CS4830 : Big Data Lab
Lab 3 Due Date : 19 February

Roll No: ME19B193, ED19B030, BE18B014 Name: Shreyas Kulkarni, Shreyas Singh, Sivakumar

1. Objective

Solution:
The dataset provided for the project is the NYC Parking Tickets Dataset which is based on the Kaggle Dataset.
There are 43 columns in the dataset and 22,436,132 rows.
The aim of the project is to predict the part in the city where the ticket was issued (Prediction column name:- Violation County).
We shall perform data pre-prcessing and model training on a spark cluster. In addition to this, we will stream the test data stored on the GCS bucket into Kafka.

2. Pre-processing

Solution:
Removing Null Columns:
Some columns in the dataset have a lot of null or NaN values. Hence, we remove the columns which have more than 50% of null or NaN values. The dropped columns are the following:
'From Hours In Effect', 'Intersecting Street', 'Time First Observed', 'Violation Legal Code', 'Unregistered Vehicle?', 'Meter Number', 'No Standing or Stopping Violation', 'Hydrant Violation', 'Double Parking Violation', 'To Hours In Effect'.

Removing Duplicate Rows:
After removing all the duplicate rows we get 21,578,156 rows.

Dropping Irrelevant Columns:
Some of the columns in the dataset which are irrelevant for our target prediction are the following:
Summons Number: It is a unique identifying key, a numeric value which has no relation to the data itself.
Plate ID: The plate id was noticed to be arbitrary. The serial formatting of the plate id itself could give an idea about the location, as different states use different formats. However, the column 'Vehicle Registration' is assumed to be a better representation of this information.

Vehicle Expiration Date: Some entries in this column had erroneous or malformed data entries, and the Expiration Date of a vehicle does not seem to be relevant to the task.
House Number: House number is a code consisting of one or more values combined together. We could not conclude on a specific pattern or relevant categorization of this column.
Vehicle Color: The column has many mismatched names. For instance White is represented by WH, WHTE, WHITE etc. Identifying all such colors can be tedious. Violation Post Code: It has close to 25% missing values and Violation Code seems to be a better representation. This has been removed to reduce the correlation between the features.
Date First Observed: We remove this time based feature as we are already considering Violation Time and that time feature when extracted shall act as a more meaningful feature. In addition to that more often then not, date first observed and violation time have the same day as not many people violate multiple times.
Days Parking in Effect: This column does not contain a simple number or an understandable feature. It has multiple categories which we think might be unnecessary for our model. In addition to that, it has more than a million null values.
Vehicle Year: It has more than 2 million values which have a numerical value of vehicle year less than 100. While the parking tickets are issued in 2010s, there are some vehicle years after the violation date issued.
Feet from Curb: Almost all values are 0 in this column and it might have zero significance due to such high skewness in data.
Street Name: Street Name is name-based and has many unique values (each name is a unique entry) and no common patterns within the names that could be exploited to uncover a feature.
Issue Code: It has more than 51k unique values. As it is categorical, StringIndexer would make the features very high dimensional.
Vehicle Make: It has more than 10k unique values and after evaluating the feature importance of the model, It had very low importance given.

Sample of unique values. As noticed here, Issue code has a lot of unique values and hence, it was dropped.

```
Column:  Violation Precinct      Count:  591

Column:  Violation In Front Of Or Opposite       Count:  6

Column:  Issuer Precinct         Count:  815

Column:  Street Code2    Count:  7145

Column:  Issuing Agency          Count:  19

Column:  Street Code1    Count:  6813

Column:  Issuer Code     Count:  51754

Column:  Violation County        Count:  21

Column:  Plate Type      Count:  89

Column:  Street Code3    Count:  6939

Column:  Violation Code          Count:  100

Column:  Registration State      Count:  69

Column:  Issuer Squad    Count:  46
```

Further processing of retained features

Now there are 20 columns left which we shall process for our model. These columns are: 'Violation Precinct', 'Violation Time', 'Violation In Front Of Or Opposite', 'Issuer Precinct', 'Street Code2', 'Issuing Agency', 'Street Code1', 'Violation County', 'Plate Type', 'Issue Date', 'Street Code3', 'Violation Code', 'Registration State', 'Issuer Squad', 'Sub Division', 'Law Section', 'Violation Location', 'Issuer Command', 'Violation Description', 'Vehicle Body Type'.

