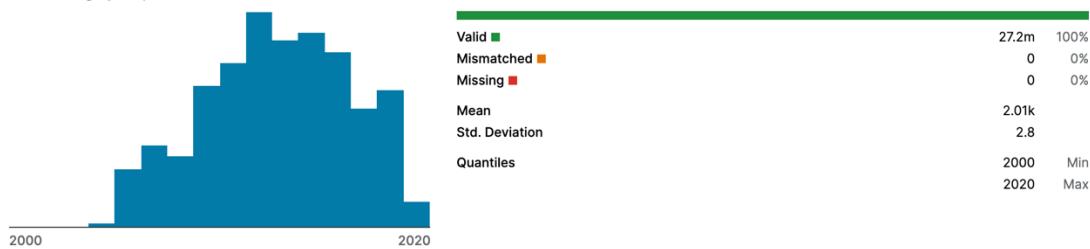
Lists which of the five administrative divisions of New York City the location is within, written as a word



Yr - The two-digit year portion of the date when the count was conducted. (From 2000 to 2020).

Yr

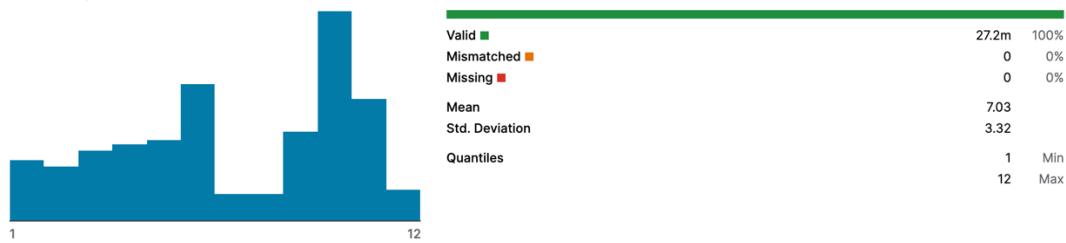
The two digit year portion of the date when the count was conducted.



M - The two-digit month portion of the date when the count was conducted.

M

The two digit month portion of the date when the count was conducted.



D - The two-digit day portion of the date when the count was conducted.

D

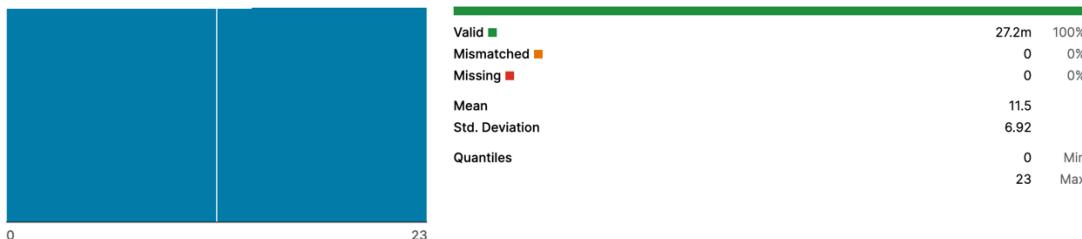
The two digit day portion of the date when the count was conducted.



HH - The two-digit hour portion of the time when the count was conducted.

HH

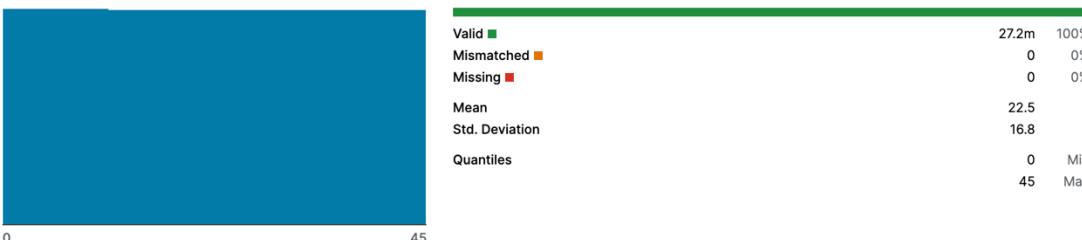
The two digit hour portion of the time when the count was conducted.



MM - The two digits start minute portion of the time when the count was conducted.

MM

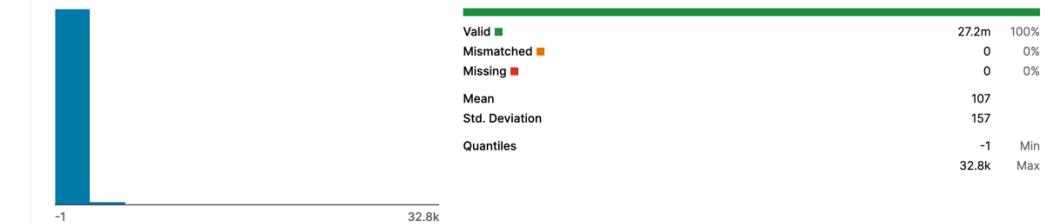
The two digit start minute portion of the time when the count was conducted.



Vol - The total sum of counts collected within 15-minute increments.

Vol

The total sum of count collected within a 15 minute increments.



SegmentID - The ID that identifies each segment of a street in the LION street network version 14.

≈ SegmentID

The ID that identifies each segment of a street in the LION street network version 14.



WktGeom - A text markup language for representing vector geometry objects on a map and spatial reference systems of spatial objects.

A WktGeom

A text markup language for representing vector geometry objects on a map and spatial reference systems of spatial objects.



Street - The 'On Street' where the count took place.

fromSt - The 'From Street' where the count took place.

toSt - The 'To Street' where the count took place.

