sUBMITTED BY:

Bhagya Shree A00431152

Rishab Gupta A00429019

**BIG DATA PROJECT**

mcda5570

Table of Contents

[Executive Summary 2](#_Toc5235447)

[Description of Data 2](#_Toc5235448)

[Inspiration 2](#_Toc5235449)

[Steps followed 3](#_Toc5235450)

[Technologies Used 4](#_Toc5235451)

[1. Hadoop Distributed File System: 4](#_Toc5235452)

[2. Apache Hive 4](#_Toc5235453)

[3. Tableau Visualization 4](#_Toc5235454)

[4. Apache PySpark 5](#_Toc5235455)

[5. Apache MLlib 5](#_Toc5235456)

[Insights 7](#_Toc5235457)

[1. Visualization 7](#_Toc5235458)

[2. Data Cleaning, Label Encoding and Vectorization 9](#_Toc5235459)

[3. Machine Learning Model 10](#_Toc5235460)

[Conclusion 11](#_Toc5235461)

# Executive Summary

Boston Crime data was used to predict the reporting area of the crime. As the first step, exploratory analysis and visualization was done on the data to understand the changing trends of crime over the past four years in each district in Boston.

Given the features like Offense Code, Date, UCR Part and the District, the model was able to predict the reporting area by ~50% accuracy.

# Description of Data

This dataset contains records from the new crime incident report system, which includes a reduced set of fields focused on capturing the type of incident as well as when and where it occurred in the Boston City. Below is the description of the fields of the dataset:

1. **INCIDENT\_NUMBER:** Unique incident number
2. **OFFENSE\_CODE**: Every Offense has a unique code
3. **OFFENSE\_CODE\_GROUP**: The name of the offense
4. **OFFENSE\_DESCRIPTION**: The description of every offense
5. **DISTRICT**: District where the offense occurred
6. **REPORTING\_AREA**: The reporting area of the offense
7. **SHOOTING**: *null* for no shooting; *Y* for shooting
8. **OCCURRED\_ON\_DATE**: The date the offense was occurred (YYYY/MM/DD HH:MM:SS)
9. **YEAR**: Year the offense occurred
10. **MONTH**: Month the offense occurred
11. **DAY\_OF\_WEEK**: Day the offense occurred
12. **HOUR**: Hour the offense occurred
13. **UCR\_PART**: The Uniform Crime Report offence belonged to
14. **STREET**: The street where offence occurred
15. **LATITUDE**: Latitude of the place where offence occurred
16. **LONGITUDE**: Longitude of the place where offence occurred
17. **LOCATION**: Latitude & Longitude combined

# Inspiration

1. How has crime changed over the years?
2. Is it possible to predict where and when a crime will be committed?
3. Which areas of the city have evolved over this time span?
4. Which areas of the city have been most affected by crime?

# Steps followed

Run Machine Learning Models

Convert Features to a vector

Data Cleaning

Create Dataframe (DF) in Apache Spark

Export DF to Hive for visualization in Tableau

Label Encoding

Copy file from Local File System to HDFS

# Technologies Used

## Hadoop Distributed File System:

HDFS was used to fetch the data uploaded on the dev server and to copy the csv file to the HDFS folder. Given below are the commands used to complete this operation:

