Master of Information Security

MIS4203 – Independent Studies in Information Security

Index Number – 16770217 | Reg. Number – 2016MIS021

Study Number – 02

Privacy Preserving Data Analysis and Mining

November 17, 2018

University of Colombo School of Computing

Table of Contents

Study Number – 02 1
Privacy Preserving Data Analysis and Mining 1
Problem 3
Approach 3
Conclusion 3

Problem

Given two data sets which are complaints about crimes and taxi rides data. Problem is to analyse and mine the data sets which can be a crime, complaints by the taxi drivers.

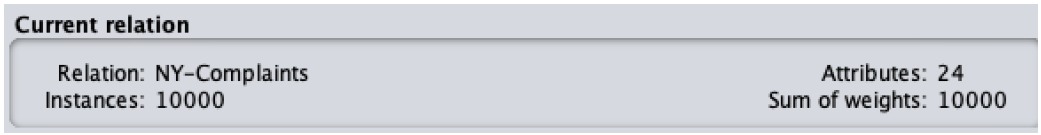
Approach

1. Analyse and understanding of the problem and data set related to the problem.
2. Understand the data set and decide which data needs to do the analysis.
3. Clean the data sets by;
    a. Removing unwanted columns.
    b. Remove unwanted records.
    c. Remove/Modify null values.
    d. Make datasets more usable for analysis.
4. Define mining strategy to get optimal solution.
    a. Compare.
    b. Identify mappings (one-to-one/ one-to-many).
    c. Categorizing.
    d. Filtering.
5. Do the analysis according to the defined strategy.
6. Get the final outcome.

Conclusion

1. Provide summary of the datasets
    a. The dataset- Complaints data has 10,000 records.
    b. The dataset – Taxi data has 20,199 records.

    Using datamining application Weka:



2. Analyse dataset and identify find relationship between crime and taxi ride.

    By Looking at the datasets, we can assume that each complaint there can be a one or more taxi recode. So that means we can find the relationship between taxi rides data and complaints data.

When we consider the complaints data, each complaints has some considerable attributes such as start data and time, end date and time, and incident (crime) location coordinates with location name.

When we consider the taxi-ride data, It also has some considerable attributes such as pickup data and time, drop-off data and time, pickup location coordinates and drop-off location coordinates.

When looking at the data hypothetically complaint must be made by the taxi driver. So that we can assume there can be few scenarios identified.

a. Crime can be happened during the taxi ride and he complained it during the same ride.
b. Driver can make the complaint after the ride – assume within 1 hour after the ride.
c. Driver can make the complaint before the ride – driver may have seem the crime happened somewhere else and he did the complaint -assume that before 1 hour to the ride.

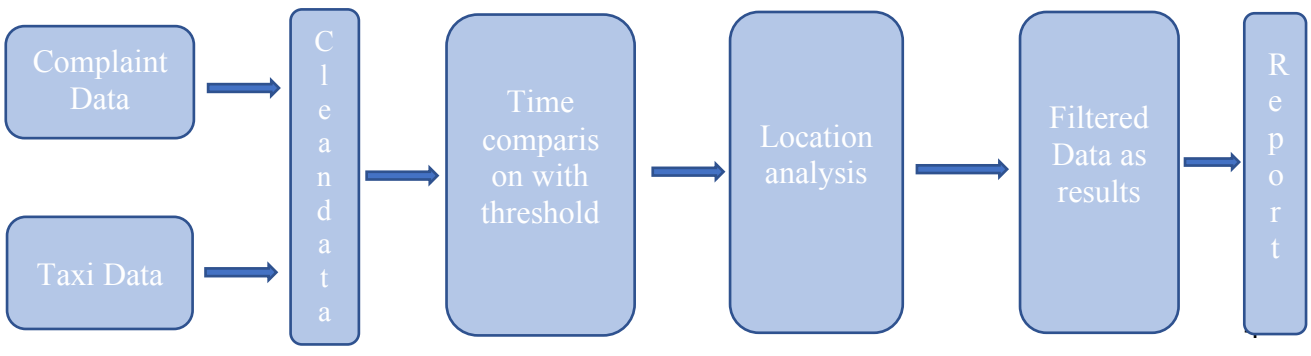
So if we consider most accurate complaint filter out from the data, it must be the complaint done during the ride. But by considering all above scenarios we can define logic with some threshold time assumption which is crime can be happened in time range 1 hour before and 1 hour after the ride.

As well as we have crime location data, so that if we consider above time frame as the time that crime happened and according to the pickup and drop-off location which nearest to the complaint data location can be identified as real crime scenario.

In this case I have setup the radius of 0.5 miles to find out the proper matching crime complaint.

So by considering the relationship between crime and taxi ride below are the only needed data to do the analysis. Other data columns will be opted from the data sets.

