BIG DATA ANALYTICS

ITE-2013

J-Component

"Crime Rate Analysis using Hadoop and Python"

17BIT0336-Shubhang

17BIT0036-Amandeep Singh

17BIT0238-Jay Paliwal

Problem Statement

Crimes have been severely increased in past few years, the Problem Statement includes analysis of crimes with different perspectives including utmost attributes possible and predicting via the study of nature of crimes committed. The problem statement is described to initially predict the crime-type based on location and time. We worked on data about historical crimes in California.

We had close to 13,000 records of crimes with data on the date and time of the crime, its location, and its type. Common types of crime include theft, criminal damage, criminal trespass, and assault. This project took on the task of predicting the type of crime that was committed given a police report in two ways one according to time that is when crime took place and another is location that is where crime took place. From a small number of overly detailed features, in time it will give the detail that at which time slot which crime is maximum and in location it will tell at which place which type of crime is maximum. They then trained various diagram based models (Graphs and Pie charts) to classify crimes by type using the generated features.

Finally, they tested the performance of their models on testing data. They conclude that predicting the type of crimes committed by time and location alone is quite difficult, but that the feature engineering greatly increases predictive power. Predictions will be made to provide local authorities with an upper hand on crime and help them plan a better strategy to tackle the same.

Introduction

At a glance, here's what we did:

- *Data Set Collection*
- *Implementation of our code*
- *Displaying results according to different objectives*

Data was collected from [https://data.cityofchicago.org/Public-Safety/Crimes-2001-to-present/ijzp-q8t2](2017 Dataset Crime). Data was pre-processed and cleaned to remove missing values and garbage values at various positions. Then, it was given as input to our code in mapper code. The output from mapper was sent to reducer code and the corresponding results were printed. The results were saved in a text file as the code was run in Hadoop. The part of code was then selected and pasted in excel file. Corresponding graphs were created to display the results more beautifully.

Output of the code:

- ***Based on Time of Crime*** *– Analysis and predictions will be based on time of crimes i.e. which time have the maximum crime rates and needs to be inspected more efficiently.*

- ***Based on Location of Crime*** *– The most Crime prone locations will provide local authorities to target specific area clusters to counter crime.*

- ***Based on District of Crime****– The most Crime prone Districts will provide local authorities to target specific area clusters to counter crime.*

Software/Hardware Requirements:

Tools used for this project:

- *VMware Virtual Workstation*
- *Windows and Cloudera OS*
- *Rapid Miner*
- *Hadoop HDFS*
- *Python*
- *Microsoft Excel*

DATA PREPARATION PHASE

2.a)

DATA SOURCE:

https://data.cityofchicago.org/Public-Safety/Crimes-2001-to-present/ijzp-q8t2

DATA SET DESCRIPTION:

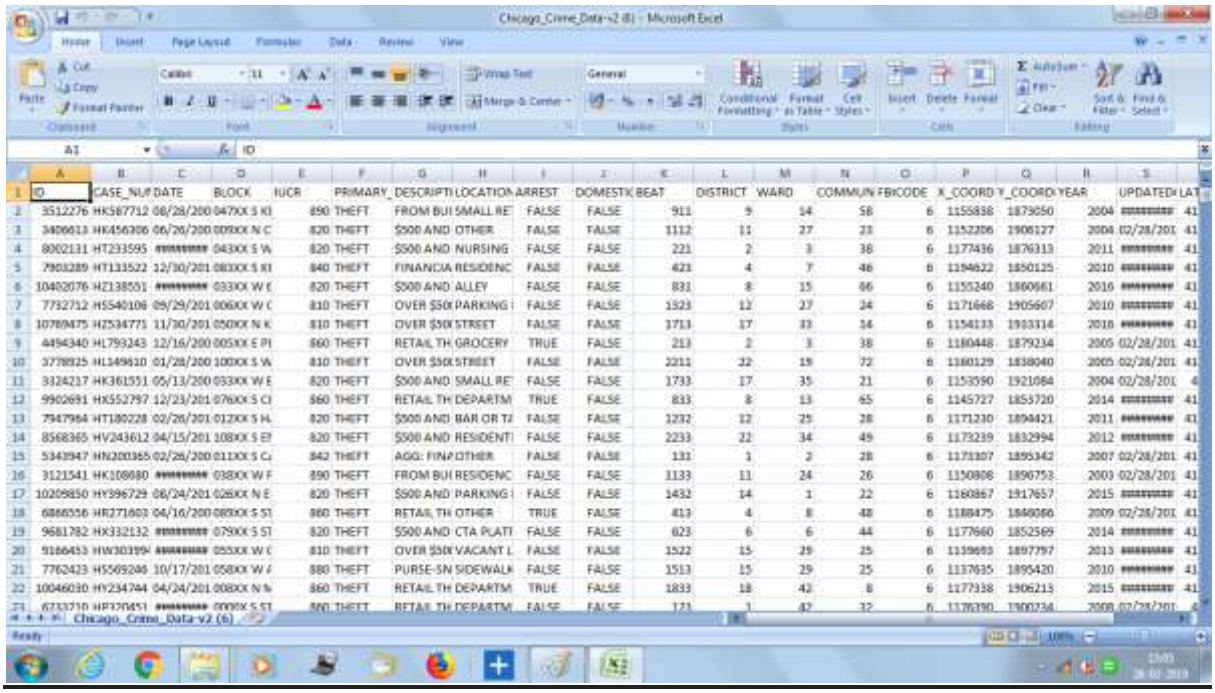
Dataset contains the following attributes:

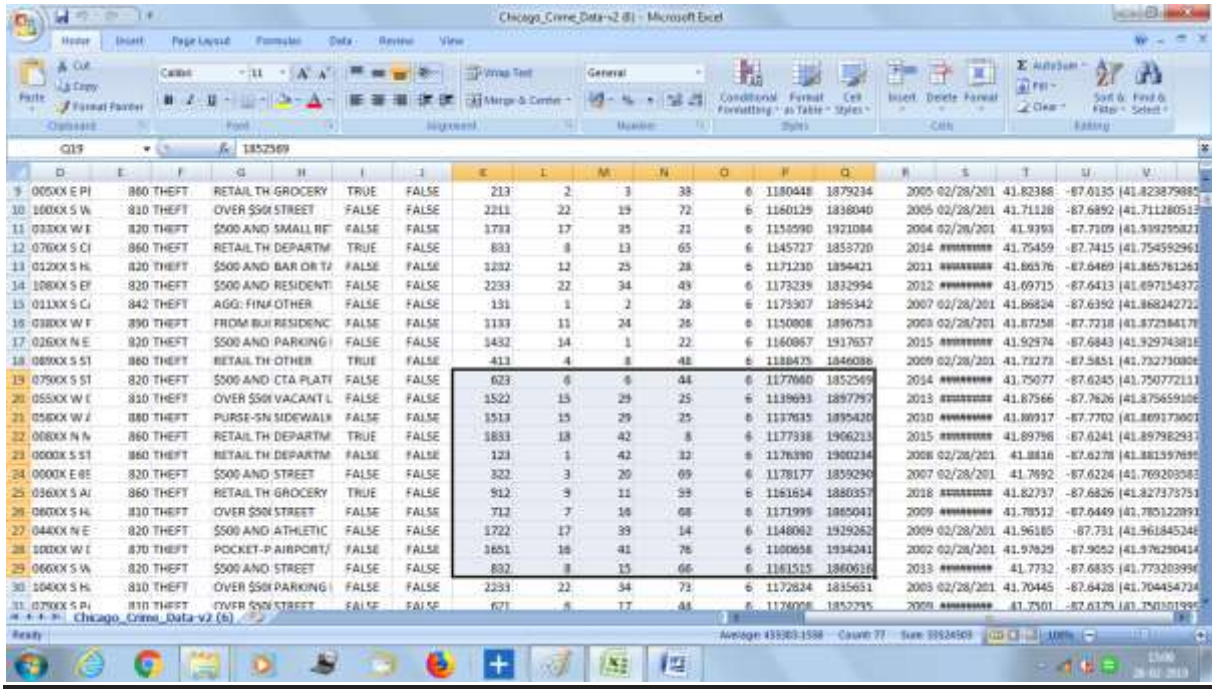
1. *ID*
2. *Case Number*
3. *Date*
4. *Block*
5. *IUCR*
6. *Primary Type Description*
7. *Location Description*
8. *Arrest*
9. *Domestic*
10. *Beat*
11. *District Ward*
12. *Community Area*
13. *FBI Code*
14. *X Coordinate*
15. *Y Coordinate*
16. *Year*
17. *Updated On*
18. *Latitude*
19. *Longitude*
20. *Location*

Column Name	Description	Type
ID	Unique identifier for the record.	Number
Case Number	The Chicago Police Department RD Number (Records Division Number), which is unique to the incident.	Plain Text
Date	Date when the incident occurred. this is sometimes a best estimate.	Date & Time
Block	The partially redacted address where the incident occurred, placing it on the same block as the actual address.	Plain Text
IUCR	The Illinois Unifrom Crime Reporting code. This is directly linked to the Primary Type and Description. See the list of IUCR codes at https://data.cityofchicago.org/d/c7ck-438e.	Plain Text
Primary Type	The primary description of the IUCR code.	Plain Text
Description	The secondary description of the IUCR code, a subcategory of the primary description.	Plain Text
Location Description	Description of the location where the incident occurred.	Plain Text
Arrest	Indicates whether an arrest was made.	Checkbox
Domestic	Indicates whether the incident was domestic-related as defined by the Illinois Domestic Violence Act.	Checkbox
Beat	Indicates the beat where the incident occurred. A beat is the smallest police geographic area – each beat has a dedicated police beat car. Three to five beats make up a police sector, and three sectors make up a police district. The Chicago Police Department has 22 police districts. See the beats at https://data.cityofchicago.org/d/aerh-rz74.	Plain Text
District	Indicates the police district where the incident occurred. See the districts at https://data.cityofchicago.org/d/fthy-xz3r.	Plain Text

Ward	*The ward (City Council district) where the incident occurred. See the wards at https://data.cityofchicago.org/d/sp34-6z76.*	*Number*	
Community Area	*Indicates the community area where the incident occurred. Chicago has 77 community areas. See the community areas at https://data.cityofchicago.org/d/cauq-8yn6.*	*Plain Text*	
FBI Code	*Indicates the crime classification as outlined in the FBI's National Incident-Based Reporting System (NIBRS). See the Chicago Police Department listing of these classifications at http://gis.chicagopolice.org/clearmap_crime_sums/crime_types.html.*	*Plain Text*	
X Coordinate	*The x coordinate of the location where the incident occurred in State Plane Illinois East NAD 1983 projection. This location is shifted from the actual location for partial redaction but falls on the same block.*	*Number*	
Y Coordinate	*The y coordinate of the location where the incident occurred in State Plane Illinois East NAD 1983 projection. This location is shifted from the actual location for partial redaction but falls on the same block.*	*Number*	
Year	*Year the incident occurred.*	*Number*	
Updated On	*Date and time the record was last updated.*	*Date & Time*	
Latitude	*The latitude of the location where the incident occurred. This location is shifted from the actual location for partial redaction but falls on the same block.*	*Number*	
Longitude	*The longitude of the location where the incident occurred. This location is shifted from the actual location for partial redaction but falls on the same block.*	*Number*	
Location	*The location where the incident occurred in a format that allows for creation of maps and other geographic operations on this data portal. This location is shifted from the actual location for partial redaction but falls on the same block.*	*Location*	

DATASET SCREENSHOTS:





2.b)

```
In [1]: import pandas as pd
```

```
In [2]: df=pd.read_csv('Desktop/crimedata.csv')
        df.head()
```