wget http://dev.cs.smu.ca/~r\_gupta/crime.csv

hadoop fs -mkdir /assignment

hadoop fs -put crime.csv /assignment

## Apache Hive

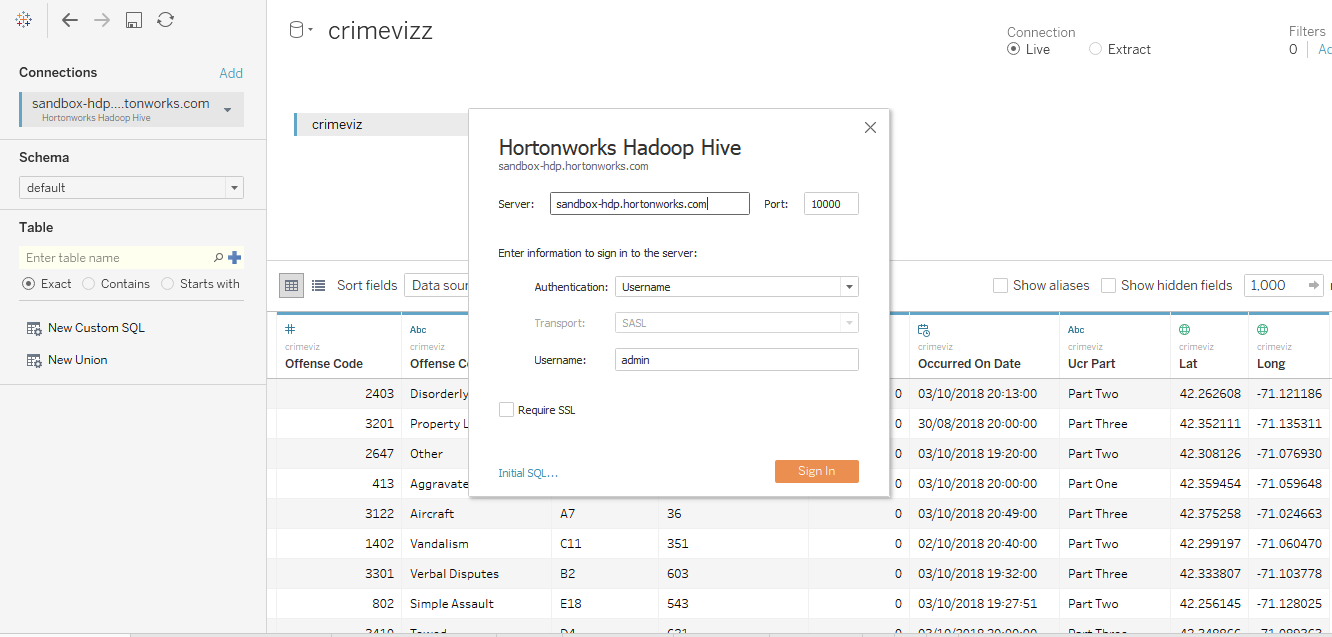
The dataset from the HDFS was imported as a PySpark dataframe. Before proceeding with the Tableau visualization, some basic cleaning was performed using PySpark for seamless visualization. The dataset was then exported to Apache Hive, to create a server connection from Tableau using Hortonworks Hadoop Hive server.

Command to export the csv file to hive:

crimeviz.write.mode("overwrite").saveAsTable("crimeviz")

## Tableau Visualization

Tableau was used for exploratory analysis and visualization of data. From the visualization we tried to understand the trend in crime over the years, which areas are most affected by crime and during what time of the day most crimes happen.



## Apache PySpark­

Apache PySpark was used for data cleaning and data manipulation. The data cleaning process involved:

* 1. As a first step, the csv file on the HDFS was imported as a data frame.

crime = spark.read.csv("/assignment/crime.csv", header = True, inferSchema = "true")

* 1. The unwanted columns were dropped and the rows containing null values were removed.

crime = crime.drop("INCIDENT\_NUMBER", "OFFENSE\_DESCRIPTION", "YEAR", "MONTH", "HOUR", "DAY\_OF\_WEEK", "STREET", "Lat", "Long", "Location")

crime = crime.na.drop(subset=["UCR\_PART", "DISTRICT", "OFFENSE\_CODE", "OFFENSE\_CODE\_GROUP", "REPORTING\_AREA", "OCCURRED\_ON\_DATE"])

* 1. The null values in the field “Shooting” were filled with 0

crime = crime.fillna("0")

* 1. The frequency count of Offense code group was calculated and only the offenses that were committed more than 10,000 times were filtered. The remaining rows were dropped. We were now left with 208201 out of 327820 rows.