Complaints Data	Taxi-Rides Data
CMPLNT_NUM	lpep_pickup_datetime
CMPLNT_FR_DT	Lpep_dropoff_datetime
CMPLNT_FR_TM	Pickup_longitude
CMPLNT_TO_DT	Pickup_latitude
CMPLNT_TO_TM	Dropoff_longitude
Latitude	Dropoff_latitude
Longitude	
BORO_NM	
OFNS_DESC	



3. Using that relationship find the identity/location of criminal.

   Please see the attached result report for the all findings.

```
======================================== Crimes within radius of 0.5 miles ========================================

OTHER STATE LAWS ---> 1
MISCELLANEOUS PENAL LAW ---> 32
OFFENSES INVOLVING FRAUD ---> 5
OTHER STATE LAWS (NON PENAL LA ---> 1
DANGEROUS DRUGS ---> 76
THEFT OF SERVICES ---> 4
POSSESSION OF STOLEN PROPERTY ---> 4
FELONY ASSAULT ---> 55
JOSTLING ---> 1
OTHER OFFENSES RELATED TO THEF ---> 3
FRAUDS ---> 10
THEFT-FRAUD ---> 8
OFFENSES AGAINST THE PERSON ---> 7
GRAND LARCENY ---> 93
HARRASSMENT 2 ---> 133
ROBBERY ---> 42
ASSAULT 3 & RELATED OFFENSES ---> 110
OFF. AGNST PUB ORD SENSBLTY & ---> 31
PETIT LARCENY ---> 148
NYS LAWS-UNCLASSIFIED FELONY ---> 2
BURGLAR'S TOOLS ---> 2
ALCOHOLIC BEVERAGE CONTROL LAW ---> 2
FORGERY ---> 13
CRIMINAL TRESPASS ---> 14
SEX CRIMES ---> 3
OFFENSES AGAINST PUBLIC ADMINI ---> 26
VEHICLE AND TRAFFIC LAWS ---> 16
PROSTITUTION & RELATED OFFENSES ---> 1
GRAND LARCENY OF MOTOR VEHICLE ---> 3
INTOXICATED & IMPAIRED DRIVING ---> 12
BURGLARY ---> 17
CRIMINAL MISCHIEF & RELATED OF ---> 64
DANGEROUS WEAPONS ---> 22

======================================== Crimes within radius of 0.5 miles - end ========================================
```

4. For each crime you need to provide crime location and time, criminal pickup and drop-off location and time, information about taxi ride and short justification.

   *Please see the attached result report(**report_2016MIS021.txt**) for the filtered results.*

5. Describe and methods and methodologies used in the analysis.

   To achieve above goals, I have implemented small java program and below steps will explain the program. I have attached the program with this submission as well as you can find it from below github repository.

   Note : I have added only important steps of the code below. For all logics, please help to see the attached java program. Main Java class is `DataAnalyser.java`

   GitHub url: https://github.com/DIL8654/2016mis021

Summary of the program steps:

```
public static void main(String[] args)
{
  double radius = 0.5;

  System.out.println(" Analysing started...");
  System.out.println(" Loading data from files....");
  loadComplaintsData();
  loadTaxiData();
  System.out.println(" Loading data from files completed..!");

  System.out.println(" Mapping before analysis data by Area started..");
  findComplaintsByArea();
  System.out.println(" Mapping before analysis data by Area completed..!");

  System.out.println(" Mapping before analysis data by Crime Type started..");
  findComplaintsByCrimeType();
  System.out.println(" Mapping before analysis data by Crime Type completed..!");

  System.out.println(" Mapping Complaints and Taxi data by time started...Processing..");
  filterTaxiridesForComplaintTime();
  System.out.println(" Mapping Complaints and Taxi data by time completed..!");

  System.out.println(" Mapping Complaints and Taxi data by distance started...Processing..");
  filterTaxiridesByDistance(radius);
  System.out.println(" Mapping Complaints and Taxi data by distance completed..!");

  System.out.println(" Mapping After analysis data by Area started..");
  findComplaintsByAreaAfter();
  System.out.println(" Mapping After analysis data by Area completed..!");

  System.out.println(" Mapping After analysis data by Crime Type started..");
  findComplaintsByCrimeTypeAfter();
  System.out.println(" Mapping before analysys data by Crime Type completed..!");


  findComplaintsByCrimeTypeByRadius(radius);
  System.out.println(" printing report..!");
  printReport();
  System.out.println("Analysing completed..!");
}
```

Step 01:

Load given datasets to the data structure in the program.

```
public static void loadComplaintsData()
{
  List<String[]> data = fileReader.readFile("filtered_complaints_data.csv");
  if (data != null && !data.isEmpty())
  {
    data.remove(0); // remove header
    for (String[] record : data)
    {
      complaintsData.add(new ComplaintData(record));
    }
  }
}

public static void loadTaxiData()
{
  List<String[]> data = fileReader.readFile("filterd_taxi_data.csv");
  if (data != null && !data.isEmpty())
  {
    data.remove(0); // remove header
    for (String[] record : data)
    {
      taxiData.add(new TaxiData(record));
    }
  }
}
```

Step 02:

As a first step, below method will help to filter data according to the pickup time and drop-off time against complaints from time and to time.
So I have assume that:

a. Crime can be happened during the taxi ride and he complained it during the same ride.
b. Driver can make the complaint after the ride – assume within 1 hour after the ride.
c. Driver can make the complaint before the ride – driver may have seem the crime happened somewhere else and he did the complaint -assume that before 1 hour to the ride.

```
public static void filterTaxiridesForComplaintTime()
{

  for (ComplaintData complaint : complaintsData)
  {
    List<TaxiData> filteredTaxiDataForComplaint = new ArrayList<TaxiData>();
    Timestamp complaintfromtime = complaint.getComplaintFromTimestamp();
    Timestamp complaintTotime = complaint.getComplaintToTimeStamp();

    for (TaxiData taxidata : taxiData)
    {
      Timestamp[] range = DateTimeUtils.getFromDataAndToDateByThresholdValue(taxidata.getPickup_datetime(),
          taxidata.getDropoff_datetime(), THRESHOLD_HOURS);

      boolean isFromDateValid = DateTimeUtils.validateGivenDateInRange(range[0], range[1], complaintfromtime);
      boolean isToDateValid = DateTimeUtils.validateGivenDateInRange(range[0], range[1], complaintTotime);

      if (isFromDateValid && isToDateValid)
      {
        filteredTaxiDataForComplaint.add(taxidata);
      }
    }
    complaint.getMappedRides().addAll(filteredTaxiDataForComplaint);
  }
}
```

Steps 03:

Below method will help to calculate distance from the given locations in the datasets, That means it will calculate taxi ride's locations within 0.5 miles of radius and will assume that may be a taxi-ride matched with complaints data.

```
public static void filterTaxiridesByDistance(double radius)
{
  if (complaintsData != null && !complaintsData.isEmpty())
  {
    logger.append("======================================== Matching Complaint relation to No. of Taxi Rides within "+ radius+" miles ========================
    logger.append("\n");
    int count = 1;
    for (ComplaintData complaint : complaintsData)
    {
      findNearestTaxiRideToComplaint(complaint);
      if(complaint.getMappedRides() != null && !complaint.getMappedRides().isEmpty() && complaint.getDistanceToCrime() <= radius)
      {

        logger.append(count++).append("\t Complaints Id : "+complaint.getComplaintNo()+" -> No. of matched taxi rides :"+ complaint.getMappedRides().size());
        logger.append("\n");
        logger.append("\t \t \t Most Matching Taxi Ride : : "+complaint.getMostMatchedRide().toString());
        logger.append("\n");
        logger.append("\t \t \t Distance to Crime : "+complaint.getDistanceToCrime());
        logger.append("\n");
      }
    }
    logger.append("\n");
    logger.append("======================================== Complaint relation to No. of Taxi Rides - End ========================================\n");
    logger.append("\n");
    //System.out.println(logger.toString());
  }
}
```

Step 04:

This method will help to find out shortest distance between complaint location and taxi-ride locations. So this will consider both pickup location and drop-off location to calculate minimum distance between complaint and taxi ride.