Out[2]:

	ID	CASE_NUMBER	DATE	BLOCK	IUCR	PRIMARY_TYPE	DESCRIPTION	LOCATION_DESCRIPTION	ARREST	DOMESTIC	...	WARD	COMM
0	3512276	HK587712	08/28/2004 05:50:58 PM	047XX S KEDZIE AVE	890	THEFT	FROM BUILDING	SMALL RETAIL STORE	False	False	...	14.0	
1	3406613	HK456306	06/26/2004 12:40:00 PM	009XX N CENTRAL PARK AVE	820	THEFT	$500 AND UNDER	OTHER	False	False	...	27.0	
2	8002131	HT233696	04-04-2011 5:45	043XX S WABASH AVE	820	THEFT	$500 AND UNDER	NURSING HOME/RETIREMENT HOME	False	False	...	3.0	
3	7903289	HT133522	12/30/2010 04:30:00 PM	083XX S KINGSTON AVE	840	THEFT	FINANCIAL ID THEFT| OVER $300	RESIDENCE	False	False	...	7.0	
4	10402076	HZ138551	02-02-2016 19:30	033XX W 66TH ST	820	THEFT	$500 AND UNDER	ALLEY	False	False	...	15.0	

5 rows × 22 columns

Structural analysis:

```
In [3]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 533 entries, 0 to 532
Data columns (total 22 columns):
ID                      533 non-null int64
CASE_NUMBER             533 non-null object
DATE                    533 non-null object
BLOCK                   533 non-null object
IUCR                    533 non-null object
PRIMARY_TYPE            533 non-null object
DESCRIPTION             533 non-null object
LOCATION_DESCRIPTION    533 non-null object
ARREST                  533 non-null bool
DOMESTIC                533 non-null bool
BEAT                    533 non-null int64
DISTRICT                533 non-null int64
WARD                    490 non-null float64
COMMUNITY_AREA_NUMBER   490 non-null float64
FBICODE                 533 non-null object
X_COORDINATE            529 non-null float64
Y_COORDINATE            529 non-null float64
YEAR                    533 non-null int64
UPDATEDON               533 non-null object
LATITUDE                529 non-null float64
LONGITUDE               529 non-null float64
LOCATION                529 non-null object
dtypes: bool(2), float64(6), int64(4), object(10)
memory usage: 84.4+ KB
```

Statistical analysis:

```
[4]: handle.describe()
```

[4]:

	ID	BEAT	DISTRICT	WARD	COMMUNITY_AREA_NUMBER	X_COORDINATE	Y_COORDINATE	YEAR	LATITUDE	L
count	5.330000e+02	533.000000	533.000000	490.000000	490.000000	5.290000e+02	5.290000e+02	533.00000	529.000000	
mean	6.228301e+06	1187.037523	11.195122	22.608163	37.542857	1.162876e+06	1.886223e+06	2008.34334	41.843429	
std	3.027454e+06	660.592263	6.496631	13.080119	21.409514	1.650197e+04	3.009980e+04	4.94940	0.082798	
min	2.114900e+04	111.000000	1.000000	1.000000	1.000000	1.100658e+06	1.814512e+06	2001.00000	41.645796	
25%	3.420516e+06	711.000000	6.000000	12.000000	23.000000	1.151715e+06	1.860454e+06	2004.00000	41.772556	
50%	6.231712e+06	1111.000000	10.000000	24.000000	30.000000	1.163235e+06	1.891447e+06	2008.00000	41.857737	
75%	8.875058e+06	1634.000000	16.000000	32.000000	58.000000	1.174618e+06	1.907711e+06	2012.00000	41.902298	
max	1.127717e+07	2535.000000	25.000000	50.000000	77.000000	1.204126e+06	1.951001e+06	2018.00000	42.021178	

BEFORE CLEANING:

```
In [4]: file1 = open("Desktop/crimedata.txt","r")
        for i in range(0,534):
            print(file1.readline().split(",")[11])
```

```
12
3
11
25
FALSE
10
10
25
25
4
11
19
19
25
11
3
6
7
2
7
```

```
In [7]: file2=open("untitled.txt","r")
        print(file2.read())
```

```
ID,CASE_NUMBER,DATE,BLOCK,IUCR,PRIMARY_TYPE,DESCRIPTION,LOCATION_DESCRIPTION,ARREST,DOMESTIC,BEAT,DISTRICT,WARD,COMMUNITY_ARE
A_NUMBER,FBICODE,X_COORDINATE,Y_COORDINATE,YEAR,UPDATEDON,LATITUDE,LONGITUDE,LOCATION
3512276,HK587712,08/28/2004 05:50:56 PM,047XX S KEDZIE AVE,890,THEFT,FROM BUILDING,SMALL RETAIL STORE,FALSE,FALSE,911,9,14,5
8,6,1155838,1873050,2004,02-10-2018 15:50,41.8074405,-87.70395585,(41.8074405, -87.703955849)
3406613,HK456306,06/26/2004 12:40:00 PM,009XX N CENTRAL PARK AVE,820,THEFT,$500 AND UNDER,OTHER,FALSE,FALSE,1112,11,27,23,6,1
152206,1906127,2004,02/28/2018 03:56:25 PM,41.89827996,-87.71640551,(41.898279962, -87.716405505)
8002131,HT233595,04-04-2011 5:45,043XX S WABASH AVE,820,THEFT,$500 AND UNDER,NURSING HOME|RETIREMENT HOME,FALSE,FALSE,221,2,
3,38,6,1177436,1876313,2011,02-10-2018 15:50,41.81593313,-87.62464213,(41.815933131, -87.624642127)
7903289,HT133522,12/30/2010 04:30:00 PM,083XX S KINGSTON AVE,840,THEFT,FINANCIAL ID THEFT| OVER $300,RESIDENCE,FALSE,FALSE,42
3,4,7,46,6,1194622,1850125,2010,02-10-2018 15:50,41.74366532,-87.56246276,(41.743665322, -87.562462756)
10402076,HZ138551,02-02-2016 19:30,033XX W 66TH ST,820,THEFT,$500 AND UNDER,ALLEY,FALSE,FALSE,831,8,15,66,6,1155240,1860661,2
016,02-10-2018 15:50,41.7734553,-87.70648047,(41.773455295, -87.706480471)
7732712,HS540106,09/29/2010 07:59:00 AM,006XX W CHICAGO AVE,810,THEFT,OVER $500,PARKING LOT|GARAGE(NON RESID ),FALSE,FALSE,13
23,12,27,24,6,1171668,1905607,2010,02-10-2018 15:50,41.89644677,-87.64493868,(41.896446772, -87.644938678)
10769475,HZ534771,11/30/2016 01:15:00 AM,050XX N KEDZIE AVE,810,THEFT,OVER $500,STREET,FALSE,FALSE,1713,17,33,14,6,1154133,19
33314,2016,02-10-2018 15:50,41.97284491,-87.70860008,(41.972844913, -87.708600079)
4494340,HL793243,12/16/2005 04:45:00 PM,005XX E PERSHING RD,860,THEFT,RETAIL THEFT,GROCERY FOOD STORE,TRUE,FALSE,213,2,3,38,
6,1180448,1879234,2005,02/28/2018 03:56:25 PM,41.82387989,-87.61350386,(41.823879885, -87.613503857)
3778925,HL140610,01/28/2005 05:00:00 PM,100XX S WASHTENAW AVE,810,THEFT,OVER $500,STREET,FALSE,FALSE,2211,22,19,72,6,1160129,
1838040,2005,01/28/2018 03:56:25 PM,41.71138051,-87.6891791,(41.711380513, -87.689179097)
```

DATE FORMAT

C
DATE
08/28/2004 05:50:56 PM
06/26/2004 12:40:00 PM
04-04-2011 5:45
12/30/2010 04:30:00 PM
02-02-2016 19:30
09/29/2010 07:59:00 AM
11/30/2016 01:15:00 AM
12/16/2005 04:45:00 PM
01/28/2005 05:00:00 PM
05/13/2004 02:15:00 PM
12/23/2014 02:15:00 PM
02/26/2011 04:00:00 PM
04/15/2012 07:30:00 PM
02/26/2007 04:00:00 PM
01-05-2003 18:30
08/24/2015 07:30:00 PM
04/16/2009 05:30:00 PM
07-04-2014 21:45
06-04-2013 9:00
10/17/2010 06:00:00 PM
04/24/2015 07:55:00 PM
05-06-2008 15:20
07/22/2007 07:25:00 AM
02/14/2018 11:15:00 AM
10-11-2009 19:00
09/13/2009 10:30:00 PM
07-03-2002 20:00
09-07-2013 2:30
08/13/2003 08:00:00 PM

LOCATION DESCRIPTION

H
LOCATION_DESCRIPTION
SMALL RETAIL STORE
OTHER
NURSING HOME,RETIREM
RESIDENCE
ALLEY
PARKING LOT,GARAGE(N
STREET
GROCERY FOOD STORE
STREET
SMALL RETAIL STORE
DEPARTMENT STORE
BAR OR TAVERN
RESIDENTIAL YARD (FRO
OTHER
RESIDENCE PORCH,HALL
PARKING LOT,GARAGE(N
OTHER
CTA PLATFORM
VACANT LOT,LAND
SIDEWALK
DEPARTMENT STORE
DEPARTMENT STORE
STREET
GROCERY FOOD STORE
STREET
ATHLETIC CLUB
AIRPORT,AIRCRAFT
STREET
PARKING LOT,GARAGE(N

AFTER CLEANING

H
LOCATION_DESCRIPTION
SMALL RETAIL STORE
OTHER
NURSING HOME|RETIRE
RESIDENCE
ALLEY
PARKING LOT|GARAGE(N
STREET
GROCERY FOOD STORE
STREET
SMALL RETAIL STORE
DEPARTMENT STORE
BAR OR TAVERN
RESIDENTIAL YARD (FRO
OTHER
RESIDENCE PORCH|HALL
PARKING LOT|GARAGE(N
OTHER
CTA PLATFORM
VACANT LOT|LAND
SIDEWALK
DEPARTMENT STORE
DEPARTMENT STORE
STREET
GROCERY FOOD STORE
STREET
ATHLETIC CLUB
AIRPORT|AIRCRAFT
STREET
PARKING LOT|GARAGE(N

TIME CLEANING CODE

```
In [9]: file1 = open("untitled.txt","r")
arr=[]
for i in range(0,534):
    k=file1.readline().split(",")[2]
    time=k.split(" ")
    if (time[len(time)-1]=="AM" or time[len(time)-1]=="PM"):
        hhmmss=time[1].split(":")
        hour=hhmmss[0]
        if (time[len(time)-1]=="PM"):
            if(hour=='12'):
                hour=hour
            else:
                hour=int(hour)+12
            hhmmss[0]=str(hour)
            time[1]=":".join(hhmmss)
            time.pop()
            time="||".join(time)

        if (time[len(time)-1]=="AM"):
            if(hour==12):
                hour='00'
            time.pop()
            time="||".join(time)
    else:
        date=time[0].split('-')
        time[0]="/".join(date)
        time="||".join(time)
    arr.append(time)
    print(time)
```

NEW FORMAT OF DATE:

```
DATE
08/28/2004||17:50:56
06/26/2004||12:40:00
04/04/2011||5:45
12/30/2010||16:30:00
02/02/2016||19:30
09/29/2010||07:59:00
11/30/2016||01:15:00
12/16/2005||16:45:00
01/28/2005||17:00:00
05/13/2004||14:15:00
12/23/2014||14:15:00
02/26/2011||16:00:00
04/15/2012||19:30:00
02/26/2007||16:00:00
01/05/2003||18:30
08/24/2015||19:30:00
04/16/2009||17:30:00
07/04/2014||21:45
06/04/2011||0:00
```