Direction - The text-based direction of traffic where the count took place.

Tool and Technology Used:

- Spark
- Google Collab Notebook
- Google Cloud Platform
- Google Data Proc
- Python Matplotlib

FUNCTIONALITIES

1. Pre-Processing of Dataset:

In this step all the null values from different columns from the dataset is removed using the python language in jupyter notebook. Particularly, pandas library is being used for converting the datatypes (from string to integer and vice versa) and for removing null values and for removing the spaces before, after and in between of each string/int variable.

2. Processing of the Dataset on the Cloud:

The project data is in the form of excel of size **3 GB** and processing this huge data on local is challenging, so processed this bunch of data using Google's cloud dataproc service which is the open-source service for running Apache Hadoop and Apache spark and other tools and frameworks. The dataproc allows a new user 300 free credits to process and play with the data.

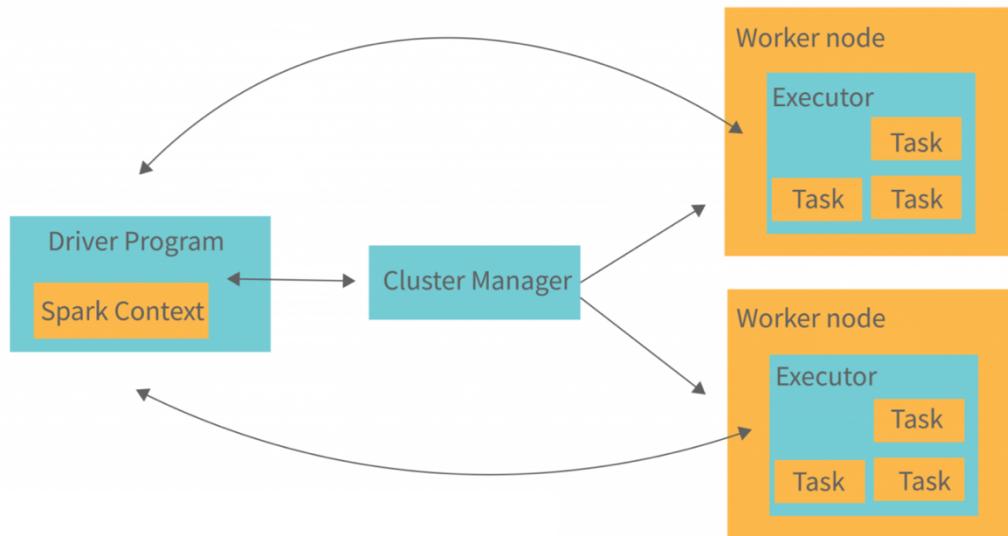
3. Data Analysis/Queries Significance:

- Enhance public safety.
- Reduce congestion.
- Improved access to travel and transit information.
- Generate cost savings to motor carriers, transit operators, toll authorities, and government agencies; and
- Reduce detrimental environmental impacts.

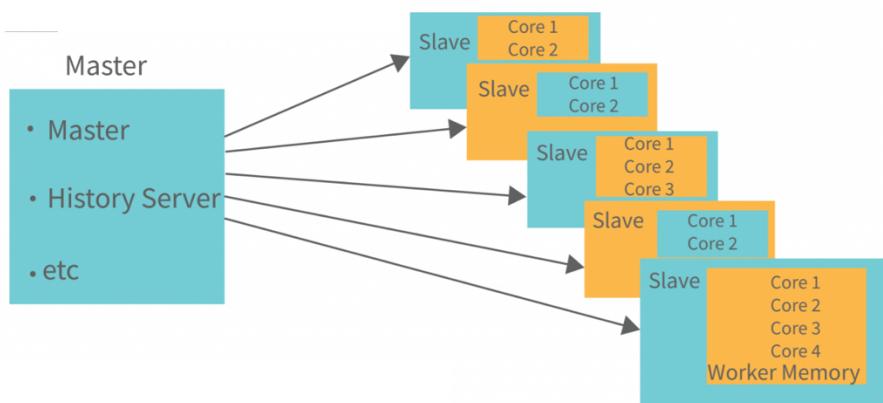
ARCHITECTURE & DESIGN

A high-level view of the Apache Spark application:

In the below diagram, spark uses the master and slave architecture. The driver program connects to a spark cluster, calls the main application, and builds a spark context (which serves as a gateway) to monitor the jobs running in the cluster. The spark context is implemented for all the operations.



