Baltimore crime data analysis and modeling

(https://query.data.world/s/axat2ortbtqqehhtmfwuaz4hffkujp (https://query.data.world/s/axat2ortbtqqehhtmfwuaz4hffkujp)).

Import package

```
from csv import reader
from pyspark.sql import Row
from pyspark.sql import SparkSession
from pyspark.sql.types import *
import pandas as pd
import numpy as np
import seaborn as sb
import matplotlib.pyplot as plt
import warnings

import os
os.environ["PYSPARK_PYTHON"] = "python3"
```

```
# download dataset
import urllib.request
urllib.request.urlretrieve("https://query.data.world/s/uwn5462sauinmmmg3kkmalm2expvnx",
"/tmp/my123.csv")
dbutils.fs.mv("file:/tmp/my123.csv", "dbfs:/chris/spark_hw1/data/Baltimore_03_18.csv")
display(dbutils.fs.ls("dbfs:/chris/spark_hw1/data/"))
```

path	name
dbfs:/chris/spark_hw1/data/Baltimore_03_18.csv	Baltimore_03_
dbfs:/chris/spark_hw1/data/sf_03_18.csv	sf_03_18.csv

```
data_path = "dbfs:/chris/spark_hw1/data/Baltimore_03_18.csv"
# use this file name later
```

Get dataframe and sql

```
from pyspark.sql import SparkSession
spark = SparkSession \
    .builder \
    .appName("crime analysis") \
    .config("spark.some.config.option", "some-value") \
    .getOrCreate()

df_opt1 = spark.read.format("csv").option("header", "true").load(data_path)
display(df_opt1)
df_opt1.createOrReplaceTempView("Baltimore_crime")



# from pyspark.sql.functions import to_date, to_timestamp, hour
# df_opt1 = df_opt1.withColumn('Date', to_date(df_opt1.OccurredOn, "MM/dd/yy"))
# df_opt1 = df_opt1.withColumn('Time', to_timestamp(df_opt1.OccurredOn, "MM/dd/yy HH:mm"))
# df_opt1 = df_opt1.withColumn('Hour', hour(df_opt1['Time']))
# df_opt1 = df_opt1.withColumn("DayOfWeek", date_format(df_opt1.Date, "EEEE"))

#from pyspark.sql.functions import col, udf
#from pyspark.sql.functions import expr
#from pyspark.sql.functions import from_unixtime

#date_func =  udf (lambda x: datetime.strptime(x, '%m/%d/%Y'), DateType())
#month_func = udf (lambda x: datetime.strptime(x, '%m/%d/%Y').strftime('%Y/%m'), StringType())

#df = df_opt1.withColumn('month_year', month_func(col('Date')))\
#            .withColumn('Date_time', date_func(col('Date')))
# select Date, substring(Date,7) as Year, substring(Date,1,2) as Month from sf_crime

from pyspark.sql.functions import *
df_update = df_opt1.withColumn("CrimeDate", to_date(col("CrimeDate"), "MM/dd/yyyy")) ##change
datetype from string to date
df_update.createOrReplaceTempView("Baltimore_crime")
crimeYearMonth = spark.sql("SELECT Year(Date) AS Year, Month(Date) AS Month, FROM
Baltimore_crime")
```

CrimeDate	CrimeTime	CrimeCode	Location	Description	Inside/Outside	Weapon	Post
11/12/2016	02:35:00	3B	300 SAINT PAUL PL	ROBBERY - STREET	O	null	111
11/12/2016	02:56:00	3CF	800 S BROADWAY	ROBBERY - COMMERCIAL	I	FIREARM	213
11/12/2016	03:00:00	6D	1500 PENTWOOD RD	LARCENY FROM AUTO	O	null	413
11/12/2016	03:00:00	6D	6600 MILTON LN	LARCENY FROM AUTO	O	null	424

Showing the first 1000 rows.