COPYING EVERYTHING TO NEW TEXT FILE:

```
In [10]: file1 = open("untitled.txt","r+")
file2 = open("Desktop/crimedata_afterprep.txt","r+")
for i in range(0,534):
    k=file1.readline().split(",")
    k[2]=arr[i]
    k=','.join(k)
    file2.write(k)
    print(k)
```

```
ID,CASE_NUMBER,DATE,BLOCK,IUCR,PRIMARY_TYPE,DESCRIPTION,LOCATION_DESCRIPTION,ARREST,DOMESTIC,BEAT,DISTRICT,WARD,COMMUNITY_ARE
A_NUMBER,FBICODE,X_COORDINATE,Y_COORDINATE,YEAR,UPDATEDON,LATITUDE,LONGITUDE,LOCATION

3512276,HK587712,08/28/2004||17:50:56,047XX S KEDZIE AVE,890,THEFT,FROM BUILDING,SMALL RETAIL STORE,FALSE,FALSE,911,9,14,58,
6,1155838,1873850,2004,02-10-2018 15:50,41.8074405,-87.70395585,(41.8074405, -87.703955849)

3406613,HK456306,06/26/2004||12:40:00,009XX N CENTRAL PARK AVE,820,THEFT,$500 AND UNDER,OTHER,FALSE,FALSE,1112,11,27,23,6,115
2206,1906127,2004,02/28/2018 03:56:25 PM,41.89827996,-87.71640551,(41.898279962, -87.716405505)

8002131,HT233595,04/04/2011||5:45,043XX S WABASH AVE,820,THEFT,$500 AND UNDER,NURSING HOME|RETIREMENT HOME,FALSE,FALSE,221,2,
3,38,6,1177436,1876313,2011,02-10-2018 15:50,41.81593313,-87.62464213,(41.815933131, -87.624642127)

7903289,HT133522,12/30/2010||16:30:00,083XX S KINGSTON AVE,840,THEFT,FINANCIAL ID THEFT| OVER $300,RESIDENCE,FALSE,FALSE,423,
4,7,46,6,1194622,1850125,2010,02-10-2018 15:50,41.74366532,-87.56246276,(41.743665322, -87.562462756)

10402076,HZ138551,02/02/2016||19:30,033XX W 66TH ST,820,THEFT,$500 AND UNDER,ALLEY,FALSE,FALSE,831,8,15,66,6,1155240,1860661,
2016,02-10-2018 15:50,41.7734553,-87.70648047,(41.773455295, -87.706480471)

7732712,HS540106,09/29/2010||07:59:00,006XX W CHICAGO AVE,810,THEFT,OVER $500,PARKING LOT|GARAGE(NON RESID ),FALSE,FALSE,132
```

CONVERTING TEXT FILE TO CSV FOR FUTURE REFERNCE:

In [104]:
```
import csv
csv_file = 'Desktop/crimedata.csv'
txt_file = 'untitled.txt'
with open(txt_file, "w") as my_output_file:
    with open(csv_file, "r") as my_input_file:
        [ my_output_file.write(",".join(row)+'\n') for row in csv.reader(my_input_file)]
    my_output_file.close()
```

Model Planning:

The given analysis consists of three modules of data analysis namely location, time and district.

Each module consists of an individual mapper and reducer functions and data visualisation outputs. The functions of the given modules are as follows:

- ***LOCATION BASED***

  The mapper functions used in this module uses the x-coordinate and y-coordinates of the place where the crime is registered.

  The function links each line to its coordinates and passes on as input to the reducer function which in turn counts the occurrence of each crime in each area and displays the results area wise with numbers associated with each crime.

- ***TIME BASED***

  The mapper functions used in this module uses the time at which the crime is registered. The hour component of the time is extracted and divided into 8 time slots and forwarded to the reducer where further analysis takes place.

  The function links each line to its hour component and the reducer function counts the occurrence of each crime in every

time slot displays the results time slot wise with numbers associated with each crime.

- ***DISTRICT BASED***

  The mapper functions used in this module uses the district of the place where the crime is registered.

  The function links each line to its district and passes on as input to the reducer function which in turn counts the occurrence of each crime in each area and displays the results district wise with numbers associated with each crime.

DIAGRAM:


P
X_COORD
1155838
1152206
1177436
1194622
Q
Y_COORDI
1873050
1906127
1876313
1850125
THEFT=n1
RAPE=n2
BURGLARY=n3
NARCOTICS=n4
ASSAULT=n5
OUTPUT

Pseudocodes:

Location mapper:

- *Start*
- *Input the .csv file*
- *using a for loop for taking each line for processing*
- *Split each line using split() function*
- *Store x and y coordinate from each array which is result of split.*
- *Map (x,y) coordinate to the line.*

Location reducer:

- *Start*
- *Input the from mapper.py*
- *using a for loop for taking each line for processing*
- *Split each line using split() function*
- *Obtain x,y coordinate and tuple in variables.*
- *Divide according to range of x,y coordinates in 5 areas.*
- *For each area distinguish crime type and their count.*
- *In each area, get total crime and type of crime most committed.*
- *Display results.*

District mapper:

- *Start*
- *Input the .csv file*
- *using a for loop for taking each line for processing*
- *Split each line using split() function*
- *Store the district from each array which is result of split.*
- *Map district to the line.*

District reducer:

- *Start*
- *Input the from mapper.py*
- *using a for loop for taking each line for processing*
- *Split each line using split() function*
- *Obtain district and tuple in variables.*
- *Divide according to range of x,y coordinates in 5 areas.*
- *For each district distinguish crime type and their count.*
- *In each district, get total crime and type of crime most committed.*

Display results.

Time mapper:

- *Start*
- *Input the .csv file*
- *using a for loop for taking each line for processing*
- *Split each line using split() function*
- *Take time attribute in a variable and split it to obtain hour of crime.*
- *Store hour from each array which is result of split.*
- *Map hour component to the line.*

Time reducer:

- *Start*
- *Input the from mapper.py*
- *using a for loop for taking each line for processing*
- *Split each line using split() function*
- *Obtain hour component and tuple in variables.*
- *Divide according to range of hour component into 8 time slots.*
- *For each time slot distinguish crime type and count.*
- *In each time slot, get total crime and type of crime most committed.*

Display results.

MODEL BUILDING:

We have built mapper and reducer code identifying various sectors of dataset which analytics would be helpful for the society and Chicago government.

TIME BASED ANALYSIS:

TIME_MAP

```
#!/usr/bin/python

import sys

flag=0

for input_line in sys.stdin:
        line = input_line.strip().split(",")

        if (flag==0):
                columns=line
                flag=1
        else:
                time = line[2].split("||")[1].split(":")
                hour = float(time[0])
                print "{0}\t{1}".format(hour,str(input_line.strip()))
```

TIME_REDUCE:

```
#!/usr/bin/python
# coding: utf-8

import sys
quad = []
tab = []
index=-1
count=0

theft="THEFT"

for i in range(8):
        quad.append([])
        tab.append([])

for input_line in sys.stdin:

        line = input_line.strip().split("\t")          # Time \t Tuple
        time = str(line[0])        #time
        count=count+1

        #print 'count"+str(count)

        hour = 0 + float(time)

        if hour>=0 and hour<3:
                quad[0].append(line[1])
        elif hour>=3 and hour<6:
                quad[1].append(line[1])
        elif hour>=6 and hour<9:
                quad[2].append(line[1])
        elif hour>=9 and hour<12:
                quad[3].append(line[1])
```

```
        elif hour>=12 and hour<15:
                quad[4].append(line[1])
        elif hour>=15 and hour<18:
                quad[5].append(line[1])
        elif hour>=18 and hour<21:
                quad[6].append(line[1])
        elif hour>=21 and hour<24:
                quad[7].append(line[1])

for i in range(0,8):
        theft_c = 0
        criminal_c=0
        burglary_c=0
        assault_c=0
        narcotics_c=0

        if i==0:
            time_slot="00:00 – 02:59"
        if i==1:
            time_slot="03:00 – 05:59"
        if i==2:
            time_slot="06:00 – 08:59"
        if i==3:
            time_slot="09:00 – 11:59"
        if i==4:
            time_slot="12:00 – 14:59"
        if i==5:
            time_slot="15:00 – 17:59"
        if i==6:
            time_slot="18:00 – 20:59"
        if i==7:
            time_slot="21:00 – 23:59"

        for j in range(len(quad[i])):
                crime_array = quad[i][j].strip().split(",")
```

```
            crime_type = str(crime_array[5])

            if crime_type == "THEFT":
               theft_c = theft_c + 1
            elif crime_type=="CRIMINAL DAMAGE":
               criminal_c = criminal_c + 1
            elif crime_type=="BURGLARY":
               burglary_c = burglary_c + 1
            elif crime_type=="ASSAULT":
               assault_c = assault_c + 1
            elif crime_type=="NARCOTICS":
               narcotics_c = narcotics_c + 1

        total_crimes = theft_c + criminal_c + burglary_c + assault_c +
narcotics_c

        print '\n\nFor Time slot { ' + time_slot + ' } Occurrence of Crime
types are:  '

        print '------------------------------------------------------'

        print '\t\tTHEFT\t\t\t: '+str(theft_c)
        print '\t\tCRIMINAL DAMAGE\t\t: '+str(criminal_c)
        print '\t\tBURGLARY\t\t: '+str(burglary_c)
        print '\t\tASSAULT\t\t\t: '+str(assault_c)
        print '\t\tNARCOTICS\t\t: '+str(narcotics_c)

        most = max(int(theft_c),int(criminal_c), int(burglary_c),
int(assault_c), int(narcotics_c))

        if most==theft_c:
            most_crime="THEFT"

        elif most==criminal_c:
            most_crime="CRIMINAL DAMAGE"
```

```
        elif most==burglary_c:
            most_crime="BURGLARY"
        elif most==assault_c:
            most_crime="ASSAULT"
        elif most==narcotics_c:
            most_crime="NARCOTICS"

        print '\n\t\t=> Total Crimes ccurred and reported in above time
slot is : ' + str(total_crimes) + ' reports.'

        print '\n\t\t=> Also the most ccurred crime in above time slot is :
' + most_crime  + str(most) + ' times.'

        tab[i].append(theft_c)
        tab[i].append(criminal_c)
        tab[i].append(burglary_c)
        tab[i].append(assault_c)
        tab[i].append(narcotics_c)

print
'\n\n**************************************************
**********************************************'

print
'\n\t\t\tTHEFT\t\tCRIMINAL\tBURGLARY\tASSAULT\t\tNARCOTICS'

for i in range(0,8):

        #for j in range(0,5):

        print                                                    "\nSlot
"+str(i+1)+"\t\t\t"+str(tab[i][0])+"\t\t"+str(tab[i][1])+"\t\t"+str(tab[i][
2])+"\t\t"+str(tab[i][3])+"\t\t"+str(tab[i][4])

print '\n\t* Total number of Tuples analysed are : ' + str(count)
```

