

**DEPT OF COMPUTER SCIENCE AND ENGINEERING**

**PES UNIVERSITY**

(Established under Karnataka Act No. 16 of 2013)

100ft Ring Road, Bengaluru- 560 085, Karnataka, India

**May-August 2022**

**UE21CS645BC1- Advanced Big Data Analytics**

***Project Report***

**Master of Technology**

**In**

**Computer Science and Engineering**

***Submitted by-***

**PES1PG21CS027 Parul Singh**

**PES1PG21CS030 Rakshith Gowda**

**2nd Semester M. Tech**

**Problem Statement Overview:**

This simulates a real-world scenario where you will be required to handle an enormous amount of data for predictive modelling. The data source is a stream and your application faces the constraint of only being able to handle batches of a stream at any given point in time.

With this project, you will learn more about how applications in the real world modify their algorithms to work on large data streams and how incremental processing can be leveraged to process and analyse streams over time.

**Software/Languages used:**

Python

Hadoop

Spark

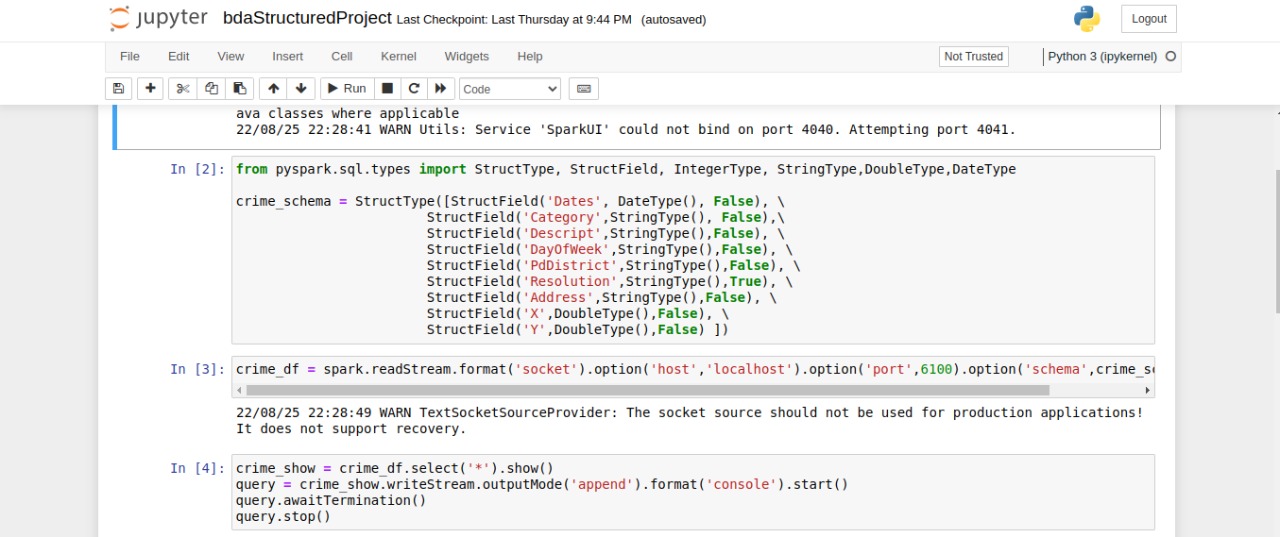
**Libraries used:**

* Pyspark
* numpy
* pandas
* scikit-learn
* matplotlib

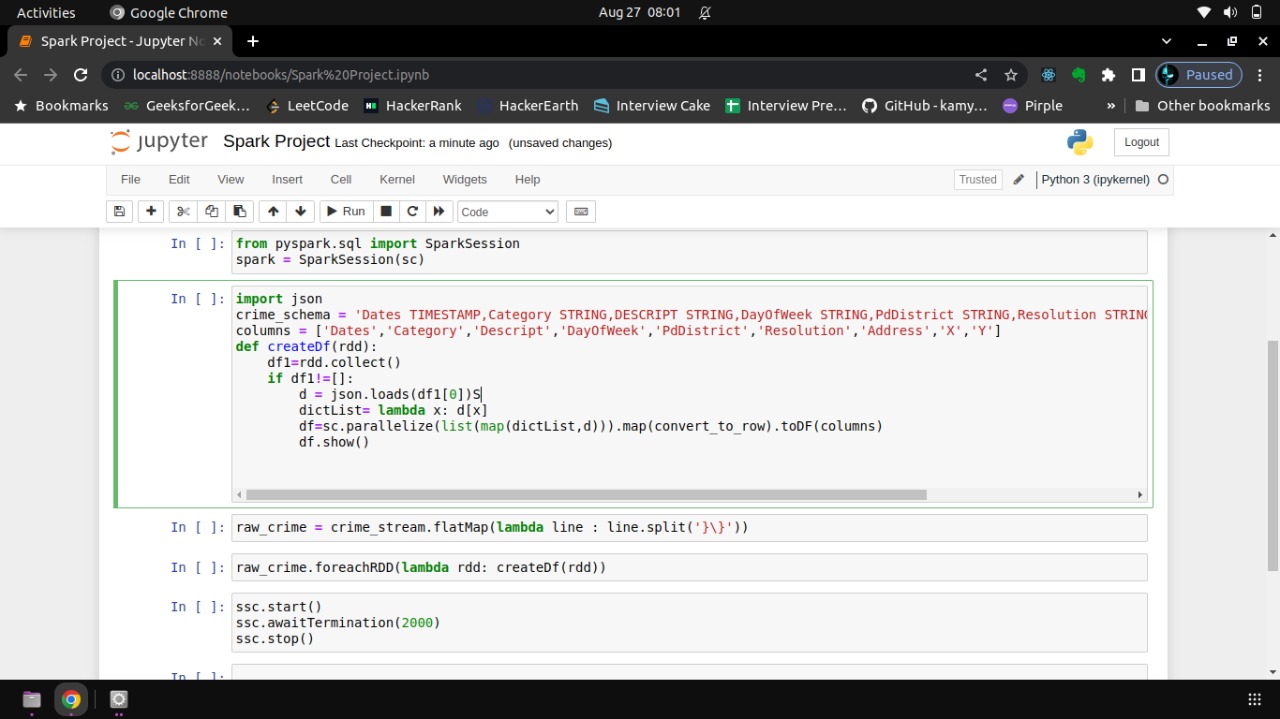
**Implementation:**

**Step 1: Streaming Data**

1. Using **spark structured streaming** – ended up with errors regarding parsing the streaming text into dataframe with header