1. Data Cleaning and Exploration

```
# transfer from spark sql into pandas Dataframe
Baltimore_crime= df_opt1.toPandas()
Baltimore_crime.head(10)
```

Out[7]:

	CrimeDate	CrimeTime	CrimeCode	Location	Description	Inside/Outside	Weapon	Post	District	Neig
0	11/12/2016	02:35:00	3B	300 SAINT PAUL PL	ROBBERY - STREET	O	None	111	CENTRAL	
1	11/12/2016	02:56:00	3CF	800 S BROADWAY	ROBBERY - COMMERCIAL	I	FIREARM	213	SOUTHEASTERN	
2	11/12/2016	03:00:00	6D	1500 PENTWOOD RD	LARCENY FROM AUTO	O	None	413	NORTHEASTERN	S
3	11/12/2016	03:00:00	6D	6600 MILTON LN	LARCENY FROM AUTO	O	None	424	NORTHEASTERN	
4	11/12/2016	03:00:00	6E	300 W BALTIMORE ST	LARCENY	O	None	111	CENTRAL	
5	11/12/2016	03:00:00	4E	6900 MCCLEAN BLVD	COMMON ASSAULT	I	HANDS	423	NORTHEASTERN	Ha
6	11/12/2016	03:45:00	3CO	1700 W LOMBARD ST	ROBBERY - COMMERCIAL	O	OTHER	933	SOUTHERN	Uni
7	11/12/2016	04:27:00	6D	0 N CONKLING ST	LARCENY FROM AUTO	O	None	223	SOUTHEASTERN	
				5200	ROBBERY -					

```
# Data information
Baltimore_crime.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 285807 entries, 0 to 285806
Data columns (total 12 columns):
CrimeDate          285807 non-null object
CrimeTime          285807 non-null object
CrimeCode          285807 non-null object
Location           284184 non-null object
Description        285807 non-null object
Inside/Outside     281611 non-null object
Weapon             97396 non-null object
Post               285616 non-null object
District           285749 non-null object
```

```
Neighborhood       284106 non-null object
Location 1         284188 non-null object
Total Incidents    285807 non-null object
dtypes: object(12)
memory usage: 26.2+ MB
```

```
# check data dimension information
print ("Num of rows: " + str(Baltimore_crime.shape[0]))
print ("Num of columns: " + str(Baltimore_crime.shape[1]))
```

```
Num of rows: 285807
Num of columns: 12
```

```
# check all the missing value
Baltimore_crime.isnull().sum()
```

```
Out[10]: CrimeDate               0
CrimeTime               0
CrimeCode               0
Location             1623
Description             0
Inside/Outside       4196
Weapon             188411
Post                  191
District               58
Neighborhood         1701
Location 1           1619
Total Incidents         0
dtype: int64
```

```
Baltimore_crime.nunique()
```

```
Out[11]: CrimeDate           2143
CrimeTime           4236
CrimeCode             81
Location           25949
Description           15
Inside/Outside         4
Weapon                 4
Post                 189
District              13
Neighborhood         280
Location 1         97951
Total Incidents        1
dtype: int64
```

```
# becasue in the Weapon column, the missing value more than 60% of the sample numbers, then drop it.
drop_columns = ['Weapon']
X = Baltimore_crime.drop(drop_columns, axis=1)
X.head(10)
```

```
Out[12]:
```

	CrimeDate	CrimeTime	CrimeCode	Location	Description	Inside/Outside	Post	District	Neighborhood
0	11/12/2016	02:35:00	3B	300 SAINT PAUL PL	ROBBERY - STREET	O	111	CENTRAL	Downtown
1	11/12/2016	02:56:00	3CF	800 S BROADWAY	ROBBERY - COMMERCIAL	I	213	SOUTHEASTERN	Fells Point
2	11/12/2016	03:00:00	6D	1500 PENTWOOD RD	LARCENY FROM AUTO	O	413	NORTHEASTERN	Stonewood-Pentwood-Winston
3	11/12/2016	03:00:00	6D	6600 MILTON LN	LARCENY FROM AUTO	O	424	NORTHEASTERN	Westfield
4	11/12/2016	03:00:00	6E	300 W BALTIMORE ST	LARCENY	O	111	CENTRAL	Downtown
5	11/12/2016	03:00:00	4E	6900 MCCLEAN BLVD	COMMON ASSAULT	I	423	NORTHEASTERN	Hamilton Hills
6	11/12/2016	03:45:00	3CO	1700 W LOMBARD ST	ROBBERY - COMMERCIAL	O	933	SOUTHERN	Union Square
7	11/12/2016	04:27:00	6D	0 N CONKLING ST	LARCENY FROM AUTO	O	223	SOUTHEASTERN	Baltimore Highlands
				5200	ROBBERY -				

```
# drop missing value
X_new = X.dropna()
X_new.head(10)
```

