



User Guide for Engineers

Envisioning Yellow Taxi Demand in NYC City



Submitted by: Simranjeet Randhawa

Student ID: 200412297

Table of Contents

1. Synopsis.....	1
2. Language and tools used:	1
3. Libraries Used:.....	2
4. Step up the project from scratch	2
4.1 Create Service Account on GCP.....	2
4.2 Setting up the credentials	3
4.3 Cloning the Github Repository	3
4.4 How to Run.....	3
4.5 Deploying on GCP (Google Cloud Platform)	5
5. Steps to set up Dashboards:	6
6. Additional notes to engineers:.....	6
7. Contact Information	7

1. Synopsis

The Envisioning Yellow Taxi demand in NYC is a web application that is very useful for all the yellow cab taxi driver to find the number of pickups in a particular area. Also, it is helpful for the TLC to monitor the yellow cabs.

This user guide is for engineering. It will illustrate how the project code can be run by setting up the platform and installing all the libraries required.



Figure: 1. Flask plus Google Cloud

2. Language and tools used:

- Python: The version that I used was 3.7.



Python 3.7



Big Query



Data Studio

Figure: 2. Tools Used

- Big Query: It offered to query petabytes of data at high speed.
- Google Cloud Platform: It was used to store the data.
- PySpark: Because it offers support for lazy computation.
- Jupyter iPython Notebook: Makes the code easier to understand and illustrate it in a storytelling way.

- Some other tools that the engineer needs to be aware of are as follows:
- Google Data Studio: To prepare the dashboards.
- App engine: Used to deploy the model
- Flask: Created a flask application to build a website.

3. Libraries Used:

Flask==1.1.0	gunicorn==19.6.0
pandas==0.22.0	numpy==1.11.2
scipy==0.18.1	scikit-learn>=0.18
lightgbm==2.3.1	beautifulsoup4==4.8.0
requests==2.22.0	requests-oauthlib==1.3.0
google-cloud-bigquery==1.24.0	

The above is required to run the Flask application. But additional libraries used are illustrated in the notebook phase. But that is illustrated in the notebook itself.

4. Step up the project from scratch

4.1 Create Service Account on GCP

1. Make a Google Account.
2. Log in to Google Cloud Platform.

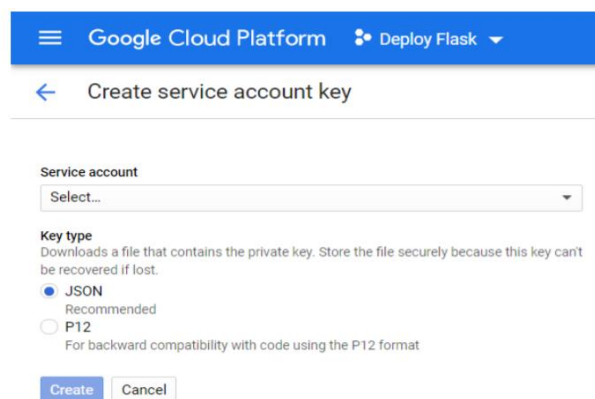


Figure 3. Google Cloud Platform

3. Setup the cloud credential on google platform. Note: It offers 500 dollars free credit so that can be good to test and deploy your model form months if you use minimum clusters and machines.

4.2 Setting up the credentials

1. Create a Service account. But note that it requires a credit card at later stages. But the first 500 dollars will be free credited so you can utilize it to deploy the model.
2. It can be created using the following link:
[Service Account Link for Google Cloud Platform](#)
3. This can be done using the following link: Service Account Page
4. The new service account can be set up from there. Select create a service account after entering the name.
5. The role can be to distribute the service account among others but prefer the owner option.
6. When you click create then a JSON file is created that contains the credentials.

4.3 Cloning the Github Repository

1. After setting up, Visit my GitHub Repositories for the project:
<https://github.com/ssrbazpur/Envisioning-Yellow-Taxi-High-Demand-Areas-in-NYC-city>
2. Clone my Github Repository.