crimecount = crime.join(crime.groupBy("OFFENSE\_CODE\_GROUP").count(), on = "OFFENSE\_CODE\_GROUP")

crimecount = crimecount.filter(crimecount["count"] > 10000)

crimecount = crimecount.drop("count")

## Apache MLlib

Apache MLlib is a Spark Machine Learning Library used to make practical machine learning scalable and easy. This library was used to:

1. Encode the categorical data to numerical data using StringIndexer. Example of label encoding for one of the columns is given below.

dis = StringIndexer(inputCol = "DISTRICT", outputCol = "DISTRICT\_CAT")

crimecount = dis.fit(crimecount).transform(crimecount)

1. The encoded features were now required to be vectorized to feed into the machine learning model. The vectorized features and the dependent variable were filtered into a new dataframe *crime\_df*

assembler = VectorAssembler(inputCols=["OFFENSE\_CODE", "SHOOTING\_CAT", "OCCURRED\_ON\_DATE", "DISTRICT\_CAT", "OFFENSE\_CODE\_GROUP\_CAT", "UCR\_PART\_CAT"], outputCol="features")

crimecount = assembler.transform(crimecount)

crime\_df = crimecount.select(["REPORTING\_AREA", "features"])

1. The new dataframe was split into training and testing set in the 0.7 and 0.3 ratio respectively

splits = crime\_df.randomSplit([0.7, 0.3])

train = splits[0]

test = splits[1]

1. After the split, multiple regression models were run and the R2 value (how close the data is to the fitted line) was compared for each model to choose the best regression model for our data. Out of the four models that were run, Gradient Boost Tree Regression model (GBTR) gave the highest R2 value of ~56%. The code for GBTR is given below

from pyspark.ml.regression import GBTRegressor

gbt = GBTRegressor(featuresCol = 'features', labelCol = 'REPORTING\_AREA', maxIter=10)

gbt\_model = gbt.fit(train)

gbt\_predictions = gbt\_model.transform(test)

gbt\_predictions.select('prediction', 'REPORTING\_AREA', 'features').show(5)

gbt\_evaluator = RegressionEvaluator(labelCol="REPORTING\_AREA", predictionCol="prediction", metricName="r2")

print("R Squared (R2) on test data = %g" % gbt\_evaluator.evaluate(gbt\_predictions))

1. The table containing the actual values and the predicted values was now exported to Hive and the graph for “Actual vs Predicted” values was plotted in Tableau.

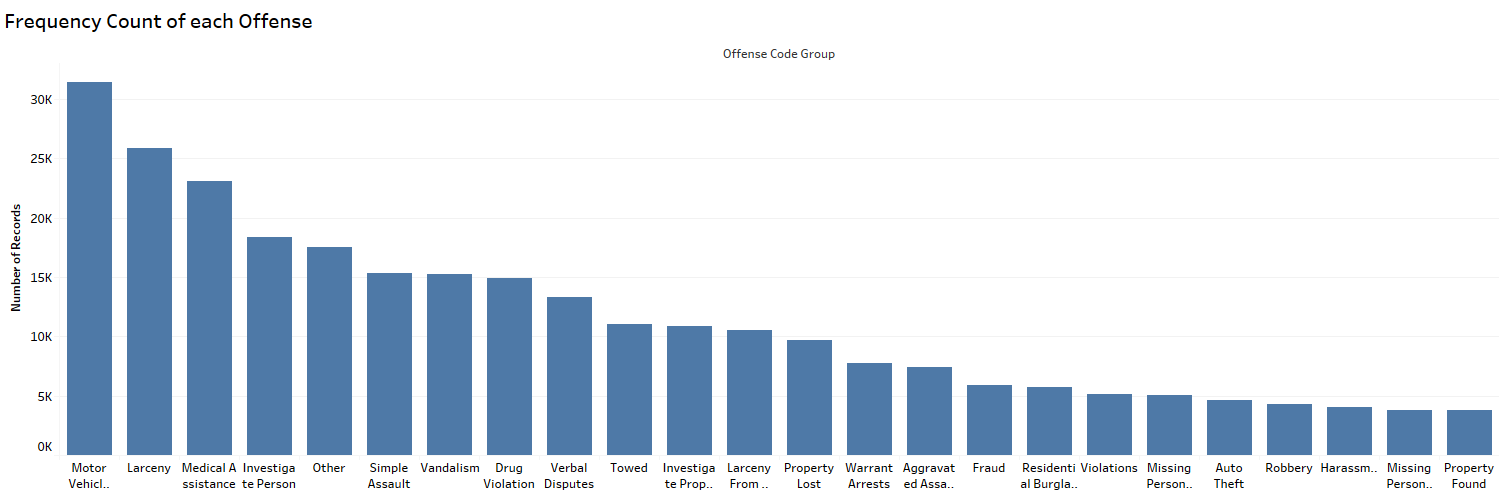
gbt\_predictions.write.mode("overwrite").saveAsTable("gbt")