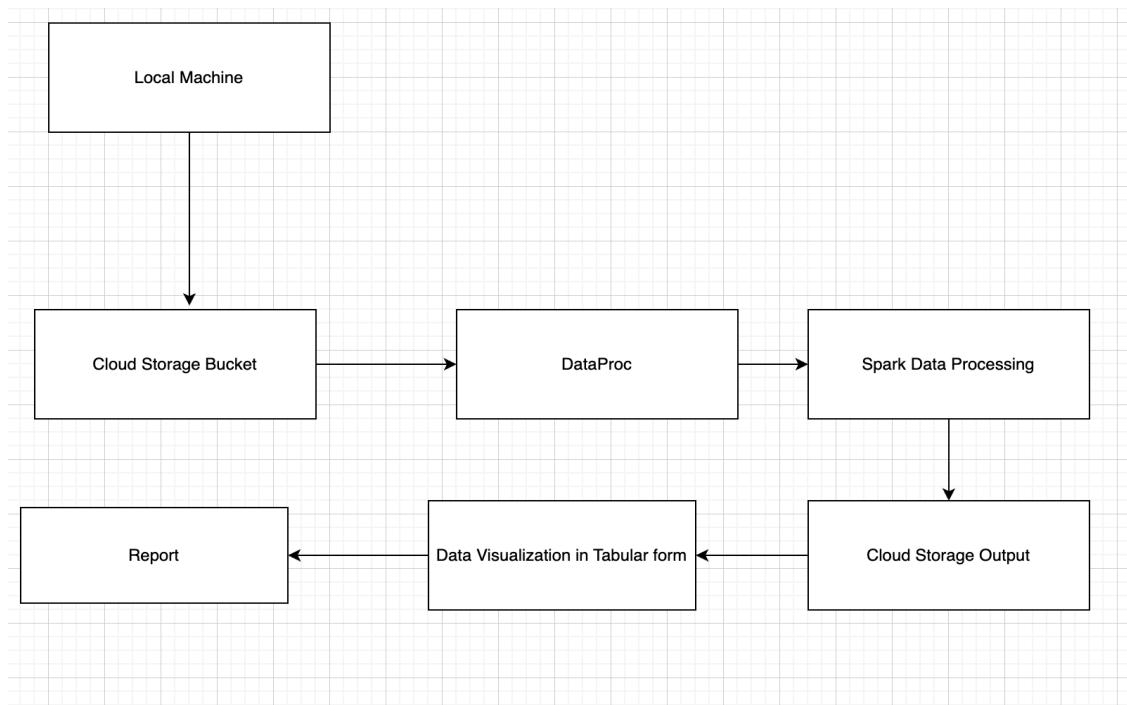
Spark Standalone Architecture



The architecture/implementation diagram of the project:

Methodology:

- **Local Machine:** The computer that you are currently logged on to as a user.
- **Cloud Storage Bucket:** Google Cloud Platform Console to upload files or folders.
- **DataProc:** Dataproc is a managed Spark and Hadoop service that lets you take advantage of open-source data tools for batch processing, querying, streaming, and machine learning.
- **Spark Data Processing:** Apache Spark is an open-source, distributed processing system used for big data workloads. It utilizes in-memory caching and optimized query execution for fast queries against data of any size. Simply put, Spark is a fast and general engine for large-scale data processing.
- **Cloud Storage Output:** Google Cloud Platform Console to upload files or folders.
- **Data Visualization:** Visualization of data using Python Library.

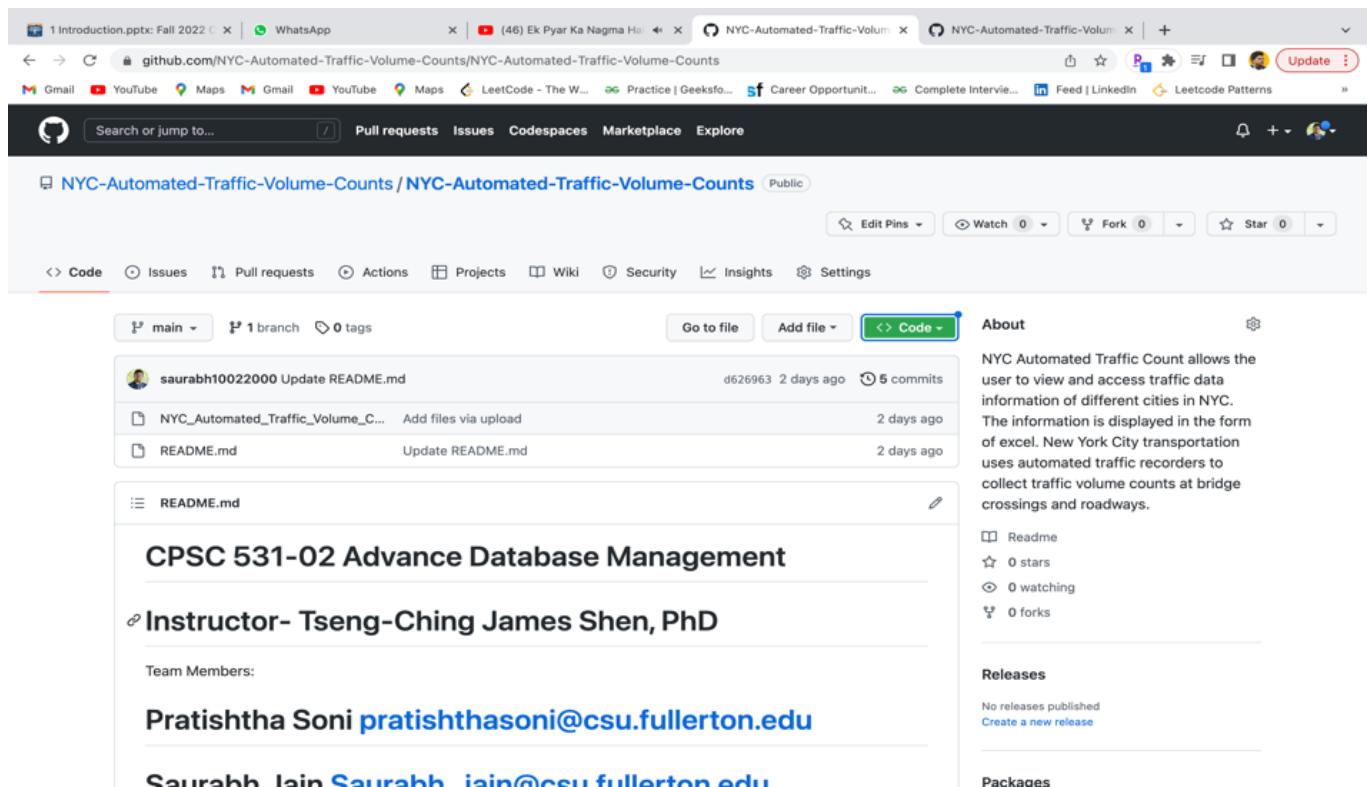


GitHub LOCATION OF CODE

Project Location: <https://github.com/NYC-Automated-Traffic-Volume-Counts/NYC-Automated-Traffic-Volume-Counts.git>.