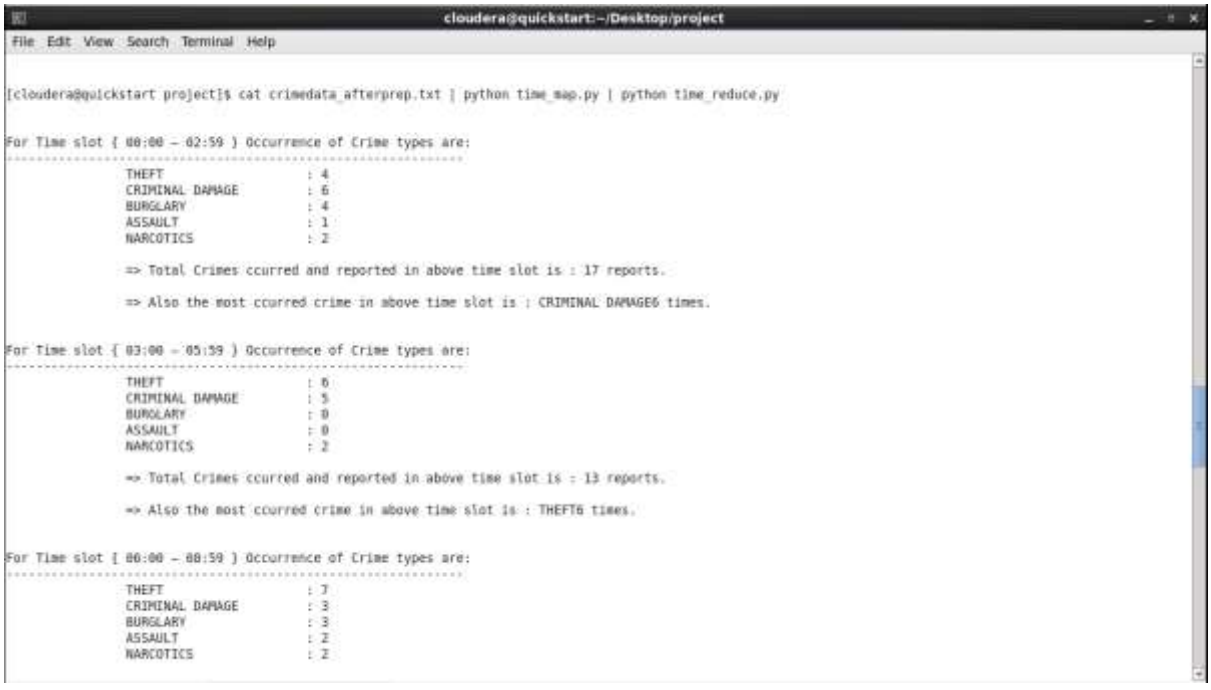
print
*'\n**********\n\n'*

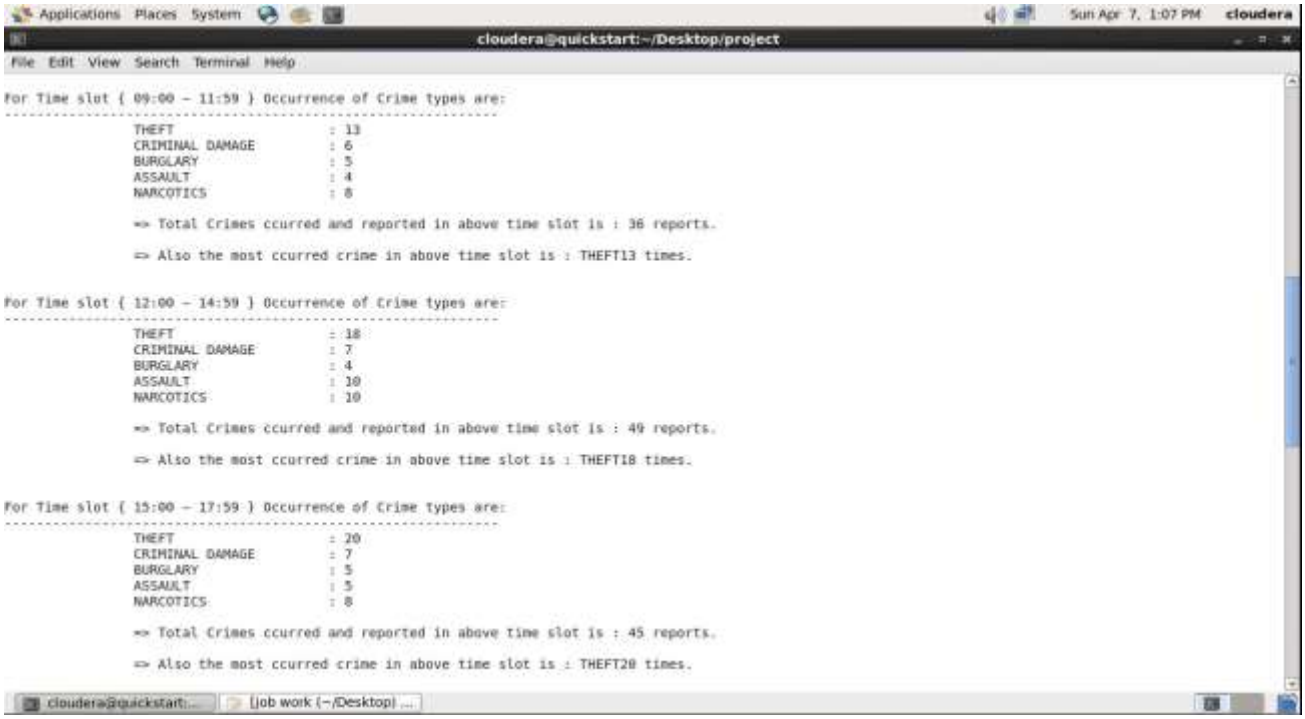
OUTPUT:

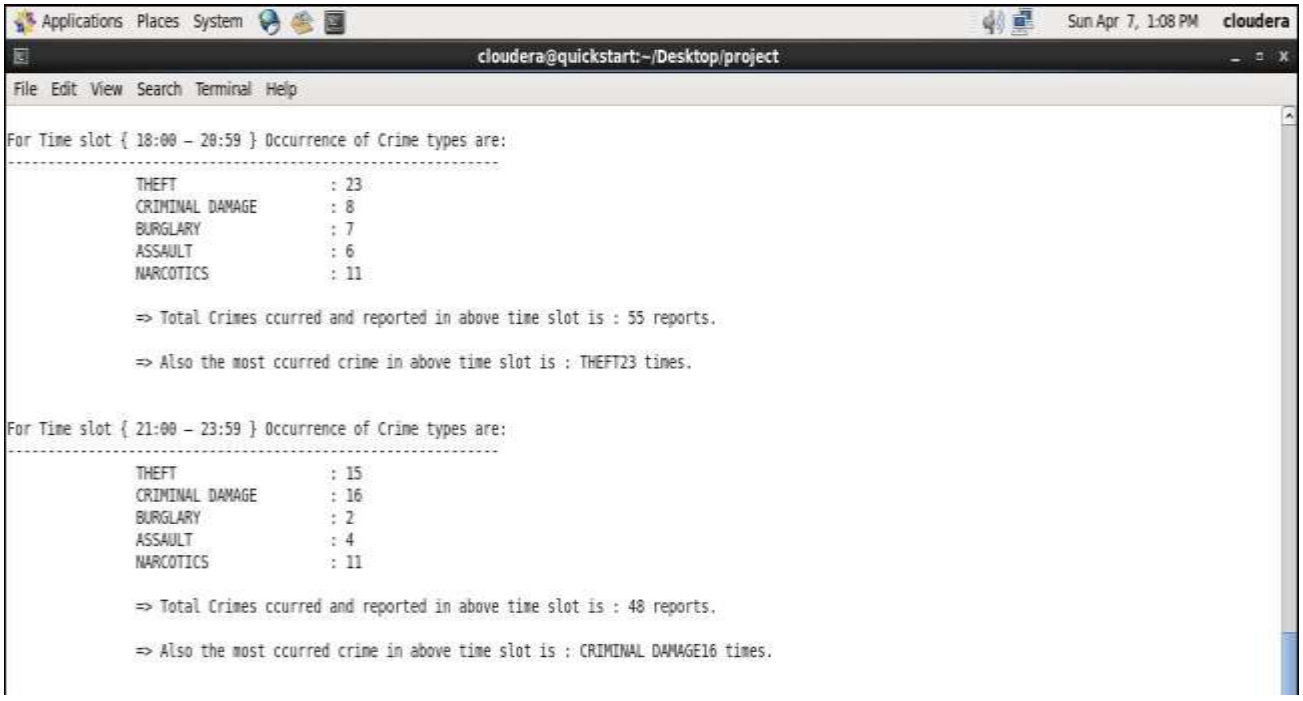
LOCAL EXECUTION:

COMMANDS USED:

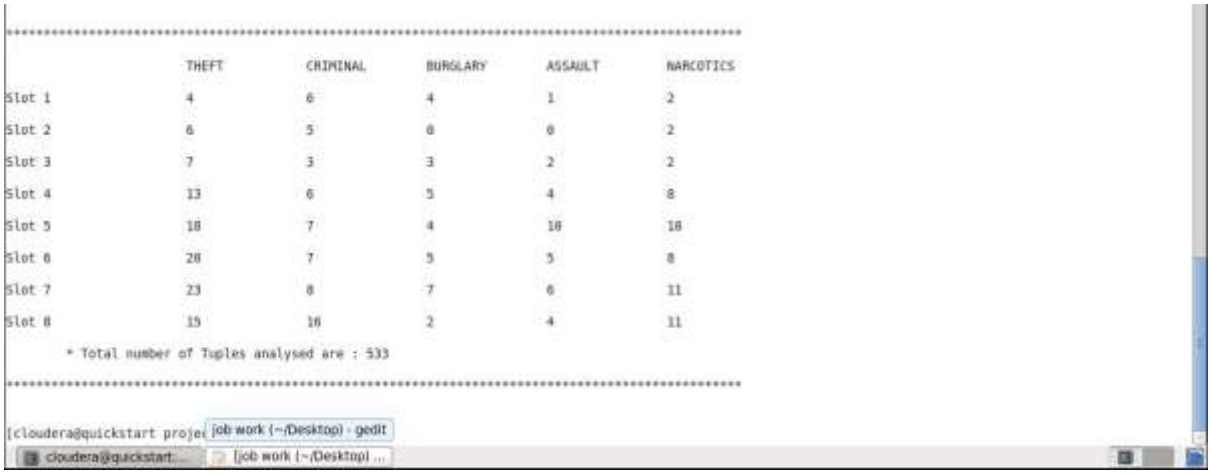
cd Desktop/project
cat crimedata_afterprep.txt | python time_map.py | python time_reduce.py




Applications Places System
Sun Apr 7, 1:07 PM cloudera
cloudera@quickstart:~/Desktop/project
File Edit View Search Terminal Help
For Time slot { 09:00 – 11:59 } Occurrence of Crime types are:
THEFT : 13
CRIMINAL DAMAGE : 6
BURGLARY : 5
ASSAULT : 4
NARCOTICS : 8
=> Total Crimes ccurred and reported in above time slot is : 36 reports.
=> Also the most ccurred crime in above time slot is : THEFT13 times.
For Time slot { 12:00 – 14:59 } Occurrence of Crime types are:
THEFT : 18
CRIMINAL DAMAGE : 7
BURGLARY : 4
ASSAULT : 10
NARCOTICS : 10
=> Total Crimes ccurred and reported in above time slot is : 49 reports.
=> Also the most ccurred crime in above time slot is : THEFT18 times.
For Time slot { 15:00 – 17:59 } Occurrence of Crime types are:
THEFT : 20
CRIMINAL DAMAGE : 7
BURGLARY : 5
ASSAULT : 5
NARCOTICS : 8
=> Total Crimes ccurred and reported in above time slot is : 45 reports.
=> Also the most ccurred crime in above time slot is : THEFT20 times.
cloudera@quickstart:...
[job work (~/Desktop) ...