# Insights

This section will talk about the insights and the results that were gathered during the exploratory analysis, visualization and while applying machine learning to the data.

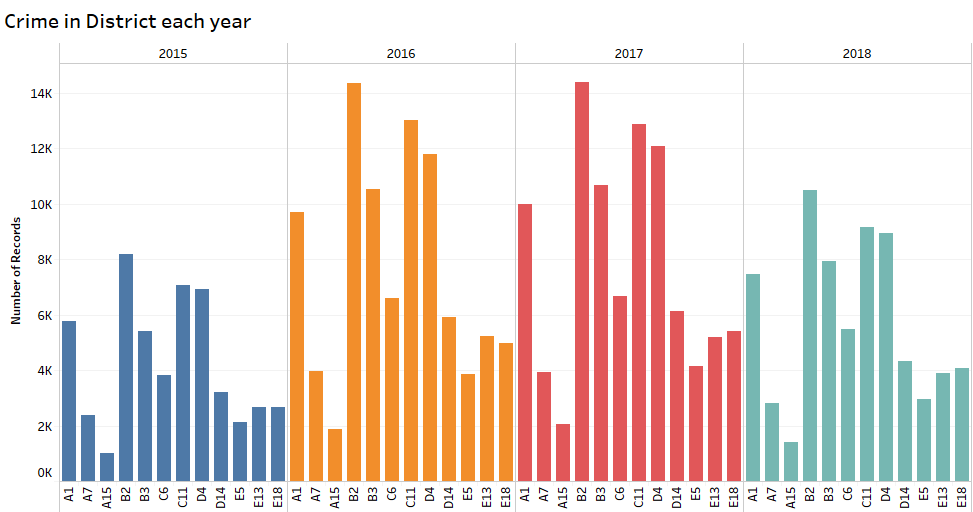
## Visualization

1. **Frequency count of each offense**

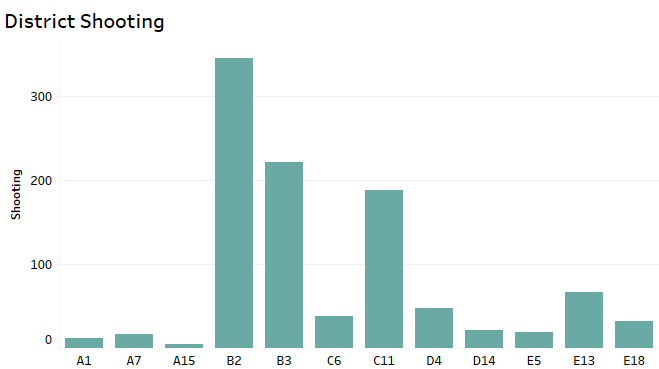
Fig 1 is the part of the snippet of the total 63 offenses committed. Only the top offenses with frequency greater than 10,000 were selected for our Machine Learning model, because we aim to predict the area with the most committed offenses.