4.4 How to Run

1. The repository contains the four notebooks that are named as :
 - Data_Discovery.ipynb
 - Data_Preparation.ipynb
 - Model_Planning.ipynb
 - Model_Building.ipynb

Now you have to start with the first notebook. Open the notebook in Google Colab and then run each cell. I have also added additional notes on each cell mentioning the cell. The steps should be Data Discovery -> Data Preparation -> Model Planning -> Model Building.

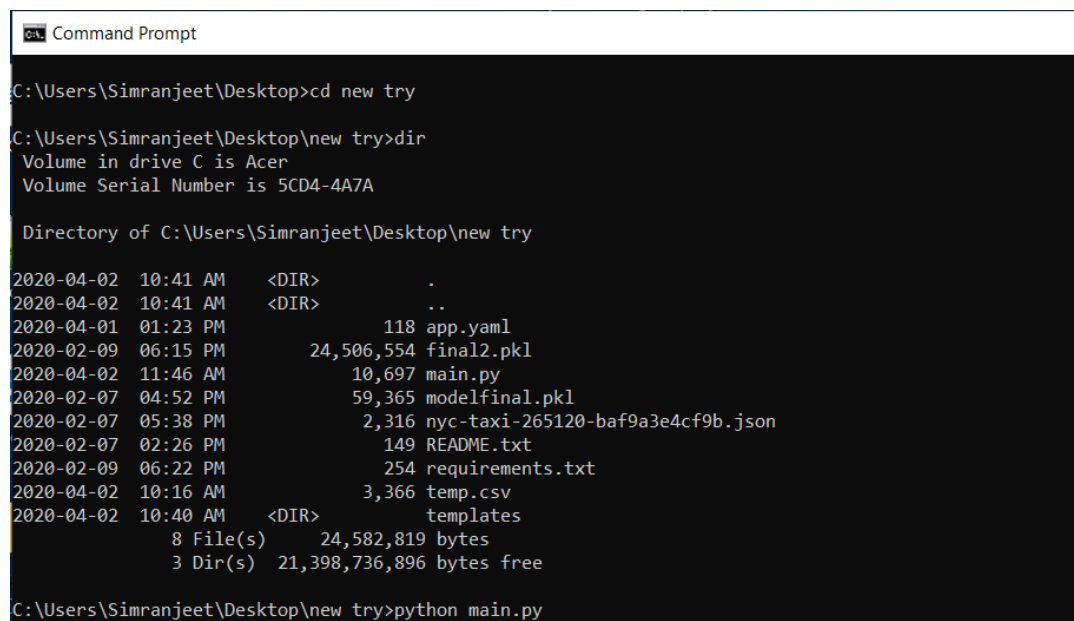
Once all the notebooks are executed then the engineers can get the trained model. (Note: I bought additional RAM to train the model. But I believe most parts without GridSearch CV can be executed on the Colab 25 GB cluster of RAM.

2. After getting the trained model, its time to switch to a Flask app that is in the Operationalize folder.
3. To run the Flask app, install the requirement:

```
Flask==1.1.0                gunicorn==19.6.0
pandas==0.22.0              numpy==1.11.2
scipy==0.18.1               scikit-learn>=0.18
lightgbm==2.3.1             beautifulsoup4==4.8.0
requests==2.22.0            requests-oauthlib==1.3.0
google-cloud-bigquery==1.24.0
```

Once the requirements are installed. Please make sure that the folder path is not messed up in previous steps. The .pkl file should be placed in the same Flask app folder.