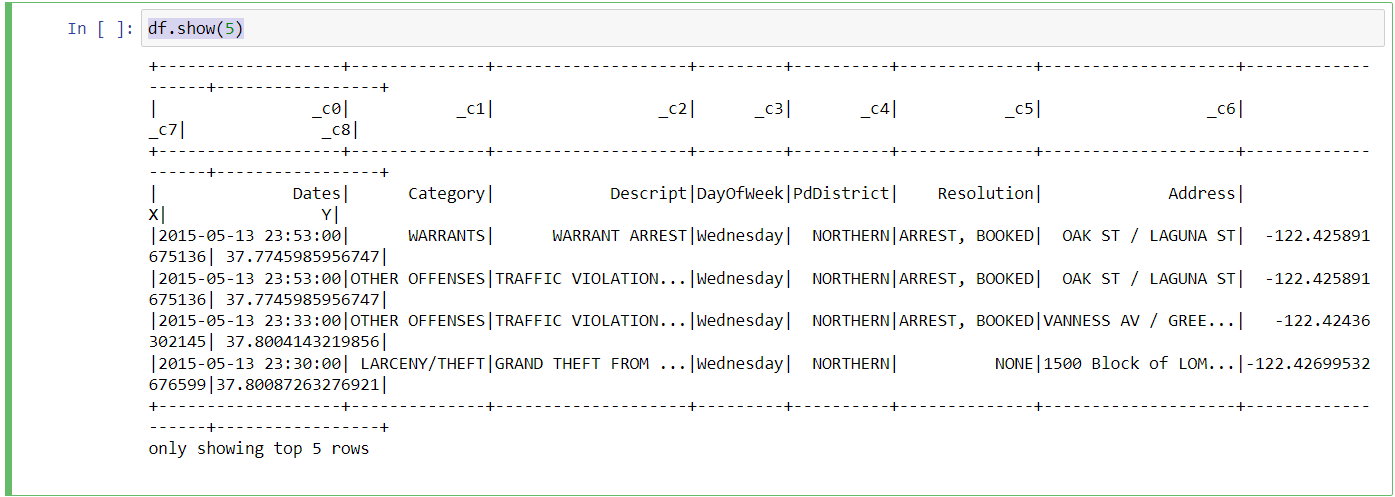


1. Using **spark streaming** – data was getting steamed but errors regarding parsing persisted

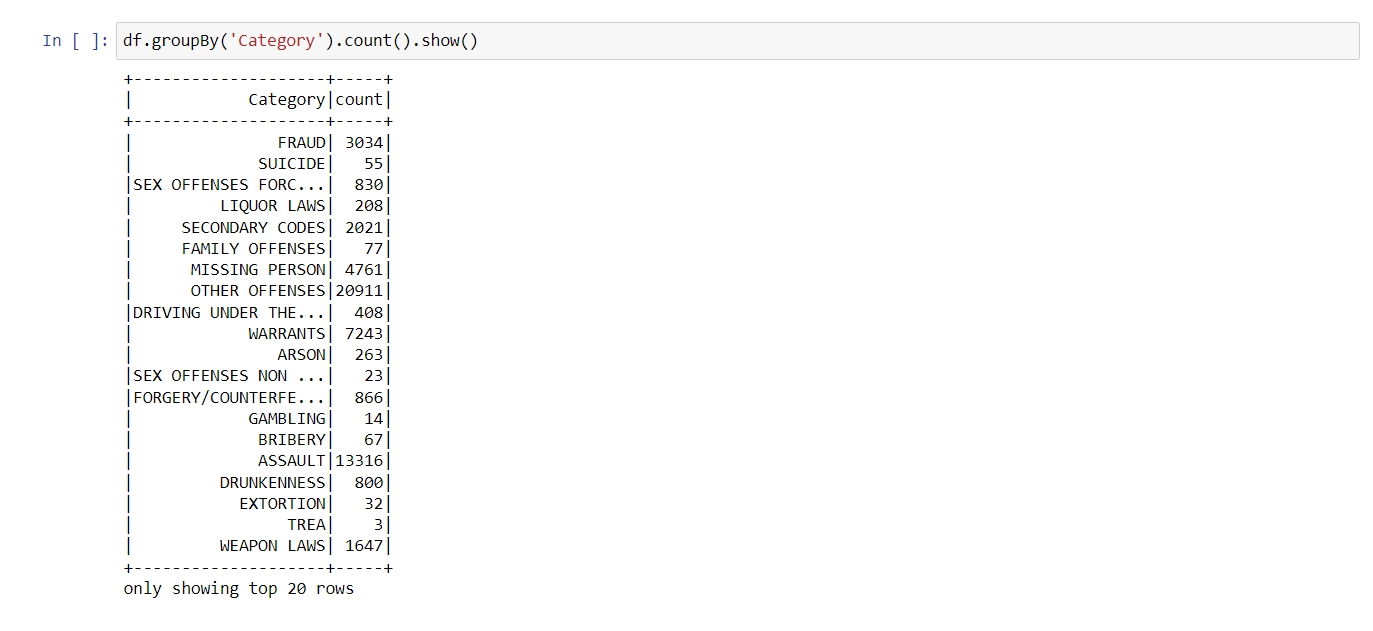