Out[13]:

	CrimeDate	CrimeTime	CrimeCode	Location	Description	Inside/Outside	Post	District	Neighborhood
0	11/12/2016	02:35:00	3B	300 SAINT PAUL PL	ROBBERY - STREET	O	111	CENTRAL	Downtown
1	11/12/2016	02:56:00	3CF	800 S BROADWAY	ROBBERY - COMMERCIAL	I	213	SOUTHEASTERN	Fells Point
2	11/12/2016	03:00:00	6D	1500 PENTWOOD RD	LARCENY FROM AUTO	O	413	NORTHEASTERN	Stonewood-Pentwood-Winston
3	11/12/2016	03:00:00	6D	6600 MILTON LN	LARCENY FROM AUTO	O	424	NORTHEASTERN	Westfield
4	11/12/2016	03:00:00	6E	300 W BALTIMORE ST	LARCENY	O	111	CENTRAL	Downtown
5	11/12/2016	03:00:00	4E	6900 MCCLEAN BLVD	COMMON ASSAULT	I	423	NORTHEASTERN	Hamilton Hills
6	11/12/2016	03:45:00	3CO	1700 W LOMBARD ST	ROBBERY - COMMERCIAL	O	933	SOUTHERN	Union Square
7	11/12/2016	04:27:00	6D	0 N CONKLING ST	LARCENY FROM AUTO	O	223	SOUTHEASTERN	Baltimore Highlands
				5200	ROBBERY -				

```
# data shape after drop missing value
print ("Num of rows after drop missing value: " + str(X_new.shape[0]))
print ("Num of columns after drop missing value: " + str(X_new.shape[1]))
```

```
Num of rows after drop missing value: 279937
Num of columns after drop missing value: 11
```

```
# from the opration below, we find there are some uncommon data type in the CrimeTime column
column = X_new["CrimeTime"]
new_column = column.to_list()
for i in range(len(new_column)):
  if len(new_column[i]) != 8:
    new_column[i] = None
Crime_Time = pd.Series(new_column)
X_new['Crime_Time'] = Crime_Time
X_new.head()
```

```
/local_disk0/tmp/1583776735075-0/PythonShell.py:8: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#in
dexing-view-versus-copy
  import signal
Out[15]:
```

	CrimeDate	CrimeTime	CrimeCode	Location	Description	Inside/Outside	Post	District	Neighborhood
0	11/12/2016	02:35:00	3B	300 SAINT PAUL PL	ROBBERY - STREET	O	111	CENTRAL	Downtown
1	11/12/2016	02:56:00	3CF	800 S BROADWAY	ROBBERY - COMMERCIAL	I	213	SOUTHEASTERN	Fells Point
2	11/12/2016	03:00:00	6D	1500 PENTWOOD RD	LARCENY FROM AUTO	O	413	NORTHEASTERN	Stonewood-Pentwood-Winston
3	11/12/2016	03:00:00	6D	6600 MILTON LN	LARCENY FROM AUTO	O	424	NORTHEASTERN	Westfield
4	11/12/2016	03:00:00	6E	300 W BALTIMORE ST	LARCENY	O	111	CENTRAL	Downtown

```
# becasue in the previous CrimeTime column, there are many uncommon data type, so we need to drop it.
drop_columns = ['CrimeTime']
X = X_new.drop(drop_columns, axis=1)
X.head(10)
```

Out[16]:

	CrimeDate	CrimeCode	Location	Description	Inside/Outside	Post	District	Neighborhood	Locatio
0	11/12/2016	3B	300 SAINT PAUL PL	ROBBERY - STREET	O	111	CENTRAL	Downtown	(39.29241000 -76.61408000
1	11/12/2016	3CF	800 S BROADWAY	ROBBERY - COMMERCIAL	I	213	SOUTHEASTERN	Fells Point	(39.28242000 -76.59288000
2	11/12/2016	6D	1500 PENTWOOD RD	LARCENY FROM AUTO	O	413	NORTHEASTERN	Stonewood-Pentwood-Winston	(39.34805000 -76.58834000
3	11/12/2016	6D	6600 MILTON LN	LARCENY FROM AUTO	O	424	NORTHEASTERN	Westfield	(39.36263000 -76.55161000
4	11/12/2016	6E	300 W BALTIMORE ST	LARCENY	O	111	CENTRAL	Downtown	(39.28938000 -76.61971000
5	11/12/2016	4E	6900 MCCLEAN BLVD	COMMON ASSAULT	I	423	NORTHEASTERN	Hamilton Hills	(39.37070000 -76.56709000
6	11/12/2016	3CO	1700 W LOMBARD ST	ROBBERY - COMMERCIAL	O	933	SOUTHERN	Union Square	(39.28624000 -76.64455000
7	11/12/2016	6D	0 N CONKLING ST	LARCENY FROM AUTO	O	223	SOUTHEASTERN	Baltimore Highlands	(39.29591000 -76.56777000
8	11/12/2016	3BJ	5200 MAYVIEW	ROBBERY -	O	442	NORTHEASTERN	Frankford	(39.33177000

```
# drop missing value
X = X.dropna()
X.head(10)
```

```
Out[17]:
```

	CrimeDate	CrimeCode	Location	Description	Inside/Outside	Post	District	Neighborhood	Locatio
0	11/12/2016	3B	300 SAINT PAUL PL	ROBBERY - STREET	O	111	CENTRAL	Downtown	(39.2924100 -76.6140800
1	11/12/2016	3CF	800 S BROADWAY	ROBBERY - COMMERCIAL	I	213	SOUTHEASTERN	Fells Point	(39.2824200 -76.5928800
2	11/12/2016	6D	1500 PENTWOOD RD	LARCENY FROM AUTO	O	413	NORTHEASTERN	Stonewood-Pentwood-Winston	(39.3480500 -76.5883400
3	11/12/2016	6D	6600 MILTON LN	LARCENY FROM AUTO	O	424	NORTHEASTERN	Westfield	(39.3626300 -76.5516100
4	11/12/2016	6E	300 W BALTIMORE ST	LARCENY	O	111	CENTRAL	Downtown	(39.2893800 -76.6197100
5	11/12/2016	4E	6900 MCCLEAN BLVD	COMMON ASSAULT	I	423	NORTHEASTERN	Hamilton Hills	(39.3707000 -76.5670900
6	11/12/2016	3CO	1700 W LOMBARD ST	ROBBERY - COMMERCIAL	O	933	SOUTHERN	Union Square	(39.2862400 -76.6445500
7	11/12/2016	6D	0 N CONKLING ST	LARCENY FROM AUTO	O	223	SOUTHEASTERN	Baltimore Highlands	(39.2959100 -76.5677700
			5200	ROBBERY -					(39.3317700

```
# get the shape after drop uncommon data type
print ("Num of rows after drop uncommon data type: " + str(X.shape[0]))
print ("Num of columns after drop uncommon data type: " + str(X.shape[1]))
```

```
Num of rows after drop uncommon data type: 270024
Num of columns after drop uncommon data type: 11
```

```
# transfer from pandas dataframe into spark dataframe
spark_df = sqlContext.createDataFrame(X)
spark_df.show()
```

```
+----------+---------+--------------------+--------------------+--------------+----+------------+-
-------------------+--------------------+---------------+----------+
| CrimeDate|CrimeCode|            Location|         Description|Inside/Outside|Post|    District|
Neighborhood|          Location 1|Total Incidents|Crime_Time|
+----------+---------+--------------------+--------------------+--------------+----+------------+-
-------------------+--------------------+---------------+----------+
|11/12/2016|       3B|   300 SAINT PAUL PL|    ROBBERY - STREET|             O| 111|     CENTRAL|
Downtown|(39.2924100000, -...|              1|  02:35:00|
|11/12/2016|      3CF|      800 S BROADWAY|ROBBERY - COMMERCIAL|             I| 213|SOUTHEASTERN|
Fells Point|(39.2824200000, -...|              1|  02:56:00|
|11/12/2016|       6D|    1500 PENTWOOD RD|   LARCENY FROM AUTO|             O| 413|NORTHEASTERN|S
tonewood-Pentwoo...|(39.3480500000, -...|              1|  03:00:00|
|11/12/2016|       6D|      6600 MILTON LN|   LARCENY FROM AUTO|             O| 424|NORTHEASTERN|
Westfield|(39.3626300000, -...|              1|  03:00:00|
|11/12/2016|       6E|  300 W BALTIMORE ST|             LARCENY|             O| 111|     CENTRAL|
Downtown|(39.2893800000, -...|              1|  03:00:00|
|11/12/2016|       4E|   6900 MCCLEAN BLVD|      COMMON ASSAULT|             I| 423|NORTHEASTERN|
Hamilton Hills|(39.3707000000, -...|              1|  03:00:00|
|11/12/2016|      3CO|   1700 W LOMBARD ST|ROBBERY - COMMERCIAL|             O| 933|    SOUTHERN|
Union Square|(39.2862400000, -...|              1|  03:45:00|
```

```
# create sql envrionment
spark_df.createOrReplaceTempView("Baltimore_crime_table")
```

