CS4830 - Big Data Lab
Final Project

Classification on the NYC Parking Tickets Dataset

Group Members

Name	Roll No.
Abhiram S	CH19B037
Aditya Das	ME19B194
Dev Panghate	MM19B059

Introduction

With the help of Google Cloud Platform tools and services learnt in class, the necessary analysis is performed on a real-world dataset as part of this project. The dataset we are provided with is the New York City Parking Tickets dataset, which includes information about the vehicle ticketed, the ticket issued, location, and time, among others.

Problem Statement

The dataset is over 8 GB in size. The aim is to predict the **Violation Precinct**, using the remaining columns as predictors.

In order to generate the predictions, our method involved pre-processing the supplied data and training a machine learning model on it. To test the model and make predictions in the subscriber on data coming in from the publisher, we use the Kafka streaming model. The following steps have to be taken in order for this project to be completed:

1. Exploratory data analysis
2. Data preprocessing
3. Model training
4. Kafka streaming and prediction

Steps like EDA, data preprocessing and model training were done on a jupyter notebook running on a dataproc cluster.

Exploratory Data Analysis and Preprocessing

The schema of the dataset is shown below:

```
In [9]: dataset.printSchema()

root
 |-- Violation Precinct: integer (nullable = true)
 |-- Feet From Curb: integer (nullable = true)
 |-- Violation Time: string (nullable = true)
 |-- Violation In Front Of Or Opposite: string (nullable = true)
 |-- Issuer Precinct: integer (nullable = true)
 |-- Street Code2: integer (nullable = true)
 |-- From Hours In Effect: string (nullable = true)
 |-- Issuing Agency: string (nullable = true)
 |-- Street Code1: integer (nullable = true)
 |-- Issuer Code: integer (nullable = true)
 |-- Violation County: string (nullable = true)
 |-- Meter Number: string (nullable = true)
 |-- Plate Type: string (nullable = true)
 |-- Unregistered Vehicle?: string (nullable = true)
 |-- Issue Date: string (nullable = true)
 |-- Violation Post Code: string (nullable = true)
 |-- Street Code3: integer (nullable = true)
 |-- Double Parking Violation: string (nullable = true)
 |-- Plate ID: string (nullable = true)
 |-- Violation Code: integer (nullable = true)
 |-- Street Name: string (nullable = true)
 |-- Registration State: string (nullable = true)
 |-- Hydrant Violation: string (nullable = true)
 |-- Days Parking In Effect: string (nullable = true)
 |-- Vehicle Expiration Date: string (nullable = true)
 |-- Issuer Squad: string (nullable = true)
 |-- Vehicle Make: string (nullable = true)
 |-- Sub Division: string (nullable = true)
 |-- Intersecting Street: string (nullable = true)
 |-- Vehicle Year: integer (nullable = true)
 |-- Vehicle Color: string (nullable = true)
 |-- Time First Observed: string (nullable = true)
 |-- Summons Number: long (nullable = true)
 |-- Law Section: integer (nullable = true)
 |-- Violation Location: integer (nullable = true)
 |-- To Hours In Effect: string (nullable = true)
 |-- Issuer Command: string (nullable = true)
 |-- Date First Observed: string (nullable = true)
 |-- House Number: string (nullable = true)
 |-- Violation Legal Code: string (nullable = true)
 |-- No Standing or Stopping Violation: string (nullable = true)
 |-- Violation Description: string (nullable = true)
 |-- Vehicle Body Type: string (nullable = true)
```

Fig 1. Schema of dataset

We see that there are a total of 43 columns, of which 11 are of integer type and 31 are of string type. The remaining column is **Violation Precinct**, which is of integer type and is also the target column.