Steps:

1. Download “NYC_Automated_Traffic_Volume_Counts.ipynb”.
2. Download Data from Kaggle (3GB)
<https://www.kaggle.com/datasets/aadimator/nyc-automated-traffic-volume-counts>
3. Import Data and project files to DataProc or on a Local Machine.
4. Run



The screenshot shows a GitHub repository page for "NYC-Automated-Traffic-Volume-Counts". The repository is public and has 5 commits in the main branch. The README.md file contains the following content:

```
CPSC 531-02 Advance Database Management
Instructor- Tseng-Ching James Shen, PhD
Team Members:
Pratishtha Soni pratishthasoni@csu.fullerton.edu
Saurabh Jain Saurabh\_jain@csu.fullerton.edu
```

The repository has 0 stars, 0 forks, and no releases published.

DEPLOYMENT INSTRUCTIONS

Steps to Set-up Spark in Local Machine:

Install JDK

```
!apt-get install openjdk-8-jdk-headless -qq > /dev/null
```

Get Spark installer (Check the path on Spark.org)

```
!wget -q https://archive.apache.org/dist/spark/spark-2.4.3/spark-2.4.3-bin-hadoop2.6.tgz
```

Check if the file is copied

```
!ls
```

Untar the Spark installer

```
!tar -xvf spark-2.4.3-bin-hadoop2.6.tgz
```

Check the spark folder after untar

```
!ls
```

Install findspark - a Python library to find Spark

```
!pip install -q findspark
```

Set environment variables

Set Java and Spark Home based on location where they are stored

```
import os  
os.environ["JAVA_HOME"] = "/usr/lib/jvm/java-8-openjdk-amd64"  
os.environ["SPARK_HOME"] = "/content/spark-2.4.3-bin-hadoop2.6"
```

Create a local Spark Session

```
import findspark  
findspark.init()  
from pyspark.sql import SparkSession  
spark = SparkSession.builder.master("local[*]").getOrCreate()
```

Test Installation

```
df = spark.createDataFrame([{"Google": "Colab", "Spark": "Scala"}, {"Google": "Dataproc", "Spark": "Python"}])  
df.show()
```

STEPS TO RUN THE APPLICATION

1. Create a Bucket and Upload Data.

- In the Google Cloud console, go to the Cloud Storage Buckets page. Go to Buckets.
- Click Create bucket.
- On the Create a bucket page, enter your bucket information. To go to the next step, click Continue. For Name, your bucket, enter a name that meets the bucket name requirements.
- Click Create.
- Upload data and Use the Access link to import data from Bucket to SSH Console.

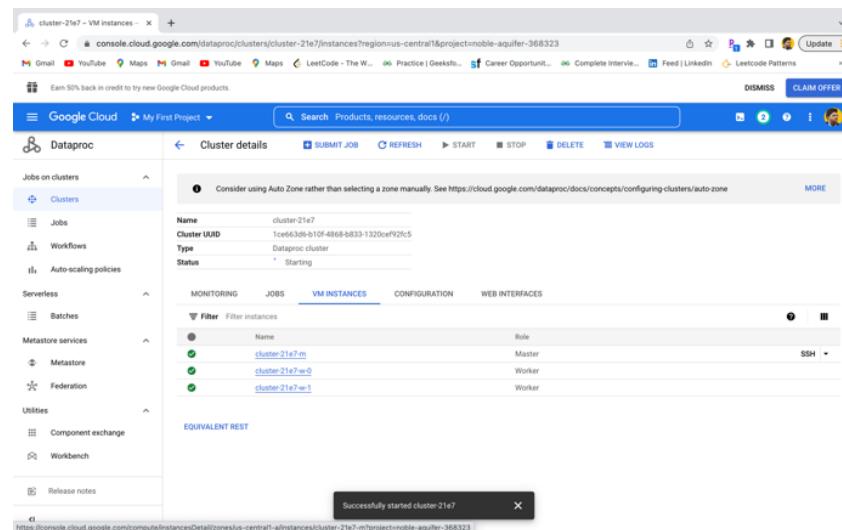
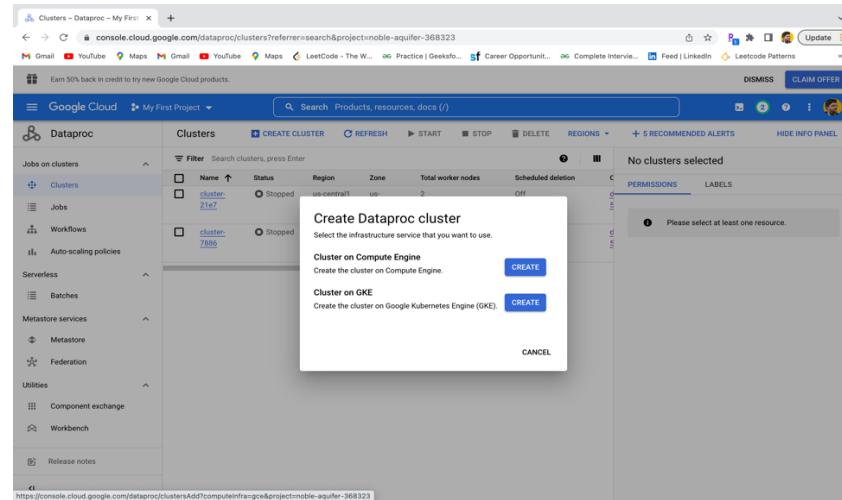
The screenshots illustrate the process of creating a bucket and uploading data to Google Cloud Storage. The top image shows the main Cloud Storage dashboard with several buckets listed. The bottom image provides a detailed view of a specific file's metadata, including download and edit options.