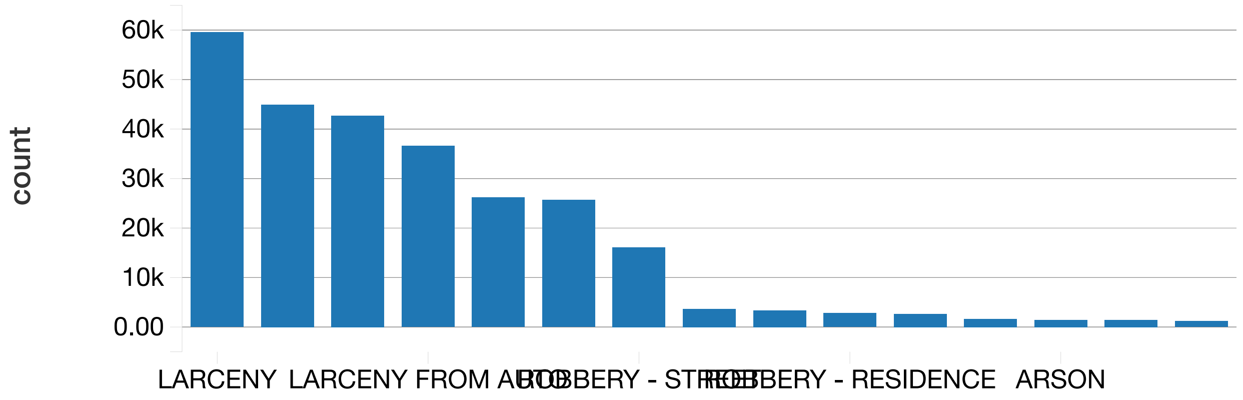
```
# change date type because of the transformation
from pyspark.sql.functions import *
df_update = spark_df.withColumn("CrimeDate", to_date(col("CrimeDate"), "MM/dd/yyyy")) ##change
datetype from string to date
df_update.createOrReplaceTempView("Baltimore_crime_table")
```

2. Data Analysis

Write a Spark program that counts the number of crimes for different category.

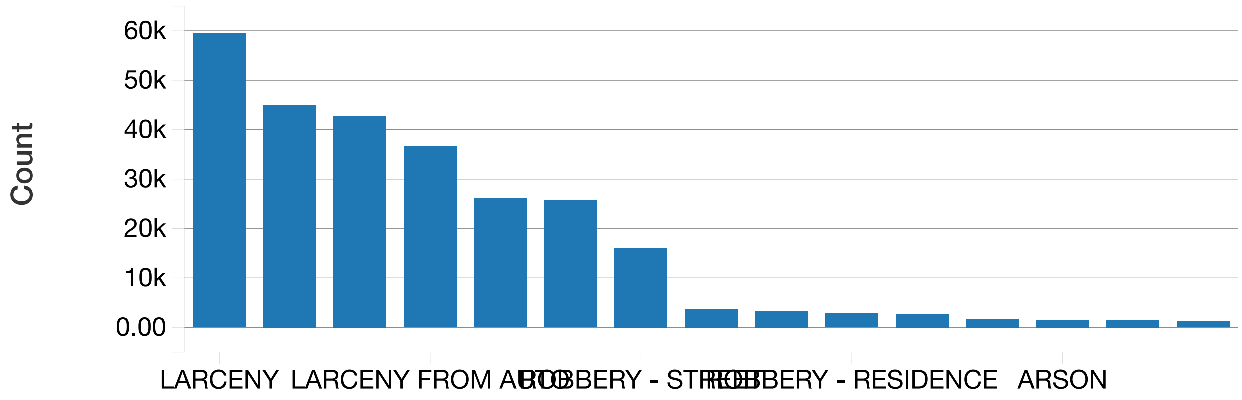
Spark dataframe based solution

```
q1_result = spark_df.groupBy('Description').count().orderBy('count', ascending=False)
display(q1_result)
```