Applications Places System
Sun Apr 7, 1:08 PM cloudera
cloudera@quickstart:~/Desktop/project
File Edit View Search Terminal Help
For Time slot { 18:00 – 20:59 } Occurrence of Crime types are:
THEFT : 23
CRIMINAL DAMAGE : 8
BURGLARY : 7
ASSAULT : 6
NARCOTICS : 11
=> Total Crimes ccurred and reported in above time slot is : 55 reports.
=> Also the most ccurred crime in above time slot is : THEFT23 times.
For Time slot { 21:00 – 23:59 } Occurrence of Crime types are:
THEFT : 15
CRIMINAL DAMAGE : 16
BURGLARY : 2
ASSAULT : 4
NARCOTICS : 11
=> Total Crimes ccurred and reported in above time slot is : 48 reports.
=> Also the most ccurred crime in above time slot is : CRIMINAL DAMAGE16 times.

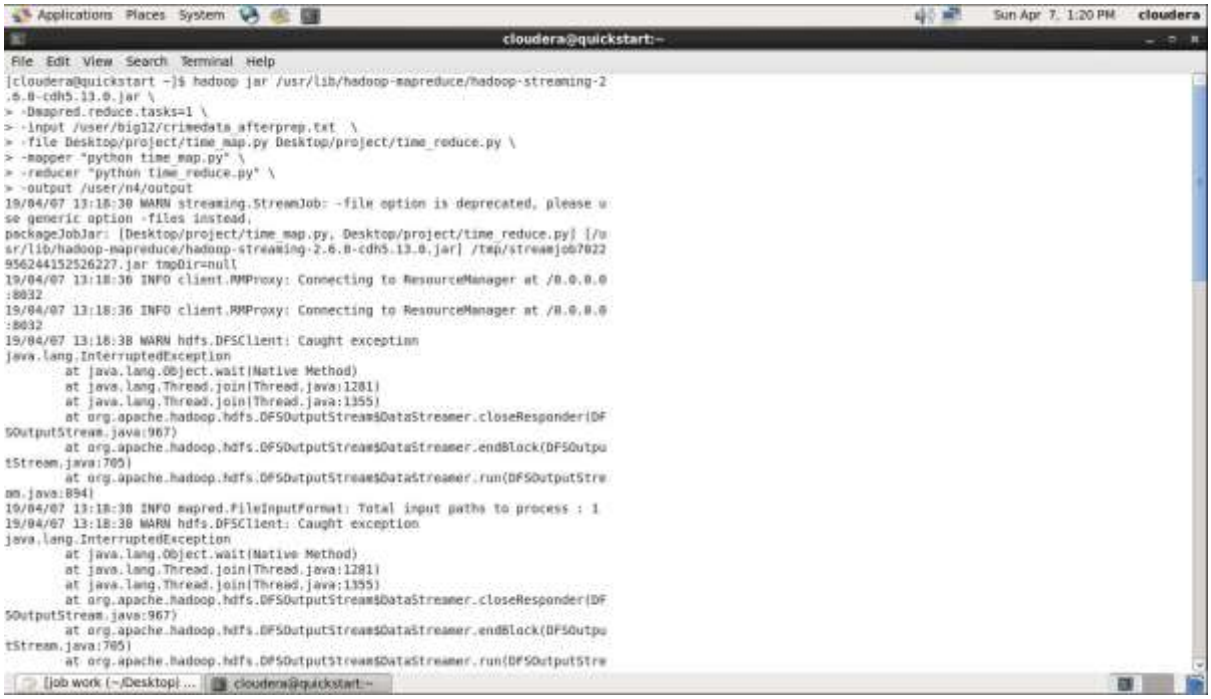
CONSLIDATED OUTPUT:

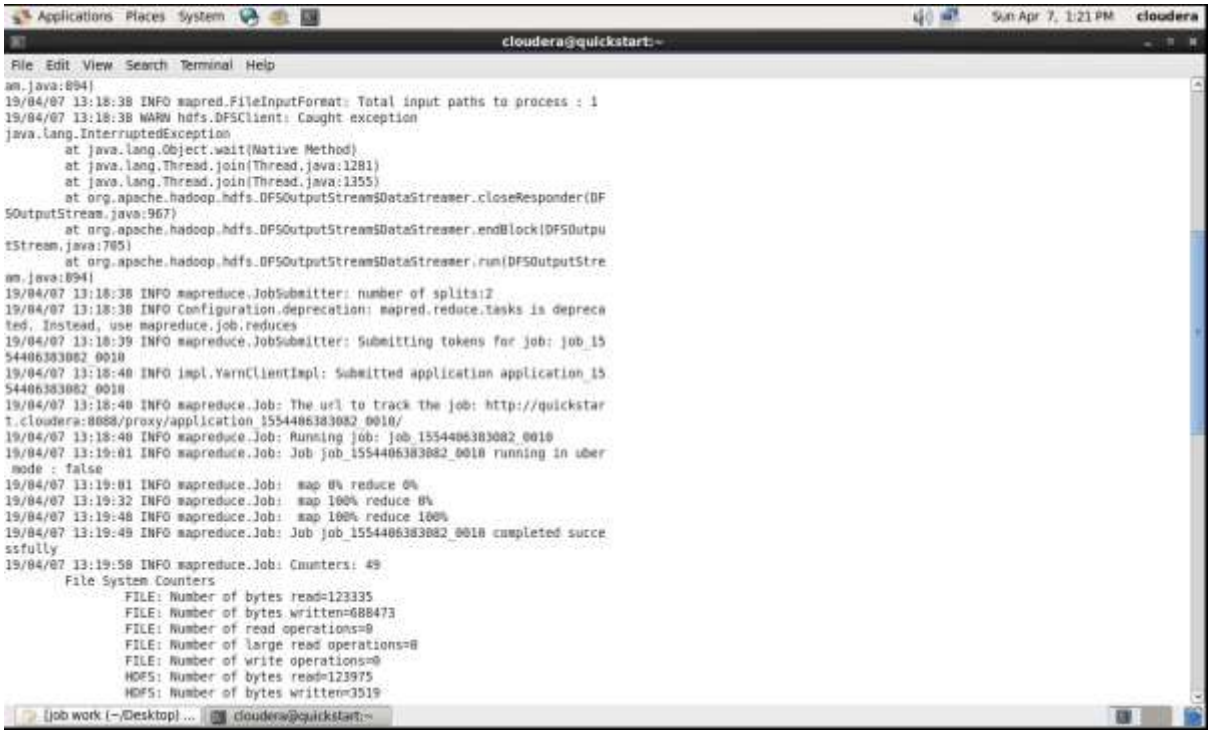


```
**********************************************************************************************

                    THEFT           CRIMINAL        BURGLARY        ASSAULT         NARCOTICS
Slot 1              4               6               4               1               2
Slot 2              6               5               0               0               2
Slot 3              7               3               3               2               2
Slot 4              13              6               5               4               8
Slot 5              18              7               4               10              18
Slot 6              28              7               5               5               8
Slot 7              23              8               7               6               11
Slot 8              15              16              2               4               11
       * Total number of Tuples analysed are : 533
**********************************************************************************************

[cloudera@quickstart projec job work (~/Desktop) - gedit
 cloudera@quickstart:...   [job work (~/Desktop) ...
```

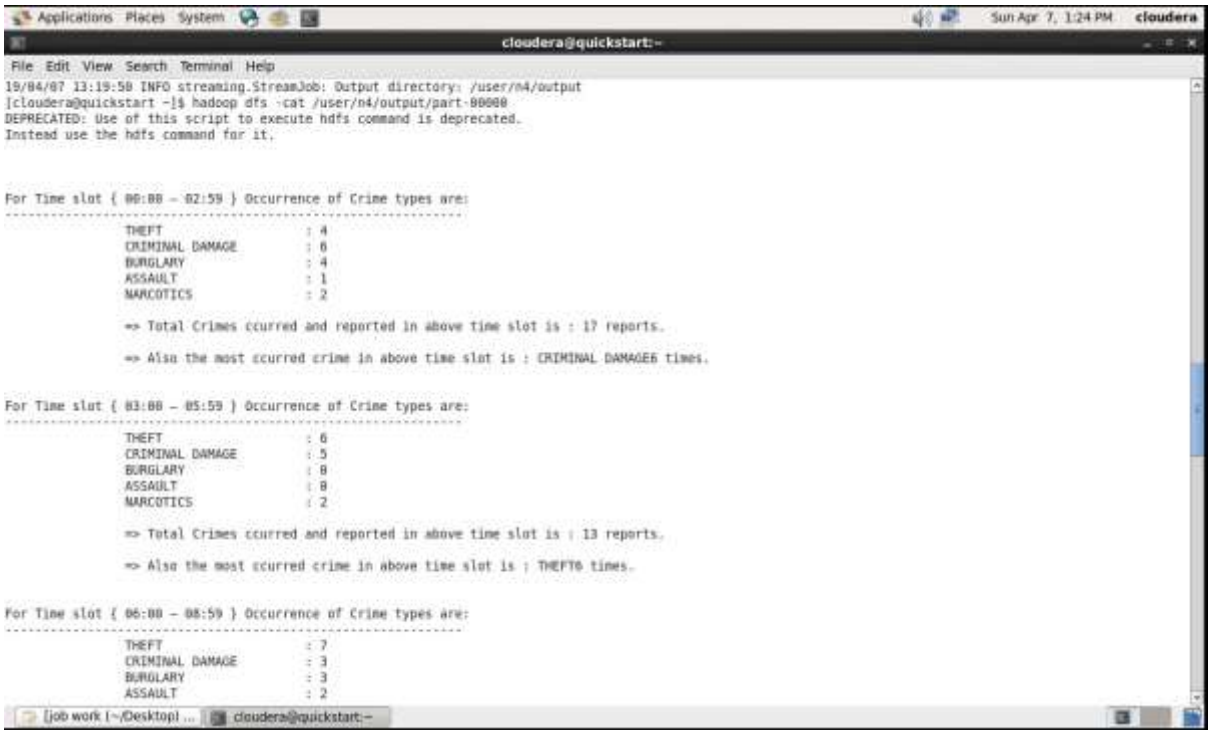
HADOOP EXECUTION:

*hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming-2.6.0-cdh5.13.0.jar *
*-Dmapred.reduce.tasks=1 *
*-input /user/big12/crimedata_afterprep.txt *
*-file Desktop/project/district_map.py Desktop/project/district_reduce.py *
*-mapper "python time_map.py" *
*-reducer "python time_reduce.py" *
-output /user/time_output/output