Object details for Automated_Traffic_Volume_Counts.csv

| Overview | Value |
|-------------------|---|
| Type | text/csv |
| Size | 3.1 GB |
| Created | 4 Dec 2022, 21:24:18 |
| Last modified | 4 Dec 2022, 21:24:18 |
| Storage class | Standard |
| Custom time | — |
| Public URL | Not applicable |
| Authenticated URL | https://storage.cloud.google.com/nytraffic/Automated_Traffic_Volume_Counts.csv |
| gsutil URI | gs://nytraffic/Automated_Traffic_Volume_Counts.csv |
| Permission | Public access |
| Protection | Not public |
| Hold status | None |
| Version history | None |
| Retention policy | None |
| Encryption type | Google-managed key |

2. Open DataProc Services and create clusters.

- In the Google Cloud console, go to the Dataproc Clusters page.^[1][GO TO CLUSTERS](#)
- Click Create cluster.
- In the Create Dataproc cluster dialog, click Create in the Cluster on Compute engine row.
- In the Cluster Name field, enter example-cluster.
- In the Region and Zone lists, select a region and zone
- For all the other options, use the default settings.
- To create the cluster, click Create.^[2] Your new cluster appears in a list on the Clusters page. The status is Provisioning until the cluster is ready to use, and then the status changes to Running. Provisioning the cluster might take a couple of minutes.



3. SSH Terminal / Jupyter Notebook to Execution

- Start pyspark.
 - customerDF = spark.read.csv("write address of bucket",header=True)
customerDF.show()
 - write spark sql queries.
 - Use this command to save output in bucket
db.write.save("write address of bucket ",format='csv',header=True)

Index (1,603) - saurabhjanin | WhatsApp | cluster-7886 - Monitoring | https://ssh.cloud.google.com/ (25) Using PySpark on Data | Cloud Monitoring | Datastream | +

SSH-in-browser

UPLOAD FILE DOWNLOAD FILE

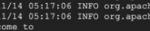
Linux cluster-7886-m 5.10.0-0.debian10.16-amd64 #1 SMP Debian 5.10.127-2-bpo1+ (2022-07-28) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/ <program>.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Copyright © 2022, The OpenSUSE Project e.V. 04/11/2022 from 35.235.244.34
saurabhjanin@cluster-7886-m:~\$ wget gs://nyctraffice/Automated_Traffic_Volume_Counts.csv: Unsupported scheme "gs".
saurabhjanin@cluster-7886-m:~\$ pyspark

Python 3.8.13 (default, Mar 25 2022, 06:04:10)
(v3.8.13:6e88c47, Mar 25 2022, 06:04:10)
Type "help", "copyright", "credits" or "license" for more information.
Setting default log level to "WARN".
To change the log level please use setLogLevel(newLevel). For Sparkr, use setLogLevel(newLevel).
22/11/14 05:17:06 INFO org.apache.spark.SparkEnv: Registering MapOutputTracker
22/11/14 05:17:06 INFO org.apache.spark.SparkEnv: Registering BlockManagerMaster
22/11/14 05:17:06 INFO org.apache.spark.SparkEnv: Registering BlockManagerMasterHeartbeat
22/11/14 05:17:06 INFO org.apache.spark.SparkEnv: Registering OutputCommitCoordinator

Welcome to

 version 3.1.3

Using Python version 3.8.13 (default, Mar 25 2022 06:04:10)
Spark context Web UI available at http://cluster-7886-m.us-central1-f.c.noble-aquifer-368323.internal:46159
Spark context available as `sc` (master = yarn, app id = application_1668401173550_0002).
SparkSession available as `spark`.

```
>>> trips = spark.read.csv("gs://nyctraffice/Automated_Traffic_Volume_Counts.csv",header=True,inferSchema=True)
>>> trips.show(10)
```