4. To run on the local machine the command are as follows:
 - Cd to the directory where the main.py file is placed.
 - Python main.py



```

C:\Users\Simranjeet\Desktop>cd new try
C:\Users\Simranjeet\Desktop\new try>dir
Volume in drive C is Acer
Volume Serial Number is 5CD4-4A7A

Directory of C:\Users\Simranjeet\Desktop\new try

2020-04-02  10:41 AM    <DIR>          .
2020-04-02  10:41 AM    <DIR>          ..
2020-04-01  01:23 PM             118 app.yaml
2020-02-09  06:15 PM    24,506,554 final2.pkl
2020-04-02  11:46 AM     10,697 main.py
2020-02-07  04:52 PM     59,365 modelfinal.pkl
2020-02-07  05:38 PM     2,316 nyc-taxi-265120-baf9a3e4cf9b.json
2020-02-07  02:26 PM      149 README.txt
2020-02-09  06:22 PM      254 requirements.txt
2020-04-02  10:16 AM     3,366 temp.csv
2020-04-02  10:40 AM    <DIR>          templates
               8 File(s)      24,582,819 bytes
               3 Dir(s)  21,398,736,896 bytes free

C:\Users\Simranjeet\Desktop\new try>python main.py
  
```

4.5 Deploying on GCP (Google Cloud Platform)

This will run the web application on the localhost. The next steps include deploying it on the cloud. You can use Heroku, AWS or Google Cloud. I used Google Cloud.

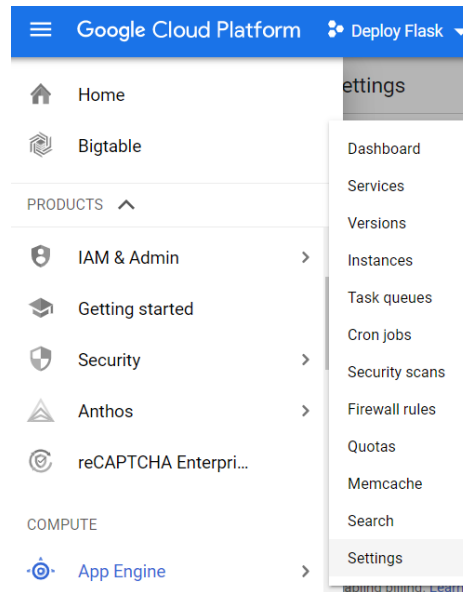
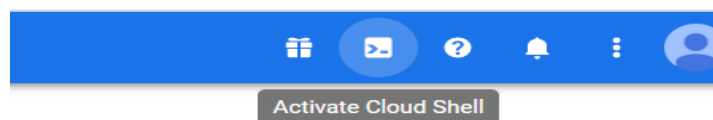
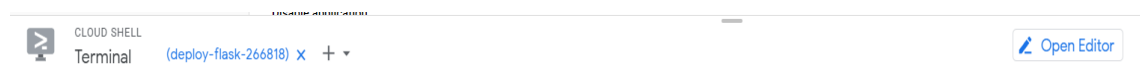


Figure 4. App Engine

1. App Engine on the google cloud platform can do the job:
 - First, enable the application to use the App Engine. Setting → Enable
 - Select Activate Cloud Shell as the screenshot shows:



- Open Editor as shown:



- Copy the Flask app directory to the google cloud platform.
- Change directory to the main.py file folder.
- The command that deploys the model:

```
$ gcloud app deploy
```

- Congrats, the app is deployed on the cloud as well as the model used.
2. Finally, the whole project can be seen live on <https://deploy-flask-266818.appspot.com/>

5. Steps to set up Dashboards:

I have made both the dashboards as public but to edit it you are required to clone it and change the data source to the source that was set up using the Google Cloud Credentials Obtained:

1. Weekly Forecasting Demand Dashboard :

The below link shows the weekly forecasting dashboard:

<https://datastudio.google.com/reporting/e73c735b-cf23-48ab-9c02-0563df4fa205?s=vLH-KVdO6Ik>

2. 2018-19 Taxi Pickups Dashboard:

The below link opens the Taxi Pickups Dashboard that can be used by engineers and can be modified by cloning on Data Studios:

https://datastudio.google.com/reporting/41194526-9091-407c-8409-030039ccaf87/page/AquJB?s=pc-GN_BSq_M