The dataset has 22436132 records, as shown by the count operation below:

```
In [9]: dataset.count()

Out[9]: 22436132
```

Fig. 2. Counting number of records

A summary of the numerical columns is shown below:

```
In [17]: dataframe.select(numeric_columns[:6]).describe().show()

[Stage 8:====================================================> (97 + 3) / 100]
```

summary	Feet_From_Curb	Issuer_Precinct	Street_Code2	Street_Code1	Issuer_Code	Street_Code3
count	21578152	21578155	21578156	21578156	21578155	21578156
mean	0.12368413198683557	48.87640440065427	21043.290231565665	24885.80743067202	351507.4804321315	21323.030743081104
stddev	0.86436461014174	62.130074088821274	21652.34142496915	22229.64916788092	225636.49198872087	21824.75520667577
min	0	0	0	0	0	0
max	16	999	98310	98260	999992	98280

Fig 3a. Summary of first half of numerical columns

```
In [18]: dataframe.select(numeric_columns[6:]).describe().show()

[Stage 14:====================================================>(99 + 1) / 100]
```

summary	Violation_Code	Vehicle_Year	Summons_Number	Law_Section	Violation_Location
count	21578156	21578152	21578156	21578154	18076494
mean	34.58567391022662	1534.7414078833071	6.519639906174689E9	527.6692627182102	54.94721481942239
stddev	19.638240768686735	852.5358352272534	2.1268497588655164E9	273.59713483382427	38.281710337343526
min	0	0	1002888438	0	1
max	99	2069	8470901369	2040	994

Fig 3b. Summary of second half of numerical columns

The following is a histogram of the target column, showing its distribution:

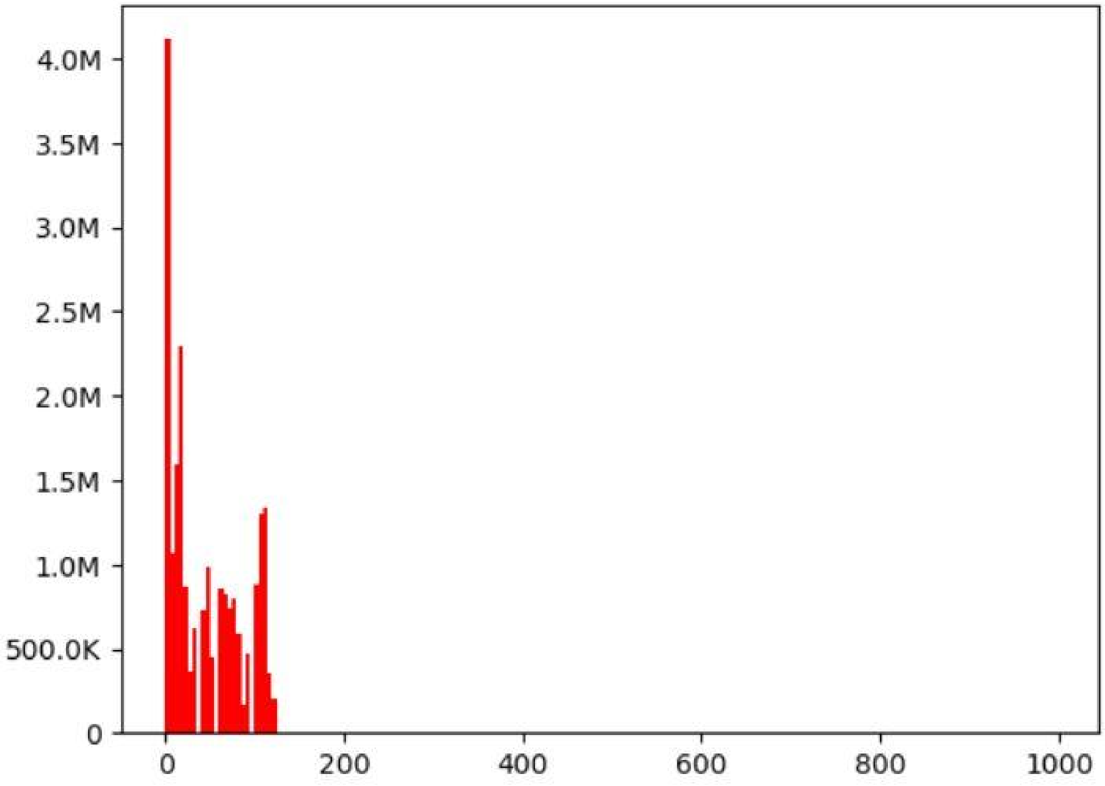

Fig 4. Distribution of Violation Precinct Column

Now, the following preprocessing steps were employed:

1. All duplicate rows were removed.
2. The spaces in all column names were replaced with underscores to convert them into valid identifier names.
3. Since MLLib's logistic regression supports prediction with not more than 100 classes, the rows with Violation Precinct greater than or equal to 99 were removed.
4. Those rows were removed which had null values in the Violation Precinct column
5. All null values were replaced by -1.
6. A string indexer was employed to perform label encoding on the string columns.
7. A vector assembler was employed to merge all columns into a single vector column

Model Training

A subset of 1000000 records of the dataset was used for model training and testing. This is because the model took too long to train for the entire dataset.

A logistic regression model with a regularisation parameter of 0.3 and an elastic-net regularisation parameter of 0.8 was used for model training. Elastic-net regularisation was used so that the benefits of both L1 and L2 regularisation could be obtained. This was the only model tested and no hyperparameter tuning was done due to lack of time. A test set accuracy of around 20% was obtained, as shown in the screenshot below:

```
In [ ]: predictions = model.transform(test)
        #predictions.select(['target', 'rawPrediction','probability','prediction']).show()
        evaluator = MulticlassClassificationEvaluator(labelCol="label", predictionCol="prediction", metricName="accuracy")
        accuracy = evaluator.evaluate(predictions)
        print("Accuracy of model is ", accuracy)
```

```
23/04/27 22:18:09 WARN org.apache.spark.scheduler.DAGScheduler: Broadcasting large task binary with size 36.1 MiB
23/04/27 22:18:21 WARN org.apache.spark.deploy.yarn.YarnAllocator: Container from a bad node: container_1682631560925_0001_01_0
00006 on host: cluster-e86c-m.c.cs4830-lab2.internal. Exit status: 137. Diagnostics: [2023-04-27 22:18:20.997]Container killed
on request. Exit code is 137
[2023-04-27 22:18:20.997]Container exited with a non-zero exit code 137.
[2023-04-27 22:18:20.998]Killed by external signal
.
23/04/27 22:18:21 WARN org.apache.spark.scheduler.cluster.YarnSchedulerBackend$YarnSchedulerEndpoint: Requesting driver to remo
ve executor 6 for reason Container from a bad node: container_1682631560925_0001_01_000006 on host: cluster-e86c-m.c.cs4830-lab
2.internal. Exit status: 137. Diagnostics: [2023-04-27 22:18:20.997]Container killed on request. Exit code is 137
[2023-04-27 22:18:20.997]Container exited with a non-zero exit code 137.
[2023-04-27 22:18:20.998]Killed by external signal
.
23/04/27 22:18:21 ERROR org.apache.spark.scheduler.cluster.YarnScheduler: Lost executor 6 on cluster-e86c-m.c.cs4830-lab2.inter
nal: Container from a bad node: container_1682631560925_0001_01_000006 on host: cluster-e86c-m.c.cs4830-lab2.internal. Exit sta
tus: 137. Diagnostics: [2023-04-27 22:18:20.997]Container killed on request. Exit code is 137
[2023-04-27 22:18:20.997]Container exited with a non-zero exit code 137.
[2023-04-27 22:18:20.998]Killed by external signal
.
23/04/27 22:18:21 WARN org.apache.spark.scheduler.TaskSetManager: Lost task 0.0 in stage 115.0 (TID 555) (cluster-e86c-m.c.cs48
30-lab2.internal executor 6): ExecutorLostFailure (executor 6 exited caused by one of the running tasks) Reason: Container from
a bad node: container_1682631560925_0001_01_000006 on host: cluster-e86c-m.c.cs4830-lab2.internal. Exit status: 137. Diagnostic
s: [2023-04-27 22:18:20.997]Container killed on request. Exit code is 137
[2023-04-27 22:18:20.997]Container exited with a non-zero exit code 137.
[2023-04-27 22:18:20.998]Killed by external signal
.
[Stage 115:>                                                          (0 + 1) / 1]

Accuracy of model is  0.20190633245382586
```

Fig 5. Screenshot showing model training and test accuracy

The model was then saved on a bucket to be used later on for Kafka streaming.