Building Serverless ML Platform

Data Analysis And Visualization Of Green Taxi Records

SRI HARSHA KORUKONDA BHATTAR (1953040)
AASHISH JOSHI (1886750)
ANKITH KANDALA (1894221)

Contents:

- Introduction
- Serverless Architecture
- Methodology
- Challenges
- Future work

INTRODUCTION

Building a serverless Machine Learning platform that analyses, visualizes Green-taxi user data and predicts future customer base.

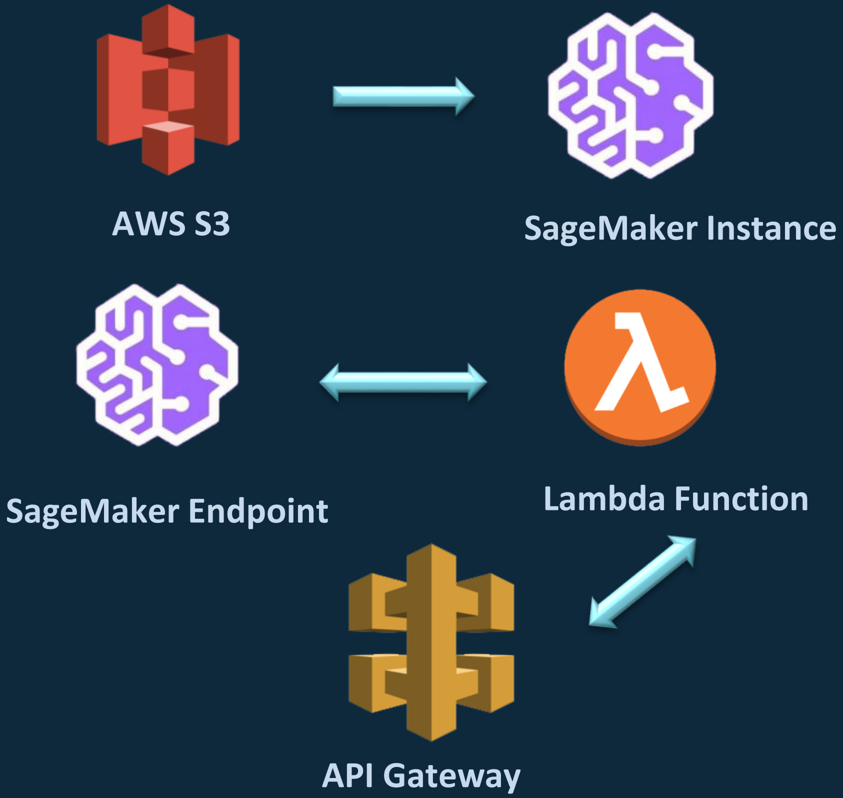
SERVERLESS ARCHITECTURE

- Serverless is the native architecture of the cloud that enables you to shift more of your operational responsibilities to AWS.
- Serverless enables us to build modern applications with increased agility and lower total cost of ownership.