1. **Crime in each district over the year**

This graph compares the number of criminal offenses in each district over the last four years. The years 2016 and 2017 faced the maximum number of criminal offenses compared to the years 2015 and 2018. A similar trend is observed in all the districts. The districts B2, C11 and D4 were the most affected by crime during all the four years.

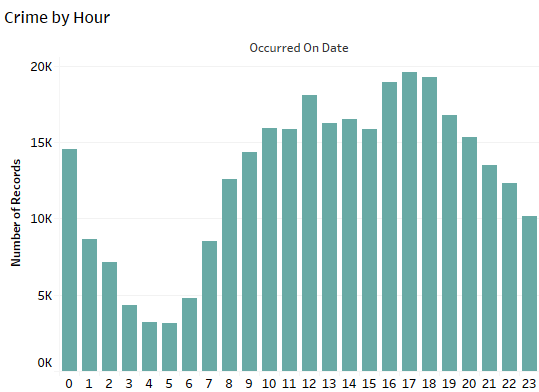


1. **Number of Shootings in each District**

This graph compares the number of shooting during the four year period in each district. Once again, district B2 is the one most affected by the number of shootings followed by B3 and C11.

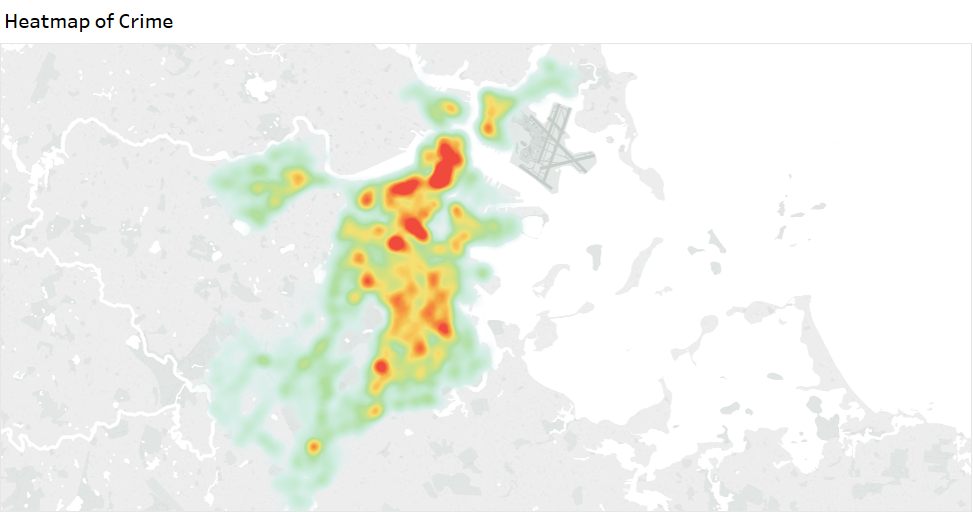
1. **Crime by Hour**

What time of the day is most prone to criminal offenses? This graph answers this question. The crime is least during the early morning hours and gradually hikes till 6 in the evening and begins to drop again. It can be reckoned from the graph that most of the crime occurred during the day.



1. **Heatmap of Crime in Boston**

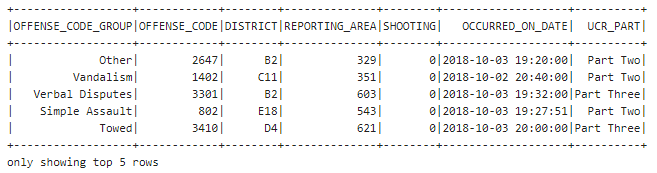
The heatmap shows the areas which are most affected by crime. As it can be seen that the outskirts of the city did not observe much crime compared to the blood red northern part of the city.



## Data Cleaning, Label Encoding and Vectorization

After the visualization, the data was cleaned, and the categorical data was label encoded and finally the features were converted to vectors before feeding it into the machine learning model.