| [RequestID] | Boro | Yr | Wk | D | HR | Min | SegmentID | WKTGeom | street | fromSt | toSt | direction | |
|-------------|-------|---------------------|-----|-----|----|-----|-----------|---------|----------------------|--------------------|-----------------------|----------------------|----|
| 1 | 20856 | Queens[2015] | 6 | 23 | 23 | 30 | 9 | 171694 | POINT (1052296.60... | 94 AVENUE | 207 Street | Francis Lewis Bou... | WB |
| 1 | 21231 | Staten Island[2015] | 9 | 18 | 4 | 15 | 6 | 99694 | POINT (46268.585... | RICHMOND TERRACE | West 2nd Avenue | Emmett Court | WB |
| 1 | 20709 | Brooklyn[2017] | 111 | 78 | 30 | 168 | 98 | 188023 | POINT (992925.43... | HUNTS POINT AVENUE | Whittier Street | Randall's Island | NB |
| 1 | 26734 | Manhattan[2017] | 111 | 78 | 11 | 78 | 109 | 137516 | POINT (104175.95... | FLATBUSH AVENUE | Brooklyn Line | Brighton Line | NB |
| 1 | 26019 | Bronx[2017] | 6 | 17 | 17 | 45 | 11 | 86333 | POINT (102174.47... | WASHINGTON BRIDGE | Harlem River Short... | Harlem River Shore | EB |
| 1 | 23041 | Manhattan[2017] | 111 | 18 | 18 | 18 | 109 | 70833 | POINT (104175.95... | WALLACE AVENUE | Rhinelander Avenue | Brondale Avenue | NB |
| 1 | 23133 | Queens[2015] | 3 | 21 | 9 | 45 | 23 | 101101 | POINT (1050277.33... | S/9 AMERICA AVENUE | EDWARD MURKIN AV/W... | 131ST ST | SB |
| 1 | 32417 | Queens[2020] | 111 | 141 | 21 | 15 | 181 | 147877 | POINT (1044172.66... | NORTHERN BOULEVARD | 220 Place | 220 Street | WB |
| 1 | 26198 | Bronx[2017] | 6 | 22 | 41 | 30 | 2 | 85935 | POINT (1021747.23... | MIDLAND PARKWAY | Dalny Road | Connector | SB |
| | | | | | | | | | THIERIOT AVENUE | Gleason Avenue | Pelham Line | NB | |