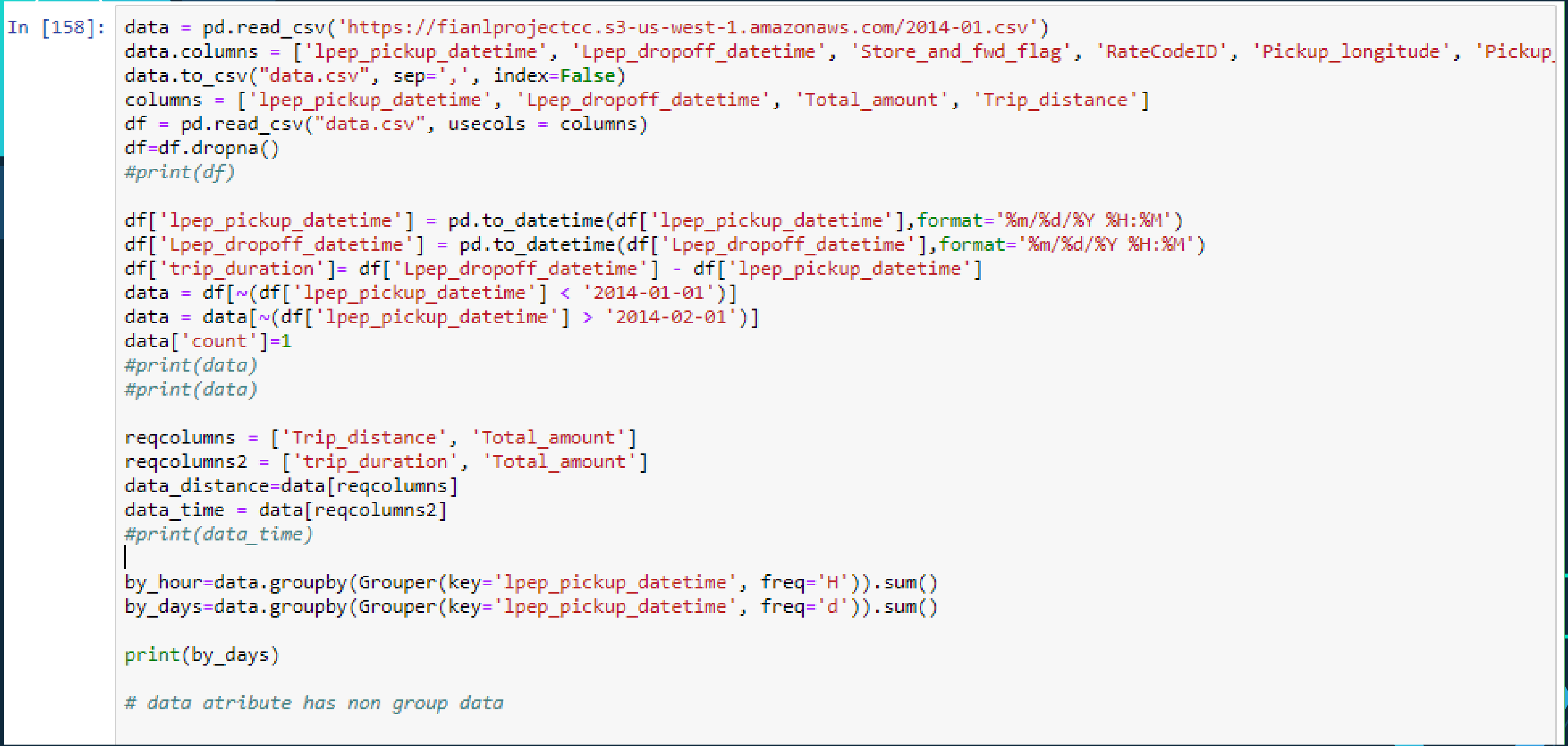
METHODOLOGY

AWS Tools used:

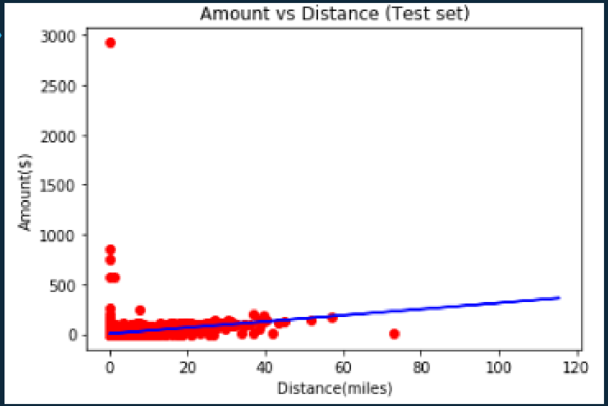
- Amazon S3
- Lambda function
- Sagemaker Instance
- Sagemaker Endpoint
- AWS API