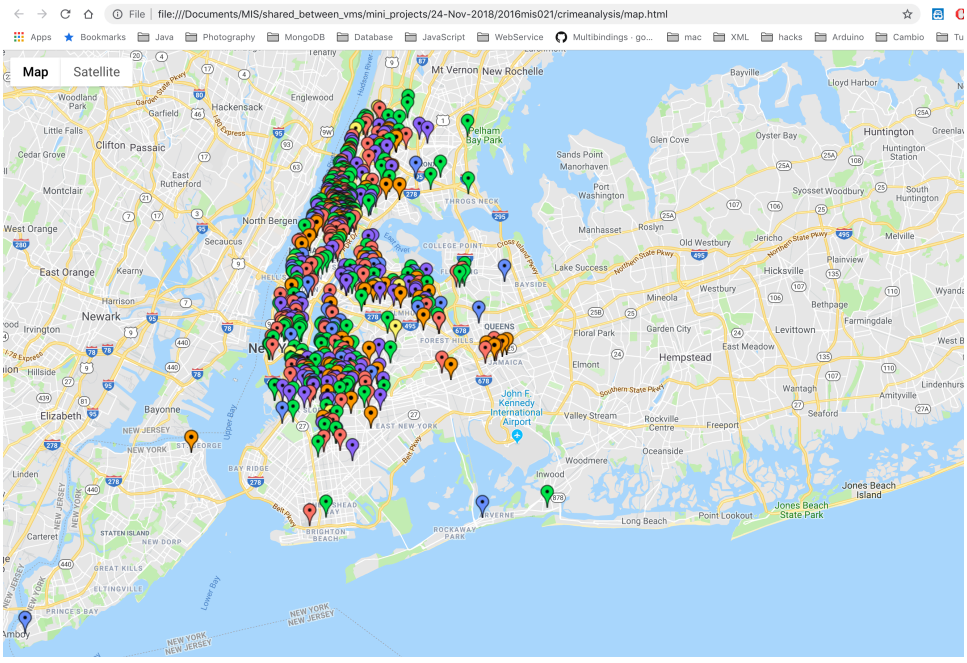
```
public static void findNearestTaxiRideToComplaint(ComplaintData complaint)
{
  List<TaxiData> filteredRides = complaint.getMappedRides();
  double shortestDistance = Double.MAX_VALUE;
  TaxiData mostMatchedride = null;

  if (filteredRides != null && !filteredRides.isEmpty())
  {
    for (TaxiData ride : filteredRides)
    {
      double pickup = LocationUtils.getDistance(ride.getPickup_latitude(), ride.getPickup_longtitude(),
          complaint.getLatitude(), complaint.getLongtitude());
      if (pickup < shortestDistance)
      {
        shortestDistance = pickup;
        mostMatchedride = ride;
      }
      double dropoff = LocationUtils.getDistance(ride.getDropoff_latitude(), ride.getDropoff_longtitude(),
          complaint.getLatitude(), complaint.getLongtitude());
      if (dropoff < shortestDistance)
      {
        shortestDistance = dropoff;
        mostMatchedride = ride;
      }
    }
    complaint.setMostMatchedRide(mostMatchedride);
    complaint.setDistanceToCrime(shortestDistance);
  }
}
```

6. Visualize your finding on New York map (You can use Python Map Visualization Library: Folium http://python-visualization.github.io/folium/). Please use different colours per each crime.

   I have tried to used python library that mentioned in the code, but seems my laptop having problem with installing “Pandas” with Folium. So below answer is based on google maps.

   But unfortunately, google maps only supports 10 different colours, so below are the assigned colours for each crime type. (some colours duplicated)

   See the map.html file for the generated map code and html template from google maps api document.



Assigned colour codes:

```
======================================== Crimes Types and assigned colors  ========================================

    OTHER STATE LAWS -->red
    MISCELLANEOUS PENAL LAW -->blue
    RAPE -->grey
    OFFENSES INVOLVING FRAUD -->green
    ADMINISTRATIVE CODE -->yellow
    OTHER STATE LAWS (NON PENAL LA -->orange
    DANGEROUS DRUGS -->green
    THEFT OF SERVICES -->grey
    AGRICULTURE & MRKTS LAW-UNCLASSIFIED -->yellow
    POSSESSION OF STOLEN PROPERTY -->purple
    FELONY ASSAULT -->black
    JOSTLING -->brown
    OTHER OFFENSES RELATED TO THEF -->blue
    OFFENSES RELATED TO CHILDREN -->orange
    FRAUDS -->yellow
    THEFT-FRAUD -->yellow
    OFFENSES AGAINST THE PERSON -->black
    nan -->grey
    ARSON -->grey
    FRAUDULENT ACCOSTING -->while
    GRAND LARCENY -->while
    CHILD ABANDONMENT/NON SUPPORT -->brown
    HARRASSMENT 2 -->green
    ROBBERY -->orange
    ASSAULT 3 & RELATED OFFENSES -->purple
    OFF. AGNST PUB ORD SENSBLTY & -->grey
    PETIT LARCENY -->red
    ENDAN WELFARE INCOMP -->black
    BURGLAR'S TOOLS -->purple
    NYS LAWS-UNCLASSIFIED FELONY -->while
    MURDER & NON-NEGL. MANSLAUGHTER -->green
    ALCOHOLIC BEVERAGE CONTROL LAW -->blue
    FORGERY -->red
    CRIMINAL TRESPASS -->brown
    UNAUTHORIZED USE OF A VEHICLE -->black
    SEX CRIMES -->brown
    OFFENSES AGAINST PUBLIC ADMINI -->red
    GAMBLING -->red
    VEHICLE AND TRAFFIC LAWS -->while
    GRAND LARCENY OF MOTOR VEHICLE -->orange
    PROSTITUTION & RELATED OFFENSES -->purple
    INTOXICATED & IMPAIRED DRIVING -->blue
    BURGLARY -->orange
    KIDNAPPING -->green
    CRIMINAL MISCHIEF & RELATED OF -->brown
    DANGEROUS WEAPONS -->purple
======================================== Crimes Types and assigned colors  ========================================
```