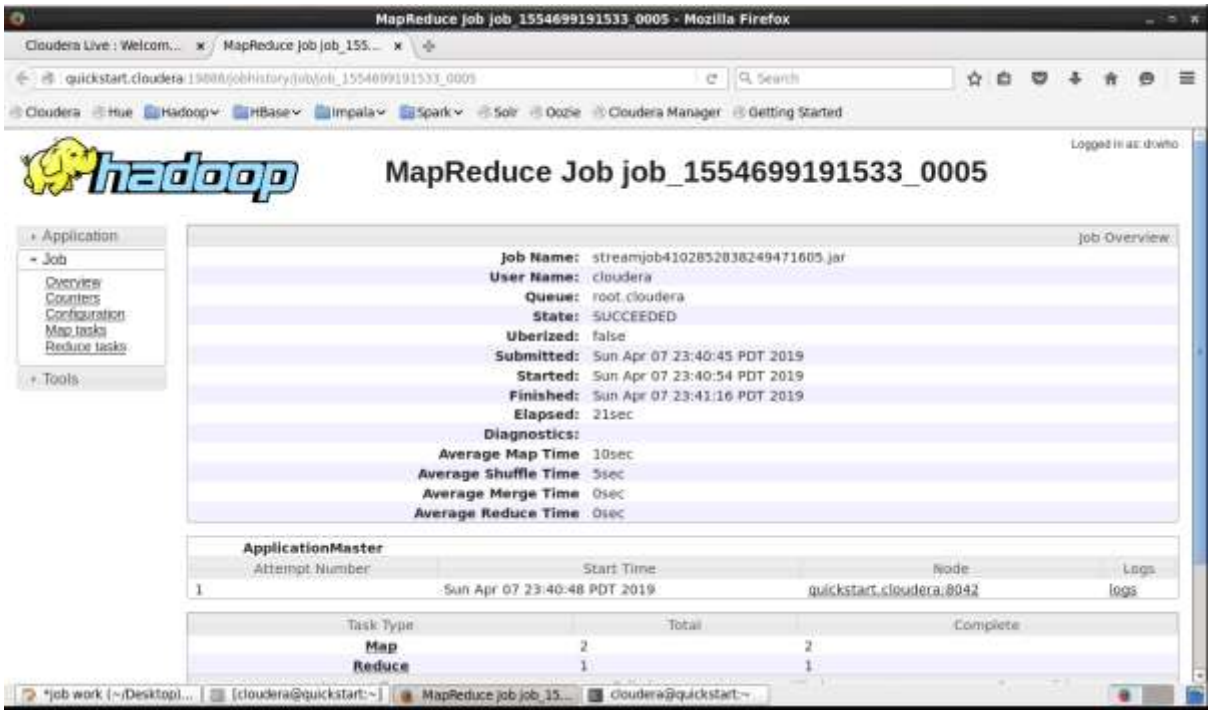


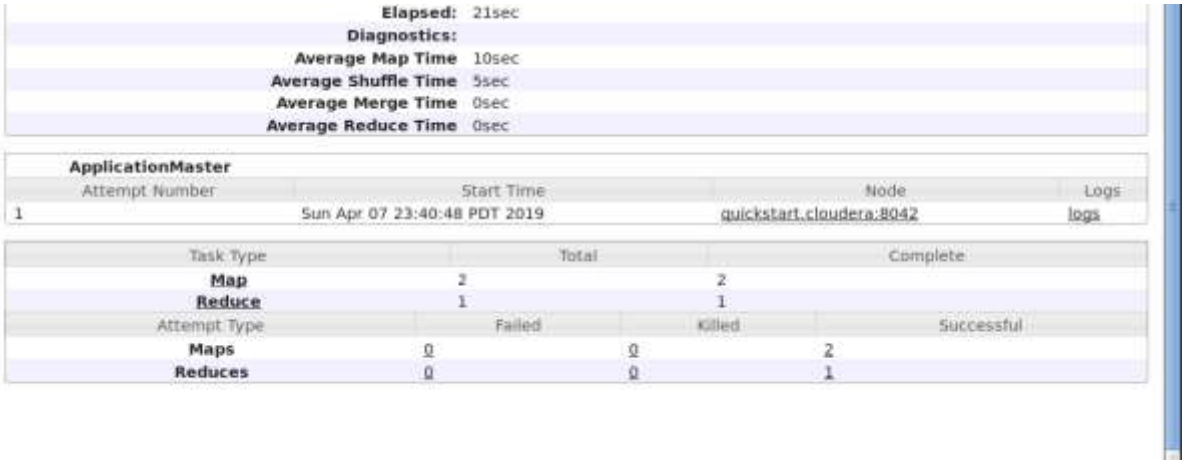


hdfs dfs –cat/user/n4/part-00000



HADOOP EXECUTION STATUS :





DISTRICT_MAP:

```
#!/usr/bin/python

import sys
flag=0

for input_line in sys.stdin:
        line = input_line.strip().split(",")
      if (flag ==0):
              columns=line
              flag=1
      else:
              disno = float(line[11])
              print "{0}\t{1}".format(disno,str(input_line.strip()))
```

DISTRICT_REDUCE:

```
#!/usr/bin/python
# coding: utf-8
import sys
tab = []
quad = []
count=0
for i in range(5):
        quad.append([])
        tab.append([])

for input_line in sys.stdin:
        line = input_line.strip().split("\t")
        disno = float(line[0])
        count=count+1
        if disno>=1 and disno<5:
                quad[0].append(line[1])
        elif disno>=5 and disno<10:
                quad[1].append(line[1])
        elif disno>=10 and disno<15:
                quad[2].append(line[1])
        elif disno>=15 and disno<20:
                quad[3].append(line[1])
```

```
        elif disno>=20 and disno<25:
                quad[4].append(line[1])

for i in range(0,5):
    theft_c = 0
    criminal_c=0
    burglary_c=0
    assault_c=0
    narcotics_c=0

    for j in range(len(quad[i])):
        crime_array = quad[i][j].strip().split(',')
        crime_type = str(crime_array[5])

        if crime_type == 'THEFT':
          theft_c = theft_c + 1
        elif crime_type=='CRIMINAL DAMAGE':
            criminal_c = criminal_c + 1
        elif crime_type=='BURGLARY':
            burglary_c = burglary_c + 1
        elif crime_type=='ASSAULT':
            assault_c = assault_c + 1
       elif crime_type=='NARCOTICS':
            narcotics_c = narcotics_c + 1
```

```
    total_crimes = theft_c + criminal_c + burglary_c + assault_c +
narcotics_c

    print '\n\nFor sector ' + str(i+1) + ' – Occurrence of Crime types are:
'
    print '-------------------------------------------------------'\
    print '\t\tTHEFT\t\t\t: '+str(theft_c)
    print '\t\tCRIMINAL DAMAGE\t\t: '+str(criminal_c)
    print '\t\tBURGLARY\t\t: '+str(burglary_c)
    print '\t\tASSAULT\t\t\t: '+str(assault_c)
    print '\t\tNARCOTICS\t\t: '+str(narcotics_c)
    most = max(int(theft_c),int(criminal_c), int(burglary_c),
int(assault_c), int(narcotics_c))

    if most==theft_c:
        most_crime='THEFT'
    elif most==criminal_c:
        most_crime='CRIMINAL DAMAGE'
    elif most==burglary_c:
        most_crime='BURGLARY'
    elif most==assault_c:
        most_crime='ASSAULT'
    elif most==narcotics_c:
```

```
        most_crime='NARCOTICS'

    print '\n\t\t=> Total Crimes ccurred and reported in above sector is : ' + str(total_crimes) + ' reports.'
    print '\n\t\t=> Also the most ccurred crime in above sector is : ' + most_crime  + str(most) + ' times.'

    tab[i].append(theft_c)
    tab[i].append(criminal_c)
    tab[i].append(burglary_c)
    tab[i].append(assault_c)
    tab[i].append(narcotics_c)
print '\n\n**************************************************
**************************************************'
print '\n\t\t\tTHEFT\t\tCRIMINAL\tBURGLARY\tASSAULT\t\tNARCOTICS'

for i in range(0,5):
        print "\nSector "+str(i+1)+"\t\t\t"+str(tab[i][0])+"\t\t"+str(tab[i][1])+"\t\t"+str(tab[i][2])+"\t\t"+str(tab[i][3])+"\t\t"+str(tab[i][4])
print '\n\t* Total number of Tuples analysed are : ' + str(count)
```

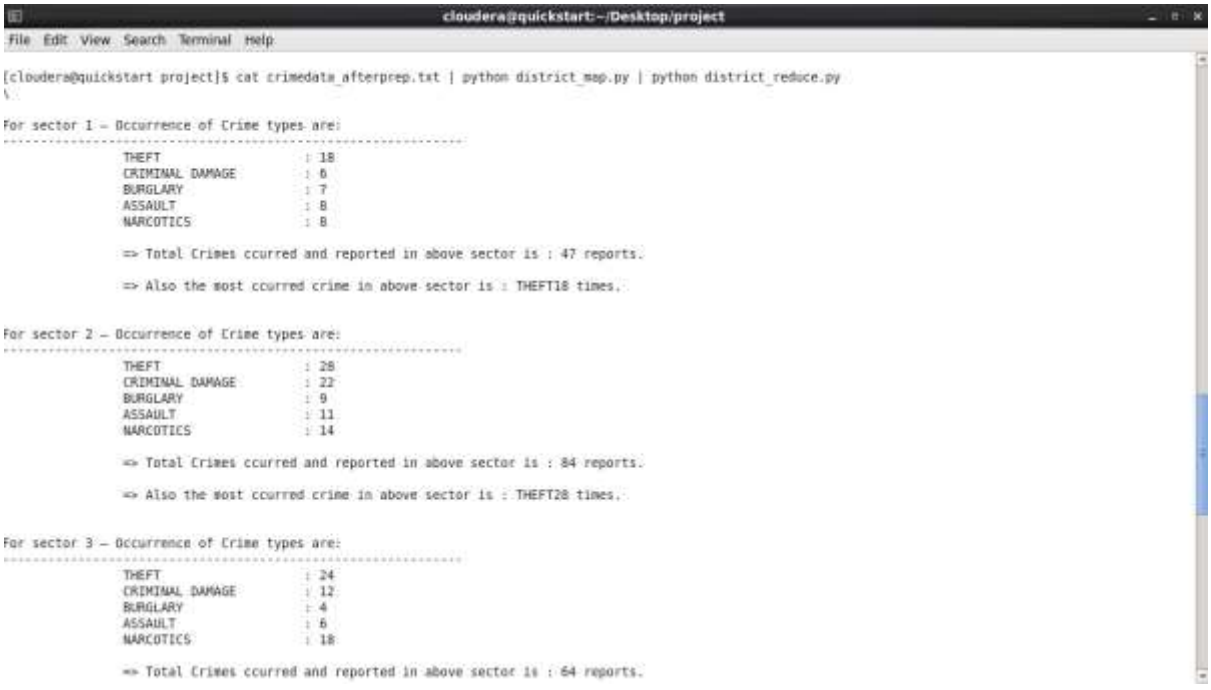
print

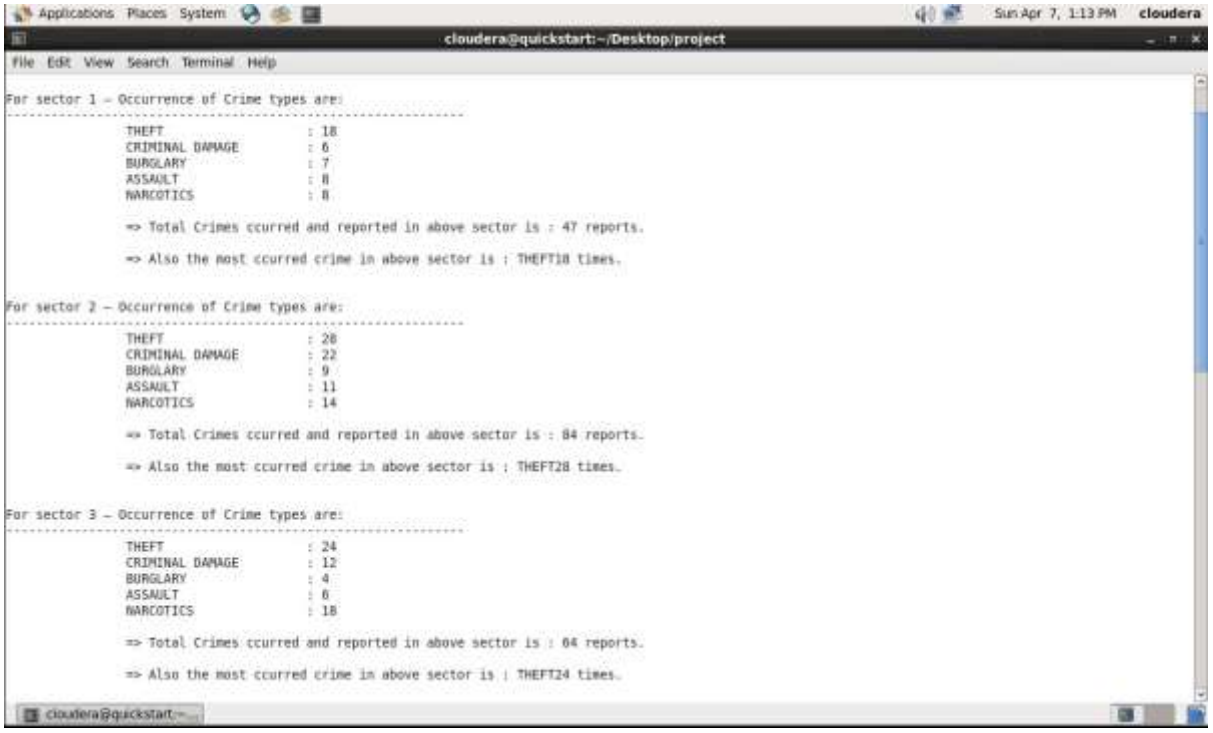
*'\n**********\n\n'*

LOCAL EXECUTION:

COMMANDS USED:

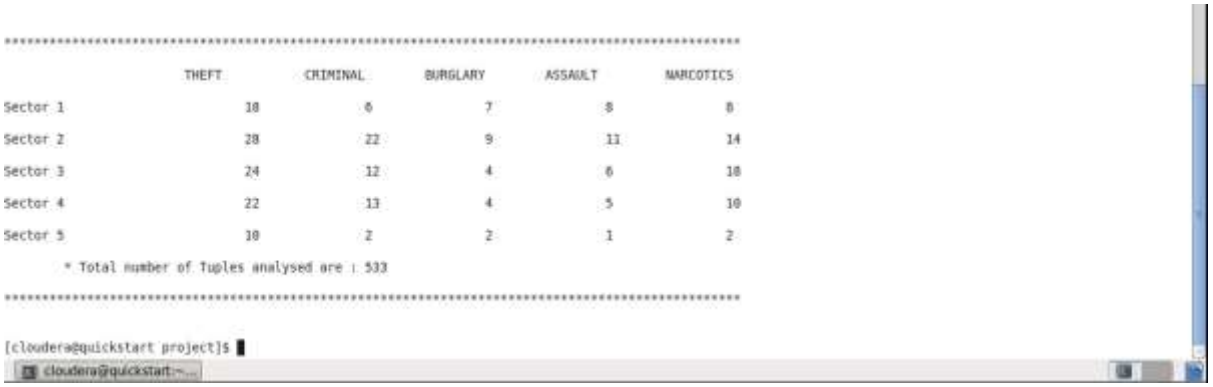
cd Desktop/project
cat crimedata_afterprep.txt | python district_map.py | python district_reduce.py




Applications Places System
Sun Apr 7, 1:13 PM cloudera
cloudera@quickstart:~/Desktop/project
File Edit View Search Terminal Help
For sector 1 – Occurrence of Crime types are:
THEFT : 18
CRIMINAL DAMAGE : 6
BURGLARY : 7
ASSAULT : 8
NARCOTICS : 8
=> Total Crimes ccurred and reported in above sector is : 47 reports.
=> Also the most ccurred crime in above sector is : THEFT18 times.
For sector 2 – Occurrence of Crime types are:
THEFT : 28
CRIMINAL DAMAGE : 22
BURGLARY : 9
ASSAULT : 11
NARCOTICS : 14
=> Total Crimes ccurred and reported in above sector is : 84 reports.
=> Also the most ccurred crime in above sector is : THEFT28 times.
For sector 3 – Occurrence of Crime types are:
THEFT : 24
CRIMINAL DAMAGE : 12
BURGLARY : 4
ASSAULT : 6
NARCOTICS : 18
=> Total Crimes ccurred and reported in above sector is : 64 reports.
=> Also the most ccurred crime in above sector is : THEFT24 times.
cloudera@quickstart:~


Applications Places System
Sun Apr 7, 1:14 PM cloudera
cloudera@quickstart:~/Desktop/project
File Edit View Search Terminal Help
For sector 4 – Occurrence of Crime types are:
THEFT : 22
CRIMINAL DAMAGE : 13
BURGLARY : 4
ASSAULT : 5
NARCOTICS : 10
=> Total Crimes ccurred and reported in above sector is : 54 reports.
=> Also the most ccurred crime in above sector is : THEFT22 times.
For sector 5 – Occurrence of Crime types are:
THEFT : 10
CRIMINAL DAMAGE : 2
BURGLARY : 2
ASSAULT : 1
NARCOTICS : 2
=> Total Crimes ccurred and reported in above sector is : 17 reports.
=> Also the most ccurred crime in above sector is : THEFT10 times.

CONSOLIDATED OUTPUT:



HADOOP EXECUTION:

*hadoop jar /usr/lib/hadoop-mapreduce/hadoop-streaming-2.6.0-cdh5.13.0.jar *

*-Dmapred.reduce.tasks=1 *

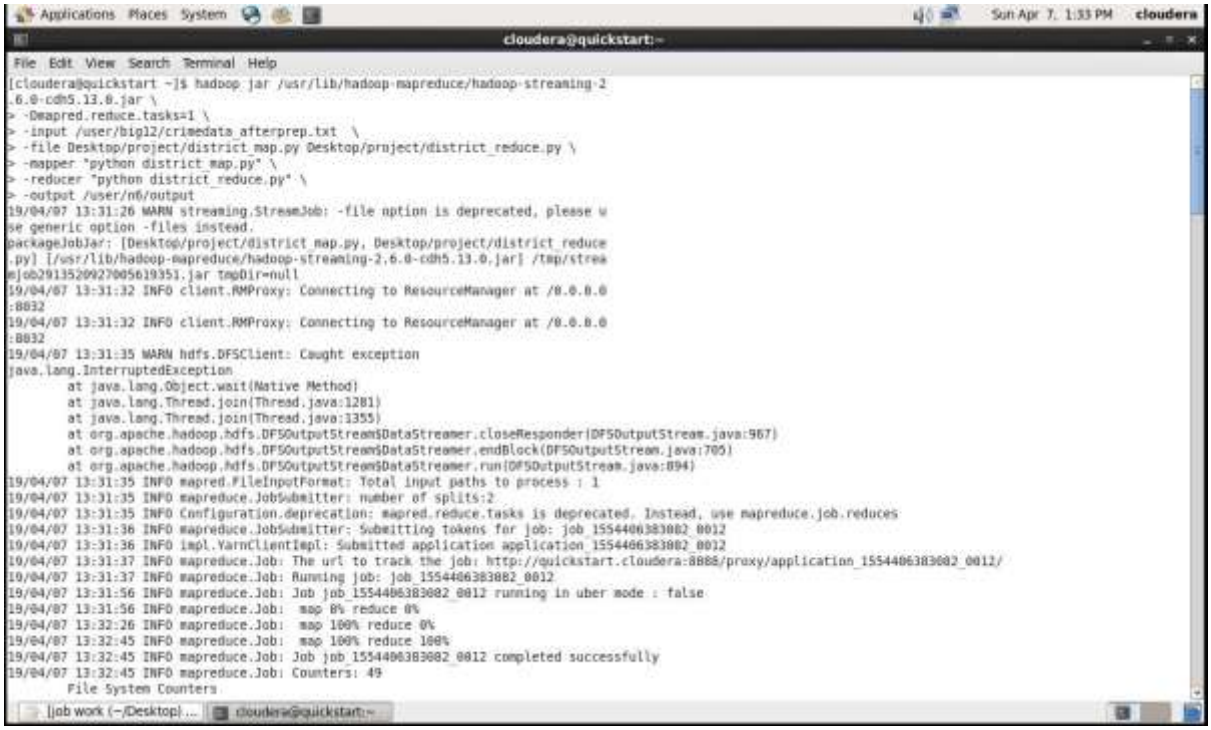
*-input /user/big12/crimedata_afterprep.txt *

*-file Desktop/project/district_map.py Desktop/project/district_reduce.py *

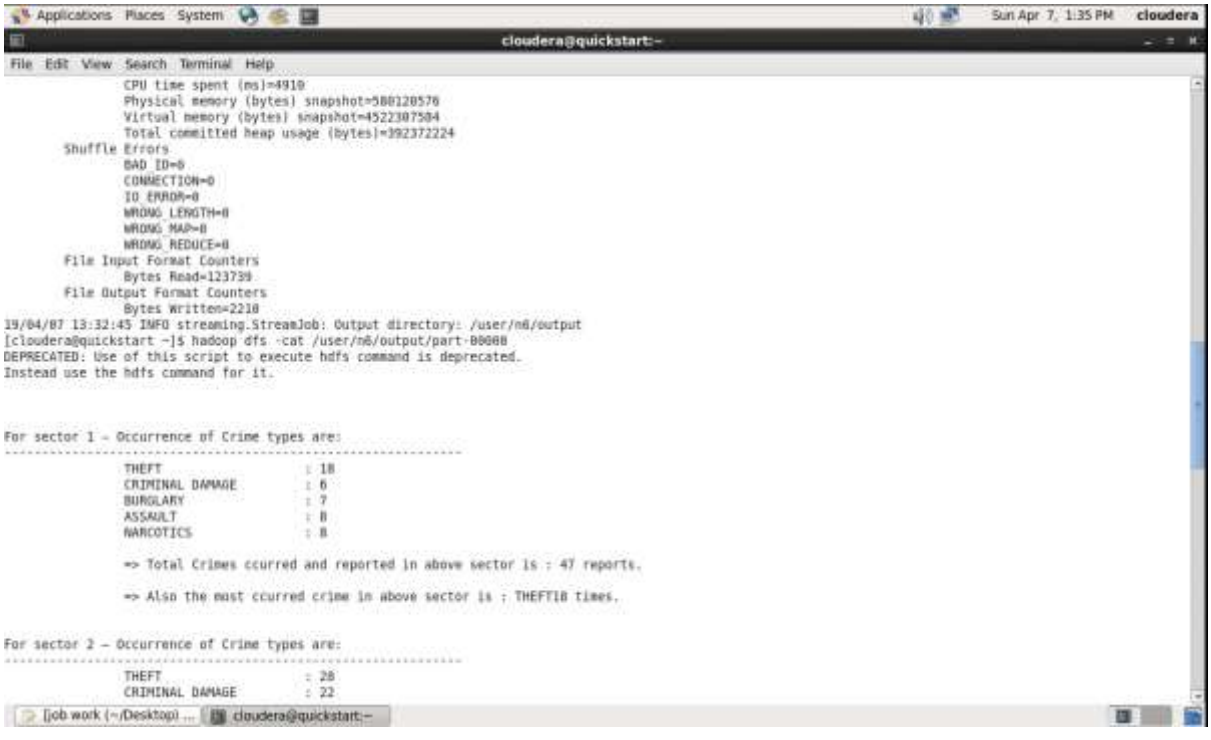
*-mapper "python district_map.py" *

*-reducer "python district_reduce.py" *

-output /user/n6/output




cloudera@quickstart:~
File Edit View Search Terminal Help
19/04/07 22:27:03 INFO mapreduce.Job: map 100% reduce 0%
19/04/07 22:27:12 INFO mapreduce.Job: map 100% reduce 100%
19/04/07 22:27:12 INFO mapreduce.Job: Job job_1554690191533_0003 completed successfully
19/04/07 22:27:12 INFO mapreduce.Job: Counters: 50
File System Counters
FILE: Number of bytes read=123217
FILE: Number of bytes written=688357
FILE: Number of read operations=0
FILE: Number of large read operations=0
FILE: Number of write operations=0
HDFS: Number of bytes read=123975
HDFS: Number of bytes written=2210
HDFS: Number of read operations=9
HDFS: Number of large read operations=0
HDFS: Number of write operations=2
Job Counters
Killed map tasks=1
Launched map tasks=2
Launched reduce tasks=1
Data-local map tasks=2
Total time spent by all maps in occupied slots (ms)=22762
Total time spent by all reduces in occupied slots (ms)=6962
Total time spent by all map tasks (ms)=22762
Total time spent by all reduce tasks (ms)=6962
Total vcore-milliseconds taken by all map tasks=22762
Total vcore-milliseconds taken by all reduce tasks=6962
Total megabyte-milliseconds taken by all map tasks=23308288
Total megabyte-milliseconds taken by all reduce tasks=7129088
Map-Reduce Framework
Map input records=534
Map output records=532
Map output bytes=121606
Map output materialized bytes=123223
Input split bytes=236
Combine input records=0
Combine output records=0
Reduce input groups=22
Reduce shuffle bytes=123223
Reduce input records=532