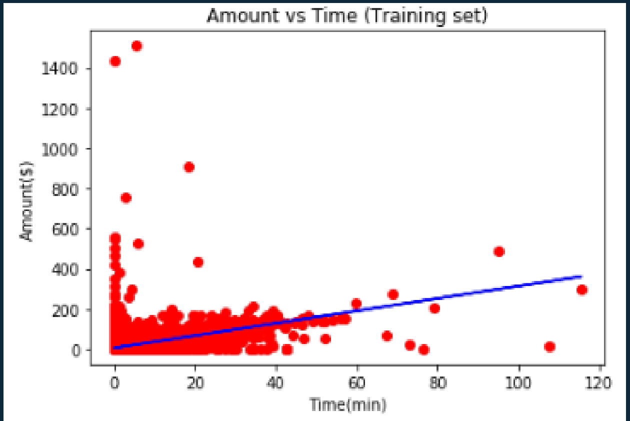


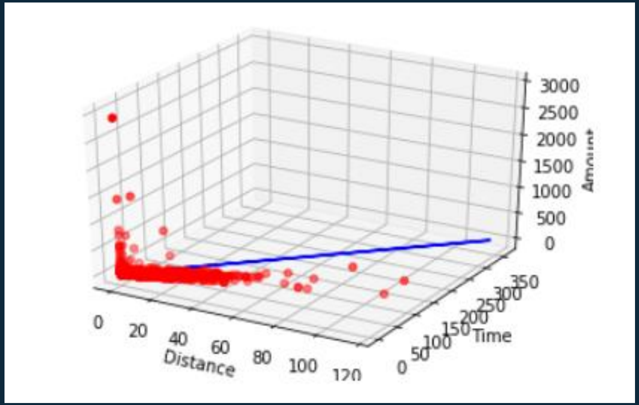
- 6 years Green taxi Trip records for the state of New York have been collected.
- Monthly Raw data files are stored in S3.
- We have implemented Python to organize and analyze the Raw data.
- ML Algorithms – Multiple Linear Regression.

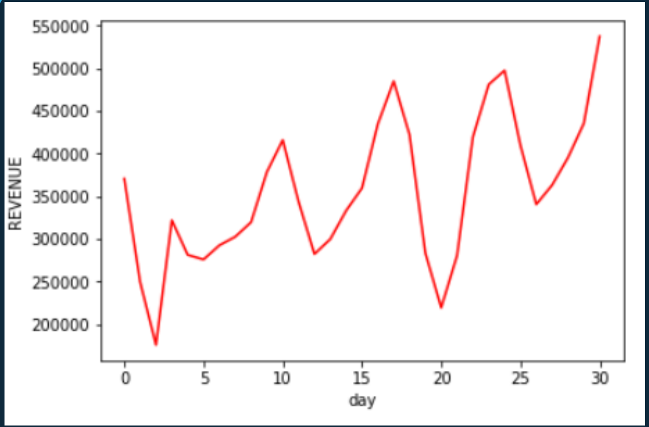


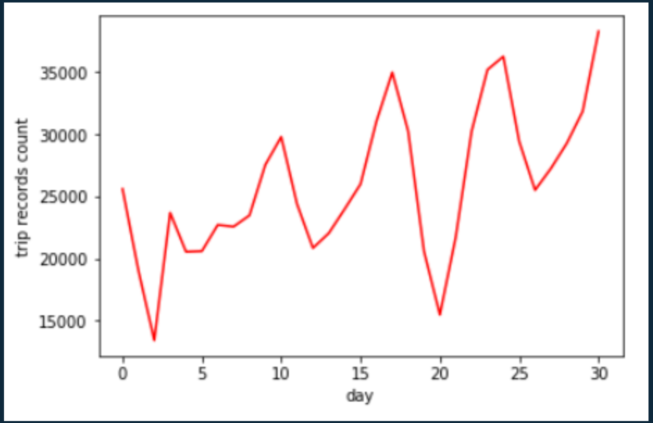
```
In [158]: data = pd.read_csv('https://fianlprojectcc.s3-us-west-1.amazonaws.com/2014-01.csv')
          data.columns = ['lpep_pickup_datetime', 'Lpep_dropoff_datetime', 'Store_and_fwd_flag', 'RateCodeID', 'Pickup_longitude', 'Pickup_
          data.to_csv("data.csv", sep=',', index=False)
          columns = ['lpep_pickup_datetime', 'Lpep_dropoff_datetime', 'Total_amount', 'Trip_distance']
          df = pd.read_csv("data.csv", usecols = columns)
          df=df.dropna()
          #print(df)

          df['lpep_pickup_datetime'] = pd.to_datetime(df['lpep_pickup_datetime'],format='%m/%d/%Y %H:%M')
          df['Lpep_dropoff_datetime'] = pd.to_datetime(df['Lpep_dropoff_datetime'],format='%m/%d/%Y %H:%M')
          df['trip_duration']= df['Lpep_dropoff_datetime'] - df['lpep_pickup_datetime']
          data = df[~(df['lpep_pickup_datetime'] < '2014-01-01')]
          data = data[~(df['lpep_pickup_datetime'] > '2014-02-01')]
          data['count']=1
          #print(data)
          #print(data)

          reqcolumns = ['Trip_distance', 'Total_amount']
          reqcolumns2 = ['trip_duration', 'Total_amount']
          data_distance=data[reqcolumns]
          data_time = data[reqcolumns2]
          #print(data_time)

          by_hour=data.groupby(Grouper(key='lpep_pickup_datetime', freq='H')).sum()
          by_days=data.groupby(Grouper(key='lpep_pickup_datetime', freq='d')).sum()

          print(by_days)

          # data atribute has non group data
```