6. Additional notes to engineers:

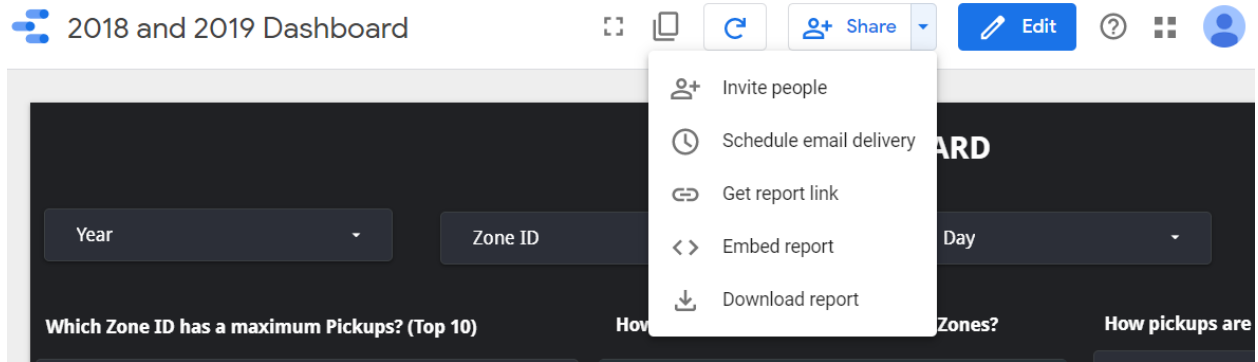
The engineers can also use the Read me provided on Github to understand more about the project:

<https://github.com/ssrbazpur/Envisioning-Yellow-Taxi-High-Demand-Areas-in-NYC-city/blob/master/README.md>

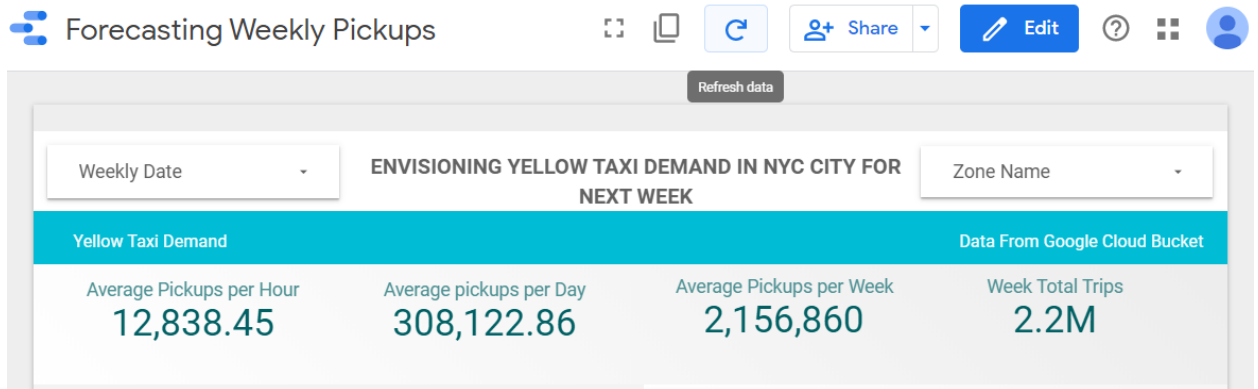
For the Dashboard, the engineers can schedule email delivery once the dashboard updates and also invite other team engineers to work on the dashboard.

Also, the engineers can follow the steps on my Github repository Readme to get the project setup if they face any technical difficulties while doing the steps indicated.

The below screenshot gives the sharing feature that can allow multiple engineers to run the dashboard.



Also, there are Refresh Data options that can be used as Data everyday changes as the model makes new predictions depending on the temperature and other features.



7. Contact Information

Email: ssrbazpur@gmail.com

-----X-----X-----