**Step 2: EDA**

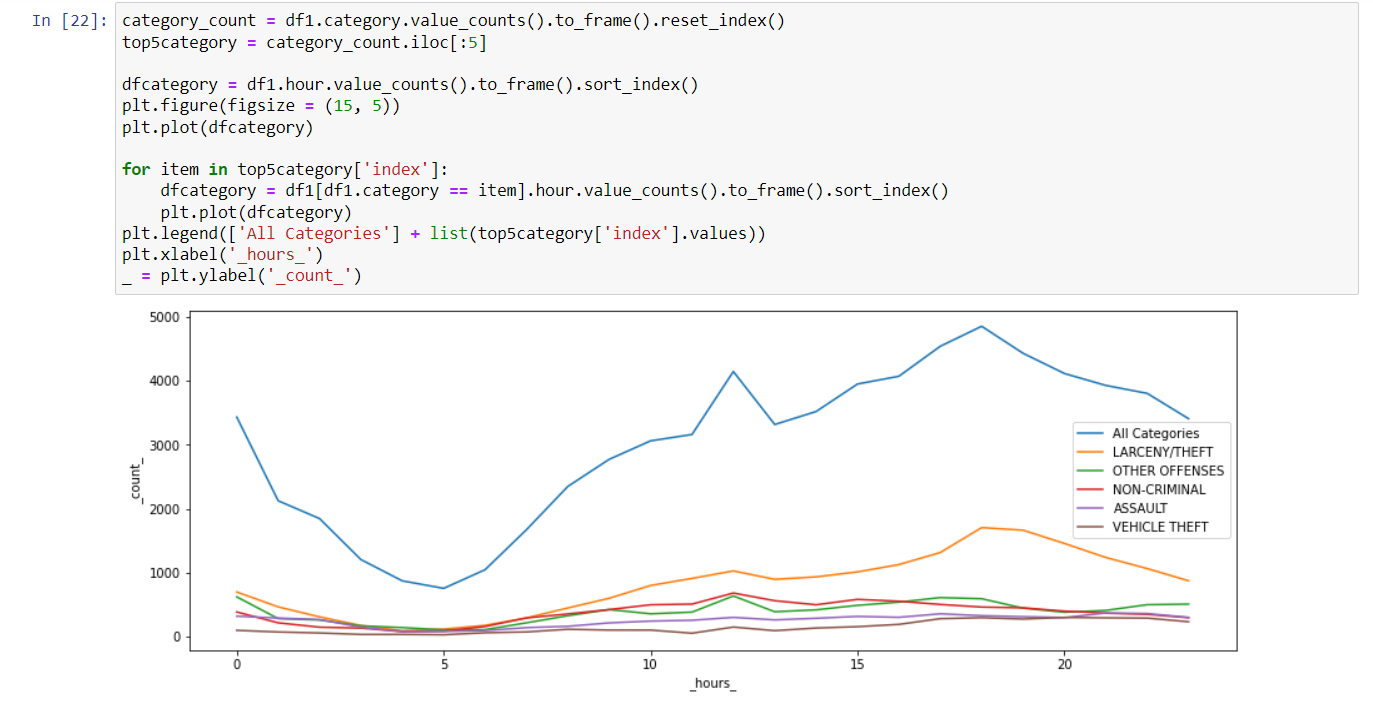
1. Get the starting 5 rows

****

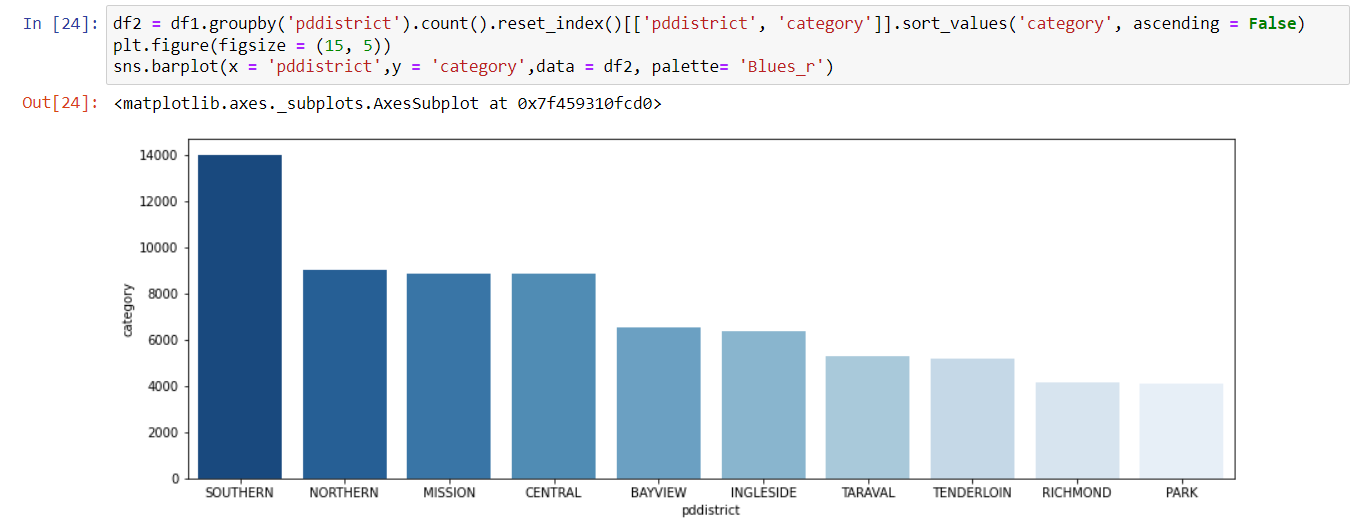
1. Get the count of each category of crimes