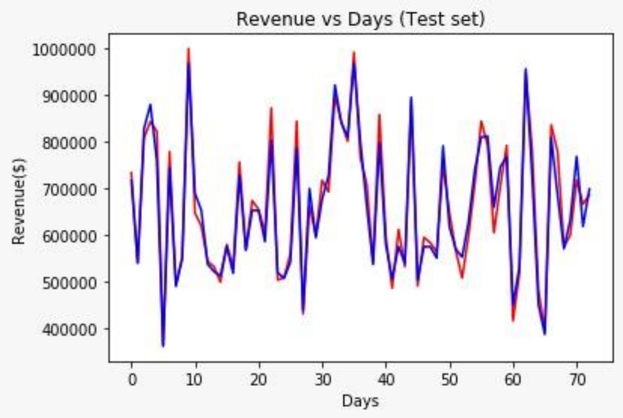

Amount vs Distance (Test set)
Amount($)
3000
2500
2000
1500
1000
500
0
0
20
40
60
80
100
120
Distance(miles)


Amount vs Time (Training set)
Amount($)
1400
1200
1000
800
600
400
200
0
0
20
40
60
80
100
120
Time(min)

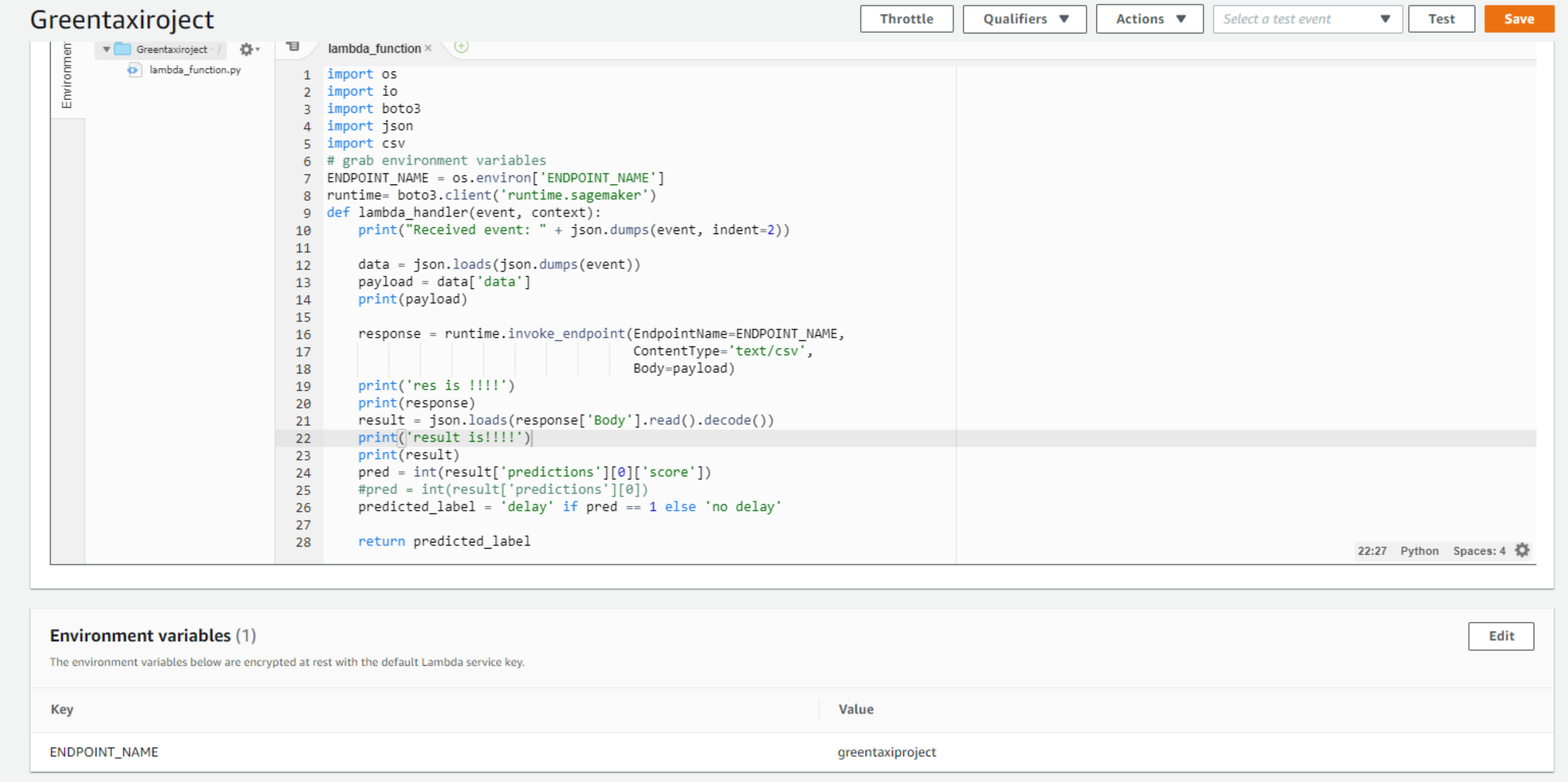

3000
2500
2000
1500
1000
500
0
Amount
0
20
40
60
80
100
120
Distance
0
50
100
150
200
250
300
350
Time


550000
500000
450000
400000
350000
300000
250000
200000
REVENUE
0
5
10
15
20
25
30
day


35000
30000
25000
20000
15000
trip records count
0
5
10
15
20
25
30
day


Revenue vs Days (Test set)
1000000
900000
800000
700000
600000
500000
400000
Revenue($)
0
10
20
30
40
50
60
70
Days

Prediction Accuracy for trip pricing – 92.4%
Prediction Accuracy for revenue and customer base – 86.8%

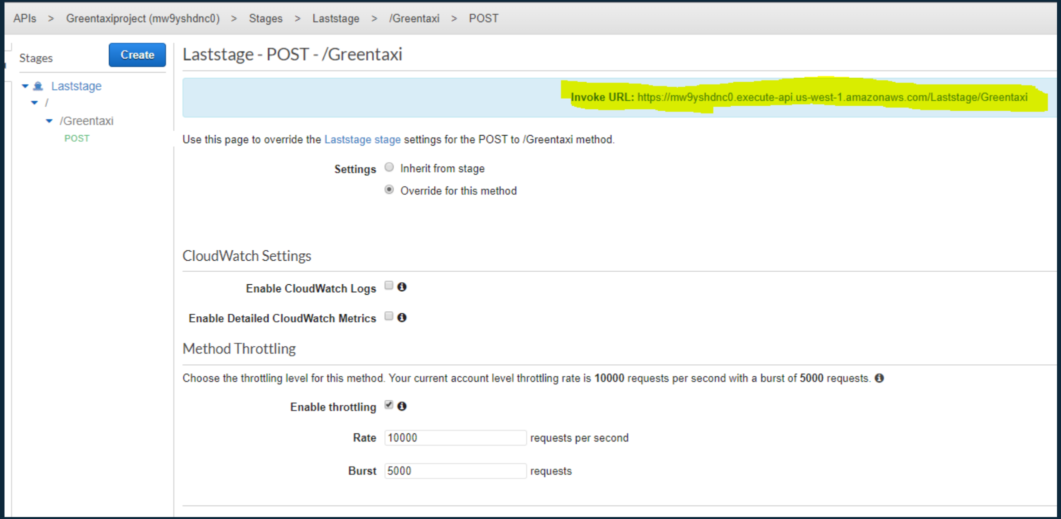


```
import os
import io
import boto3
import json
import csv
# grab environment variables
ENDPOINT_NAME = os.environ['ENDPOINT_NAME']
runtime= boto3.client('runtime.sagemaker')
def lambda_handler(event, context):
    print("Received event: " + json.dumps(event, indent=2))

    data = json.loads(json.dumps(event))
    payload = data['data']
    print(payload)

    response = runtime.invoke_endpoint(EndpointName=ENDPOINT_NAME,
                                       ContentType='text/csv',
                                       Body=payload)
    print('res is !!!!')
    print(response)
    result = json.loads(response['Body'].read().decode())
    print('result is!!!!')
    print(result)
    pred = int(result['predictions'][0]['score'])
    #pred = int(result['predictions'][0])
    predicted_label = 'delay' if pred == 1 else 'no delay'

    return predicted_label
```