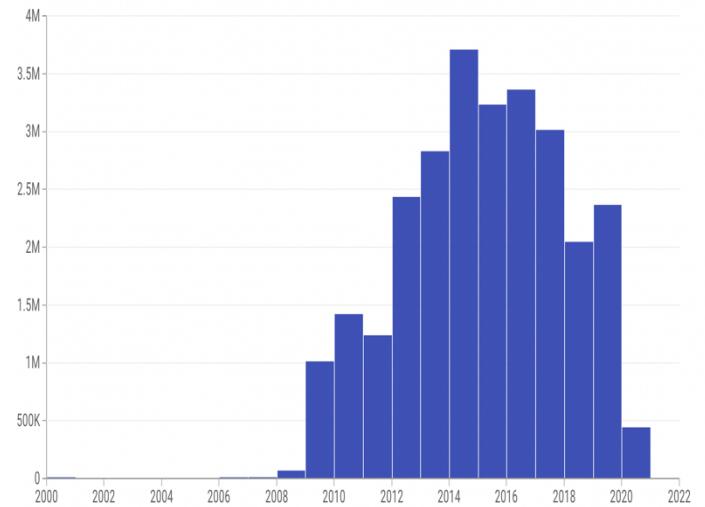
only showing top 10 rows

| SSH-in-browser | | | | | | | | | | | UPLOAD FILE | | |
|--|-----------------|------|-----|-----|-----|--------|---------|-------------------------|-------------------------|-----------------------------|----------------------|-------------------|------------|
| RequestID | Boro | Yr | M | D | HH | MM | Vol | SegmentID | WktGeom | Street! | fromSt! | toSt! | Direction! |
| 1 | Manhattan[2000] | 1 | 51 | 71 | 30 | 2161 | 32956 | POINT (986884.7 2...) | W/B UNION SQUARE ... | E 14 ST/BROADWAY | 4 AVI | NB | |
| 7625 | Queens[2007] | 4 | 51 | 23 | 01 | 1481 | 156596 | POINT (1023975.5 7...) | LINDEN BLVD | 78 ST | AMBER ST | NB | |
| 7627 | Brooklyn[2008] | 4 | 241 | 51 | 07 | 924 | 141367 | POINT (1013806.1 7...) | JACKIE ROSENBLIT PVY | MILLER AV | VERMONT ST | NB | |
| 7628 | Bronx[2008] | 4 | 131 | 122 | 01 | 45 | 103033 | POINT (1020551.5 7...) | CODNER ST | WYCKOFF AV | CYPRESS AV | NB | |
| 7624 | Queens[2007] | 4 | 51 | 41 | 01 | 41 | 190387 | POINT (1023782.6 7...) | LINDEN BLVD | 78 ST | AMBER ST | NB | |
| 3442 | Bronx[2008] | 4 | 131 | 81 | 01 | 1641 | 88874 | POINT (1020685.6 7...) | S/B VAN CORTLANDT ... | E 242 ST/KATONAH AV | KATONAH AVENUE | SBN | |
| 3473 | Queens[2008] | 6 | 51 | 111 | 01 | 3291 | 13892 | POINT (1060197.4 7...) | E/B LAURELTON 9 C... | DEAD END/BELT PKWY BEX EXIT | E/B | EB | |
| 7489 | Queens[2008] | 5 | 261 | 151 | 45 | 7591 | 153405 | POINT (1004881.5 7...) | O/B KOSCZUSKO BR... | 54 RD | EQE BE EX 35 F-W | EB | |
| 7499 | Queens[2008] | 3 | 171 | 101 | 49 | 1191 | 156209 | POINT (1066765.1 7...) | E/B UNION TURNK... | LANGDALE ST | LANGDALE ST | EB | |
| 3479 | Manhattan[2000] | 4 | 171 | 22 | 151 | 07 | 2614 | 109492 | POINT (99940.4 7...) | E/B UNION TURNK... | LANGDALE ST | LANGDALE ST | EB |
| 7521 | Queens[2008] | 3 | 171 | 111 | 101 | 101 | 109305 | POINT (1023701.9 7...) | E/B MERRICK BLVD | 244 ST | 243 ST | EB | |
| 7463 | Bronx[2008] | 5 | 131 | 81 | 45 | 694 | 190807 | POINT (1031782.6 7...) | N/B HUTCHINSON RI... | NEW ENGLAND THRWY | NEW ENGLAND THRWY | NB | |
| 7464 | Bronx[2008] | 5 | 181 | 171 | 301 | 01 | 154955 | POINT (1033736.9 7...) | N/B HUTCHINSON RI... | HUTCHINSON RIVER ... | HUTCHINSON RIVER ... | NB | |
| 3475 | Manhattan[2008] | 7 | 231 | 141 | 151 | 91 | 23255 | POINT (981449.3 2...) | E/B CHAMBERS ST 9 ... | HUDSON ST/W BROADWAY | GREENWICH ST | EB | |
| 7600 | Bronx[2008] | 5 | 51 | 11 | 01 | 202699 | 13525 | POINT (1015681.1 7...) | S/B HENRY HUDSON ... | DEAD END | MOSHULU PKWY | EB | |
| 3434 | Bronx[2008] | 6 | 231 | 22 | 01 | 274 | 85879 | POINT (1038494.4 7...) | S/B NEW ENGLAND T... | CONNECTOR | DEAD END | SBN | |
| 3450 | Brooklyn[2008] | 5 | 101 | 151 | 30 | 106 | 105839 | POINT (987431.6 7...) | S/N 3 AVE BRIDGE | 3 ST | 6 ST | NB | |
| 7464 | Bronx[2008] | 5 | 71 | 161 | 30 | 1066 | 154955 | POINT (1053736.9 7...) | N/B HUTCHINSON RI... | HUTCHINSON RIVER ... | HUTCHINSON RIVER ... | NB | |
| only showing top 20 rows | | | | | | | | | | | | | |
| >>> new_results = spark.sql("select * from Traffic order by Yr DESC").show() | | | | | | | | | | | | | |
| +-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+ | | | | | | | | | | | | | |
| RequestID Boro Yr M D HH MM Vol SegmentID WktGeom Street! fromSt! toSt! Direction! | | | | | | | | | | | | | |
| +-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+ | | | | | | | | | | | | | |
| f0 fs24 | vcf0 Reg.. | Boro | Yr | M | D | HH | MM | Vol SegmentID | WktGeom | street! | fromSt! | toSt! | Direction! |
| 32384 | Manhattan[2020] | 10 | 20 | 11 | 30 | 81 | 1 | 158808 | POINT (989240.431...) | BROADWAY | West 60 Street | West 61 Street | NB |
| 32407 | Bronx[2020] | 10 | 19 | 11 | 01 | 14040 | 69362 | POINT (1095248.9 7...) | EAST 149 STREET | Park Avenue | Morris Avenue | SB | |
| 32384 | Manhattan[2020] | 10 | 20 | 11 | 30 | 81 | 1 | 158808 | POINT (989240.431...) | BROADWAY | 8 Avenue Line | 8 Avenue Line | NB |
| 32417 | Queens[2008] | 10 | 161 | 111 | 151 | 22 | 12370 | POINT (1033038.18 7...) | QUEENS BOULEVARD | Dead End | Dead End | NB | |
| 32407 | Bronx[2020] | 10 | 131 | 31 | 30 | 101 | 184562 | POINT (1085647.74 7...) | GRAND CONCOURSE | Concourse Line | Concourse Line | SB | |
| 32417 | Queens[2008] | 10 | 111 | 81 | 45 | 991 | 1567373 | POINT (1041559.46 7...) | UTOPIA PARKWAY | 65 Avenue | 67 Avenue | SB | |
| 32417 | Queens[2008] | 10 | 131 | 151 | 30 | 1941 | 1505493 | POINT (1031410.3 7...) | CROSS BAY BOULEVARD | Dead End | Connector | NB | |
| 32384 | Manhattan[2020] | 10 | 28 | 11 | 30 | 1951 | 34338 | POINT (987878.166 7...) | AMSTERDAM AVENUE | West 60 Street | West 61 Street | NB | |
| 32407 | Bronx[2020] | 10 | 13 | 81 | 22 | 1201 | 78870 | POINT (1031440.7 7...) | WESTCHESTER AVENUE | East 148 Street | Home Street | WB | |
| 32407 | Bronx[2020] | 10 | 111 | 181 | 111 | 22 | 1201 | 104400 | POINT (1031440.7 7...) | MERRICK BOULEVARD | 11th Street | End | SB |
| 32406 | Queens[2008] | 10 | 111 | 81 | 111 | 01 | 1 | 171277 | POINT (1051766.2 7...) | BEACH 14 STREET | Harrison Road | Seagirt Boulevard | SB |
| 32406 | Queens[2008] | 9 | 81 | 111 | 45 | 531 | 91184 | POINT (1031689.11 7...) | KISSENA BOULEVARD | 41 Avenue | Barclay Avenue | EB | |
| 32417 | Queens[2008] | 10 | 111 | 221 | 191 | 151 | 61 | 150539 | POINT (1038323.08 7...) | 164 STREET | 65 Avenue | 67 Avenue | SB |
| 32417 | Queens[2008] | 20 | 17 | 22 | 30 | 751 | 57192 | POINT (1038292.60 7...) | LIBERTY AVENUE | Waltham Street | 146 Street | SB | |
| 32407 | Bronx[2020] | 10 | 18 | 41 | 45 | 341 | 72026 | POINT (1069749.58 7...) | JEROME AVENUE | Goble Place | East Mt Eden Avenue | NB | |
| 32407 | Bronx[2020] | 10 | 181 | 111 | 51 | 57 | 10376 | POINT (1069749.58 7...) | UNION AVENUE | Hewlett Avenue | East 155 Street | SB | |
| 32407 | Bronx[2020] | 10 | 201 | 111 | 51 | 57 | 10376 | POINT (1069811.13 7...) | 3 NARROWS | 1 North East 154 Street | East 155 Street | SB | |
| 32384 | Manhattan[2020] | 10 | 27 | 21 | 01 | 271 | 173249 | POINT (102839.60 7...) | 145 STREET BRIDGE | Dead End | Harlem River Shore | EB | |
| 32407 | Bronx[2020] | 10 | 151 | 131 | 01 | 751 | 121069 | POINT (1013077.48 7...) | CROTONE AVENUE | Dead End | Crotone Park North | SB | |