Applications Places System
Sun Apr 7, 1:35 PM cloudera
cloudera@quickstart:~
File Edit View Search Terminal Help
CPU time spent (ms)=4910
Physical memory (bytes) snapshot=580120576
Virtual memory (bytes) snapshot=4522307584
Total committed heap usage (bytes)=392372224
Shuffle Errors
BAD_ID=0
CONNECTION=0
IO_ERROR=0
WRONG_LENGTH=0
WRONG_MAP=0
WRONG_REDUCE=0
File Input Format Counters
Bytes Read=123739
File Output Format Counters
Bytes Written=2210
19/04/07 13:32:45 INFO streaming.StreamJob: Output directory: /user/n6/output
[cloudera@quickstart ~]$ hadoop dfs -cat /user/n6/output/part-00000
DEPRECATED: Use of this script to execute hdfs command is deprecated.
Instead use the hdfs command for it.
For sector 1 - Occurrence of Crime types are:
THEFT : 18
CRIMINAL DAMAGE : 6
BURGLARY : 7
ASSAULT : 8
NARCOTICS : 8
=> Total Crimes ccurred and reported in above sector is : 47 reports.
=> Also the most ccurred crime in above sector is : THEFT18 times.
For sector 2 - Occurrence of Crime types are:
THEFT : 28
CRIMINAL DAMAGE : 22
[job work (~/Desktop) ...
cloudera@quickstart:~

LOCATION_MAP:

```
#!/usr/bin/python

import sys
cnt=0
for input_line in sys.stdin:
        line = input_line.strip().split(",")
        if (flag ==0):
                columns=line
                flag=1
        else:
                x = float(line[15])
                y = float(line[16])
                print "{0}\t{1}\t{2}".format(x,y,str(input_line.strip()))
```

LOCATION_REDUCE:

```
#!/usr/bin/python

import sys
tab = []
quad = []
count=0
for i in range(5):
        quad.append([])
        tab.append([])
for input_line in sys.stdin:
        line = input_line.strip().split("\t")        # X-coordinate \t Y-coordinate \t Tuple
        x = float(line[0]) # x-coordinate
        y = float(line[1]) # y-coordinate
        count=count+1

        if x>=1095000 and x<1117000 and y>=1836000 and y<1856600:
                quad[0].append(line[2])
        elif x>=1117000 and x<1139000 and y>=1856600 and y<1877200:
                quad[1].append(line[2])
        elif x>=1139000 and x<1161000 and y>=1877200 and y<1897800:
```

```
            quad[2].append(line[2])
        elif x>=1161000 and x<1183000 and y>=1897800 and
y<1918400:
            quad[3].append(line[2])
        elif x>=1183000 and x<1205000 y>=1918400 and y<1939000:
            quad[4].append(line[2])
for i in range(0,5):

        theft_c = 0
        criminal_c=0
        burglary_c=0
        assault_c=0
        narcotics_c=0
        for j in range(len(quad[i])):
            crime_array = quad[i][j].strip().split(',')
            crime_type = str(crime_array[5])
            if crime_type == 'THEFT':
                theft_c = theft_c + 1
            elif crime_type=='CRIMINAL DAMAGE':
                criminal_c = criminal_c + 1
            elif crime_type=='BURGLARY':
                burglary_c = burglary_c + 1
            elif crime_type=='ASSAULT':
                assault_c = assault_c + 1
```

```
            elif crime_type=='NARCOTICS':
                narcotics_c = narcotics_c + 1


        total_crimes = theft_c + criminal_c + burglary_c + assault_c +
narcotics_c
        print '\n\nFor Area ' + str(i+1) + ' – Occurrence of Crime types
are:  '
        print '------------------------------------------------------'
        print '\t\tTHEFT\t\t\t: '+str(theft_c)
        print '\t\tCRIMINAL DAMAGE\t\t: '+str(criminal_c)
        print '\t\tBURGLARY\t\t: '+str(burglary_c)
        print '\t\tASSAULT\t\t\t: '+str(assault_c)
        print '\t\tNARCOTICS\t\t: '+str(narcotics_c)
        most    =    max(int(theft_c),int(criminal_c),    int(burglary_c),
int(assault_c), int(narcotics_c))
        if most==theft_c:
            most_crime='THEFT'
        elif most==criminal_c:
            most_crime='CRIMINAL DAMAGE'
        elif most==burglary_c:
            most_crime='BURGLARY'
        elif most==assault_c:
            most_crime='ASSAULT'
```

```
        elif most==narcotics_c:
                most_crime='NARCOTICS'
        print '\n\t\t=> Total Crimes ccurred and reported in above Area is : ' + str(total_crimes) + ' reports.'
        Print '\n\t\t=> Also the most ccurred crime in above Area is : ' + most_crime + ' – ' +str(most) + ' times.'
        tab[i].append(theft_c)
        tab[i].append(criminal_c)
        tab[i].append(burglary_c)
        tab[i].append(assault_c)
        tab[i].append(narcotics_c)
print '\n\n**************************************************************************************************'
print '\n\t\t\tTHEFT\t\tCRIMINAL\tBURGLARY\tASSAULT\t\tNARCOTICS'
for i in range(0,5):
        print "\nArea "+str(i+1)+"\t\t\t"+str(tab[i][0])+"\t\t"+str(tab[i][1])+"\t\t"+str(tab[i][2])+"\t\t"+str(tab[i][3])+"\t\t"+str(tab[i][4])
print '\n\n**************************************************************************************************'
print '\n\t* Total number of Tuples analysed are : ' + str(count)
```

print

*'\n**********\n\n'*

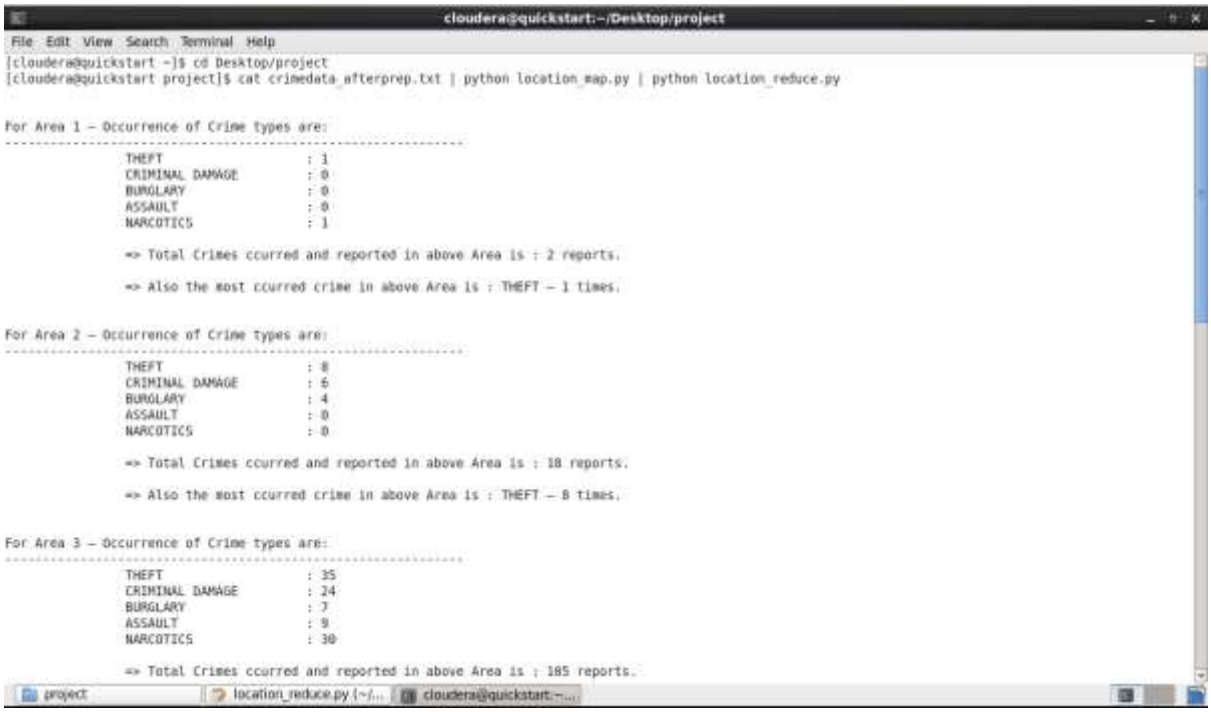
#print "{0}\t{1}".format(output, 1)

#print "{0}\t{1}".format(output, 1)

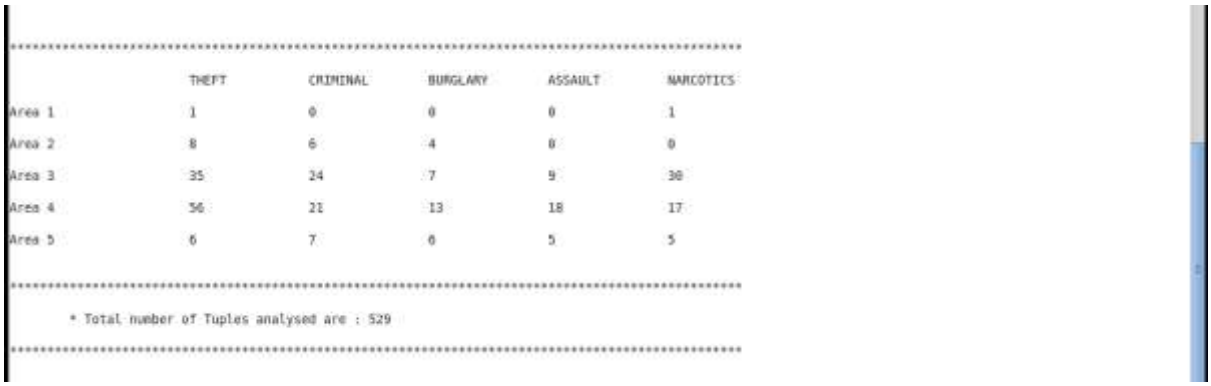
LOCAL EXECUTION:

COMMANDS USED:

cd Desktop/project
cat crimedata_afterprep.txt | python location_map.py | python location_reduce.py



Consolidated output:



	THEFT	CRIMINAL	BURGLARY	ASSAULT	NARCOTICS
Area 1	1	0	0	0	1
Area 2	8	6	4	0	0
Area 3	35	24	7	9	30
Area 4	56	21	13	18	17
Area 5	6	7	6	5	5

* Total number of Tuples analysed are : 529