Environment variables (1)

The environment variables below are encrypted at rest with the default Lambda service key.

Key	Value
ENDPOINT_NAME	greentaxiproject


APIs > Greentaxiproject (mw9yshdnc0) > Stages > Laststage > /Greentaxi > POST
Stages
Create
Laststage
/
/Greentaxi
POST

Laststage - POST - /Greentaxi

Invoke URL: https://mw9yshdnc0.execute-api.us-west-1.amazonaws.com/Laststage/Greentaxi

Use this page to override the Laststage stage settings for the POST to /Greentaxi method.

Settings ○ Inherit from stage
● Override for this method

CloudWatch Settings

Enable CloudWatch Logs □

Enable Detailed CloudWatch Metrics □

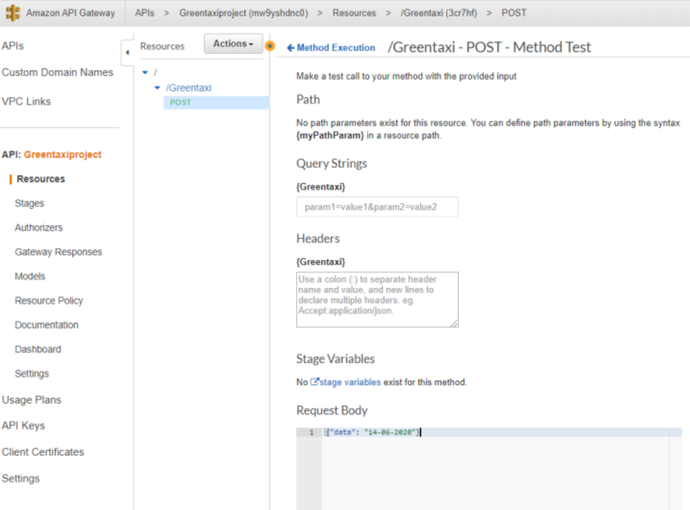
Method Throttling

Choose the throttling level for this method. Your current account level throttling rate is **10000** requests per second with a burst of **5000** requests.

Enable throttling ☑

Rate 10000 requests per second

Burst 5000 requests



/Greentaxi - POST - Method Test

Make a test call to your method with the provided input

Path

No path parameters exist for this resource. You can define path parameters by using the syntax **{myPathParam}** in a resource path.

Query Strings

{Greentaxi}

param1=value1&param2=value2

Headers

{Greentaxi}

Use a colon (:) to separate header name and value, and new lines to declare multiple headers. eg. Accept:application/json.

Stage Variables

No stage variables exist for this method.

Request Body

```
{"data": "14-06-2020"}
```

CHALLENGES

- Column names and data types were different for different monthly reports so preprocessing the data for analysis was a challenging job
- We were unable to integrate Location specific data in our predictions.

FUTURE WORK

- We would like to use API collaboration platforms like Postman and provide two user interfaces. We would also integrate location data into our analysis.
- User-side Application.
- Company Interface.

Thank You!

Any questions?