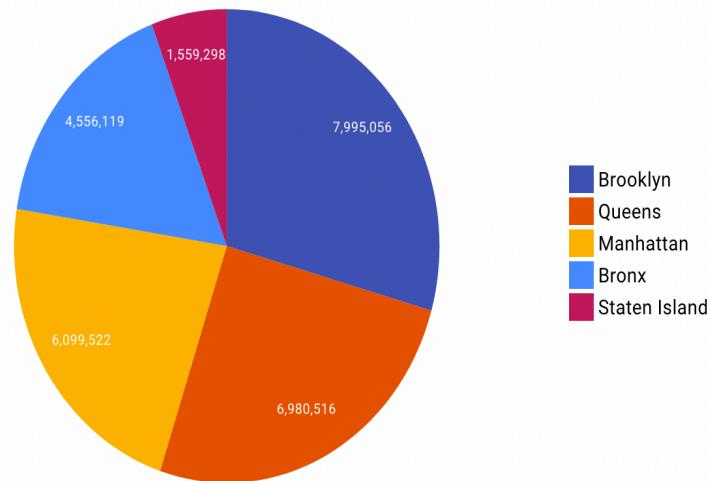
TEST RESULT

Data visualization Using NYC-Automated-Traffic-Volume-Counts

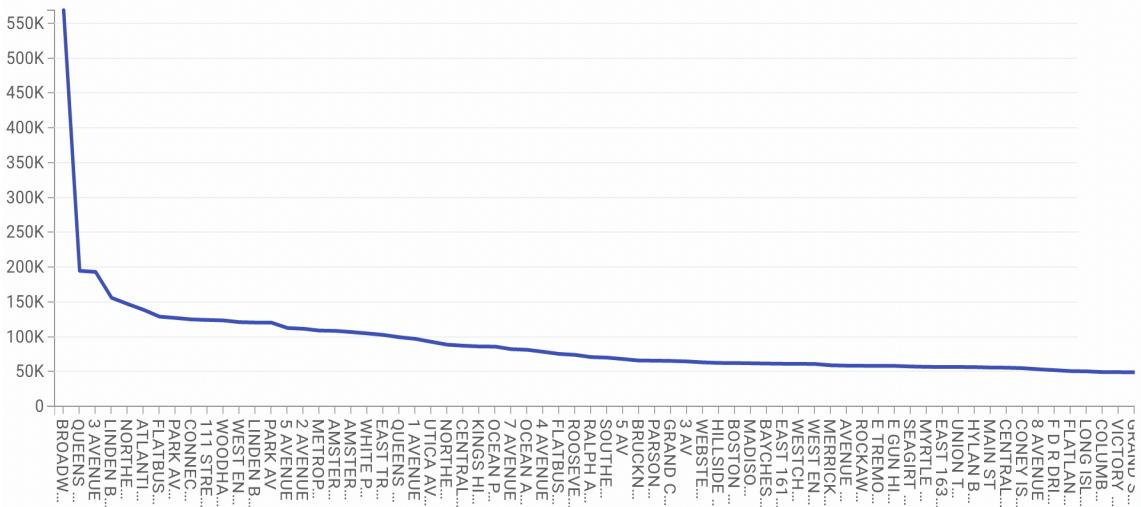
- Variation in Volume of data/vehicle: After pre-processing, the main analysis of this traffic dataset shows the number of increases in the volume of cars from the year 2000 to 2020. (Queries being displayed in the implementation section)



- Busiest Cities of NYC: After pre-processing and running queries and we extracted the top 3 busiest cities and, it found that Brooklyn is the most crowded and busiest city with the count of vehicles (7,995,056, 29%), Queens being the 2nd busiest (6,980,516, 26%) and Manhattan being the 3rd busiest (6,099,052, 22%).

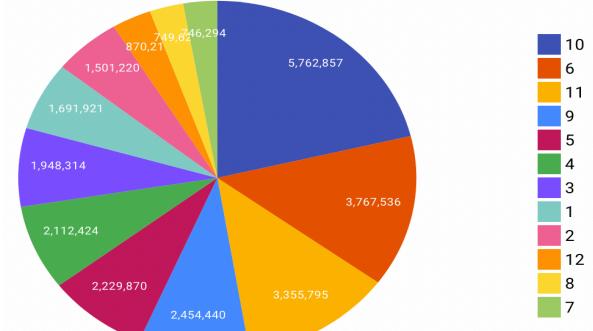


- Most visited streets: From the dataset, we found the most visited and busiest street in different cities in NYC.



- The volume of car increased every 15 mins in New York: This query displays how many numbers of cars increased city of Boro.
- Grouped according to Busiest Month in the year of different Cities: By applying group by, sorting, and count query we found that October was the busiest month in Manhattan in the entire year.

- 1st pie chart shows the number distribution according to month.



- 2nd pie chart shows the busiest city according to the month.