****

1. Count of types of crime occurred in different hours of the day

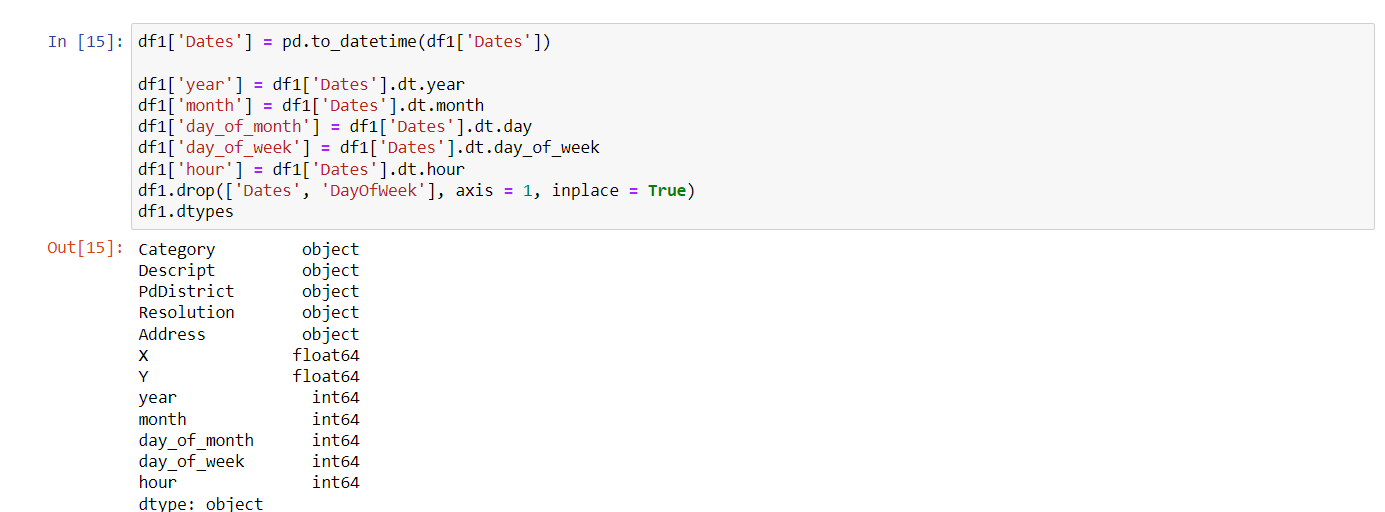


1. District x category plot

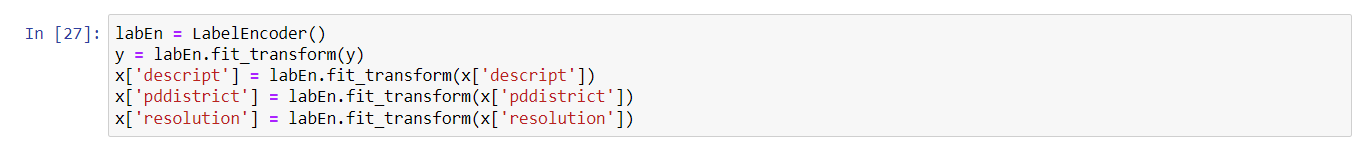


**Step 3: Pre-processing**

* Create new columns from dates column like hour, month, year for better querying



* Convert categorical values into numeric features



* Scale values in the dataset using MinMaxScaler and StandardScaler

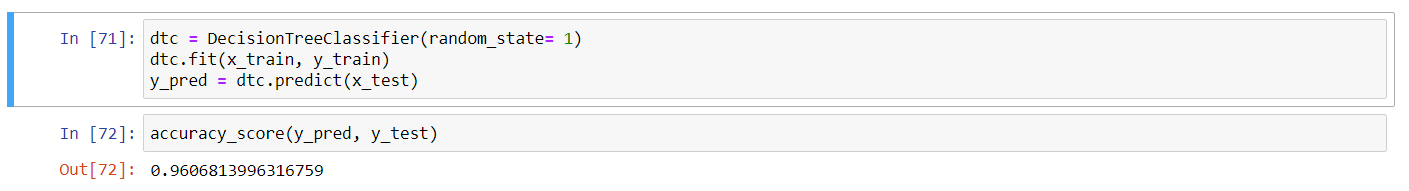


**Step 4: Train and testing the model**

1. Random Forest Classifier



1. Decision Tree Classifier



1. XG Boost Classifier



1. KNN Classifier



**Results**

By observing the accuracy of each of the classification algorithms applied, we can see that Decision tree classifier gives us the best accuracy. However, this may not be the case, by tuning the hyper parameters each classifier may produce better results.

|  |  |
| --- | --- |
| CLASSIFIER | PREDICTION\_ACCURACY |
| RANDOM FOREST | 0.8980662983425415 |
| DECISION TREE | 0.9606813996316759 |
| XG BOOST | 0.800828729281768 |
| KNN | 0.800828729281768 |

**Conclusion**

Though unable to stick to the instructions for the project, we produced partial results on the machine learning aspect of the project.