Apart from Violation Time, all these columns represent categorical data. Even in columns which have integer values, are in effect a categorical attribute (example: Street Code). The exact numeric value has no significance. Therefore, they can be considered as categorical variables in this case.

'Violation Time' is binned into 6 time intervals and a new column 'Time Bin' is created, after which 'Violation Time' is dropped.

Violation Time	Time bin
0740A	2
0458P	5
1204P	4
1050P	6
0351P	4
0823A	3
1152P	6
0938A	3
0158P	4
0108P	4

only showing top 10 rows

From 'Issue Date' feature, the day, month and year following which new features called 'Issue Day', 'Issue Month' and 'Issue Year' were created.

Issue Date	Issue Day	Issue Month	Issue Year
2015-08-09	1	8	2015
2016-03-01	3	3	2016
2015-05-01	6	5	2015
2015-06-12	6	6	2015
2015-02-23	2	2	2015
2015-12-18	6	12	2015
2015-02-12	5	2	2015
2014-10-01	4	10	2014
2015-05-28	5	5	2015
2016-04-20	4	4	2016

only showing top 10 rows

For the existing Null values in the columns, they are replaced by the most occurring elements.

```
default_values = {
    'Registration State': 'NY',
    'Plate Type': 'PAS',
    'Issue Month': 6,
    'Issue Year': 2015,
    'Vehicle Body Type': 'SUBN',
    'Issuing Agency': 'T',
    'Street Code1': 0,
    'Street Code2': 0,
    'Street Code3': 0,
    'Issuer Command': 'T103',
    'Issuer Squad': 'A',
    'Violation In Front Of Or Opposite': 'F',
    'Violation County' : '00000',
    'Violation Description' : "38-Failure to Display Muni Rec",
    'Violation Location' : 54,
    'Law Section' : 459,
    'Sub Division' : 'h1',
    'Violation Code': 35,
    'Issuer Precinct': 56
    }
```

3. Performance of the best model and Inferences

Solution: Model Pipeline
Input → String Indexers → Vector Assembler → Model → Prediction → IndexToString → Evaluation

Random Forest Classifer Training We have used RandomForestClassifier for our training. We use $maxBins = 10000$ as that needs to be less than the number of distinct labels in every column of the features.

```
print(f"F1: {F1:.5f}")
print(f"Accuracy: {Accuracy:.5f}")
```

```
F1: 0.79876
Accuracy: 0.89644
```

```
test_data.select(['predictedLabel', 'Violation County']).show(10)
```

```
+--------------+----------------+
|predictedLabel|Violation County|
+--------------+----------------+
|            NY|              NY|
|             Q|               Q|
|            NY|              NY|
|             K|               K|
|             K|               K|
|            NY|              NY|
|             K|               K|
|            NY|               Q|
|             K|               K|
|            NY|               R|
+--------------+----------------+
```

4. Real-time computation (with screenshots) and latency of processing each window

Solution: We used Kafka with Structured Streaming for real time computation.

```
|predictionLabel|Violation_County|
+---------------+----------------+
|             NY|              NY|
|              K|               K|
|              Q|               Q|
|             BK|              BK|
|               |                |
|              Q|             ONS|
|              K|               Q|
+---------------+----------------+

--> Batch 60 Processing Time:  29.860914 secs
--> Batch 60 Accuracy:  85.71 %
--> Batch 60 F1-score:  0.90

+---------------+----------------+
|predictionLabel|Violation_County|
+---------------+----------------+
|             NY|              NY|
|             NY|               Q|
|              Q|               Q|
|             BK|              BK|
|               |                |
|              K|               Q|
|              K|               K|
+---------------+----------------+

--> Batch 61 Processing Time:  29.808074 secs
--> Batch 61 Accuracy:  71.43 %
--> Batch 61 F1-score:  0.62
```

5. Conclusion

Solution:
We retrieved data from the given NYC Parking Dataset on GCP and used pyspark library to analyse, train and infer on the dataset.
We used ML libraries and provided predictions on Violation County. We had first oriented all our code towards Violation Precinct but later it turned out to have 591 classes and we did not enough compute allowed to process it fast enough. As mentioned in the recent mail, we incorporated that.
We used Kafka for real time processing and predicting through Structered Streaming.

We also noticed some repetitions in Violation County dataset. However, we did not make any changes to that as target variable should not be altered with half knowledge.
The dataset was very huge and we noticed that such high level processing could not be handled by google colab or our own systems. We had a hands on experience in this.