1. **Data Cleaning**

During the data cleaning process, the rows with null values were dropped and some columns (features) which did not contribute much towards building a good regression model and did not have a strong relationship to the dependent variable i.e. Reporting Area, were dropped. The dataset finally looked like:

1. **Label Encoding**

The features containing the categorical data cannot be fed to the machine learning model. Therefore, to overcome this, the categorical data was label encoded. Below is the snippet of the dataset after the label encoding.

1. **Vectorization**

Vector is a data type that is used by Spark ML. For most of the machine learning algorithms, Spark ML requires converting features into vectors. After applying vectorization, the dataset looked like:

## Machine Learning Model

Now that the dataset was ready to be fed into the machine learning model; the challenge faced was to decide which regression model will be the most suitable to predict the area of the crime. To overcome this challenge, we ran four regression models and compared their results. The results of each of the regression models are presented below.

1. **Linear Regression Model**

Linear Regression Model is the most common form of regression model. It is used to explain the relationship between one continuous dependent variable and two or more independent variables.

1. **Decision Tree Regression Model**

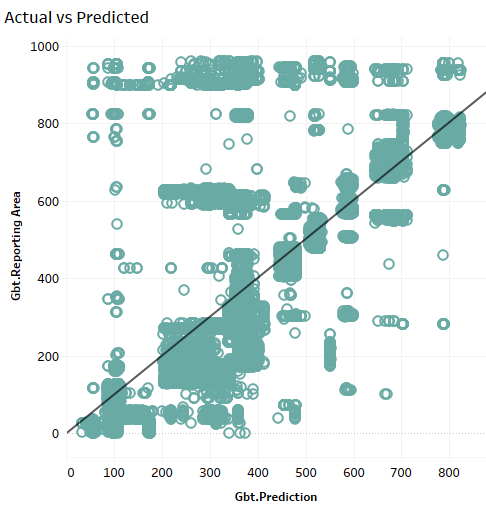
A decision tree is a machine learning algorithm that partitions the data into its subsets. The portioning process begins with a binary split and continues until no further splits can be made.

1. **Random Forest Regression Model**

Random Forest Regression is an ensemble learning method for regression that operates by constructing a multitude of decision trees at training time

1. **Gradient Boost Tree Regression Model**

Gradient boosting is a machine learning technique for regression and classification problems, which produces a prediction model in the form of an ensemble of weak prediction models, typically decision trees. (*Wikipedia*)



# Description of Data

E-Commerce dataset is consisted of two txt files. First file is ‘os\_order.txt’ contains general information about orders between April 2010 to December 2011. The columns have buyer, date, location (ip) and status related information. Second file is ‘os\_order\_items.txt’ which contains more granular information about the details of every transaction. We can access individual item id’s in each transaction alongside with buying price and retail price with the quantities. Datasets are merged using order id’s when necessary through the analysis. Unofficial metadata is as follows:

**OS\_ORDER**

1. **ORDER\_ID:** Unique order id
2. **ORDER\_CODE:** 12 digit unique order code
3. **BUYER\_ID:** 5 digit unique number assigned to buyer
4. **PAY\_DT:** ‘yyyy-mm-dd HH:MM:SS’ formatted date for time of transaction
5. **CREATE\_IP:** IP Address where the order is placed from
6. **ORDER\_STATUS:** Status information takes values between 1-9, A, B, z

**OS\_ORDER\_ITEM**

1. **ITEM\_ID:** Unique index number for the purchased item
2. **ORDER\_ID:** Unique order id
3. **GOODS\_ID:** 7 digit unique item id of the purchased item
4. **GOODS\_NUMBER:** Quantity of the purchased item
5. **SHOP\_PRICE:** Retail price of the item
6. **GOODS\_PRICE:** Wholesale price of the item for the company
7. **GOODS\_AMOUNT:** Revenue generated from the transaction

# Conclusion

We can now cut crimes, save time and optimize resources by using data driven insights to help know what’s coming. Now, given the Offense Code, Date and District, the area where the crime is most likely to occur can be predicted.

It is vivid from the insights that Gradient Boost Regression Model gives us the most accurate prediction results out of the four models.