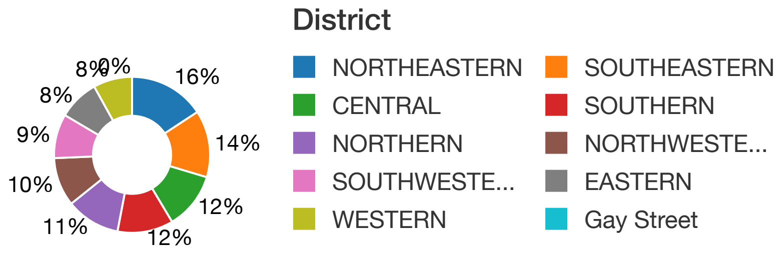
Spark SQL based solution

```
#Spark SQL based
crimeCategory = spark.sql("SELECT  Description, COUNT(*) AS Count FROM Baltimore_crime_table GROUP
BY 1 ORDER BY 2 DESC")
display(crimeCategory)
```



Counts the number of crimes for different district, and visualize your results

```
crime_nums = spark.sql("SELECT District, count(*) as count from Baltimore_crime_table group by 1
order by 2 DESC")
display(crime_nums)
```



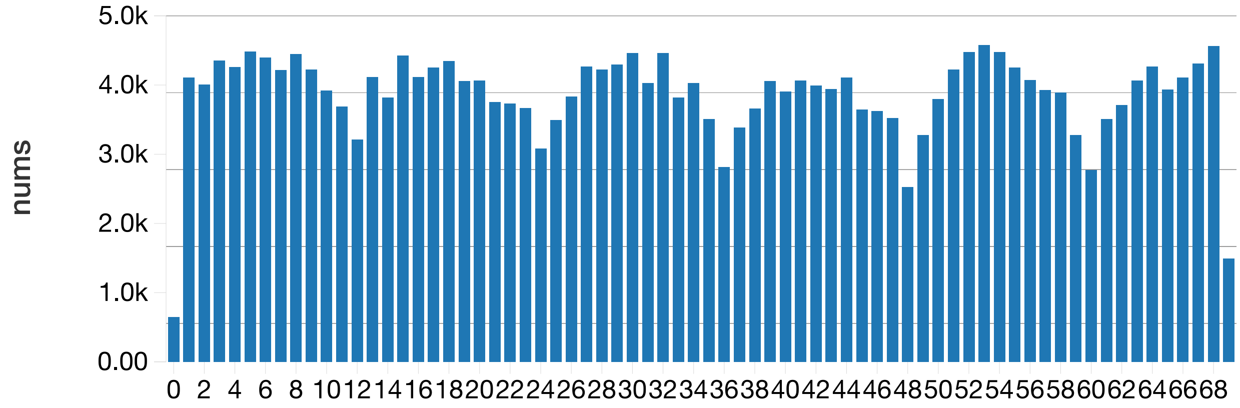
Count the number of crimes at "Baltimore downtown".

```
# check the google map, we find that downtwon area of Baltimore located between(39.28, -76.62) to
(39.32, -76.58)
down_town = X["Location 1"]
count = 0
for data in down_town:
    x1 = float(data[1:14])
    y1 = float(data[16:29])
    if 39.28 <= x1 <= 39.32 and -76.62 <= y1 <= -76.58:
      count += 1
print("the total numbers of crime in downtown Baltimore: " + str(count))
```

```
the total numbers of crime in downtown Baltimore: 58177
```

Analysis the number of crime in each month of year (2011-2016). Then, give your insights for the output results. What is the business impact for your result?

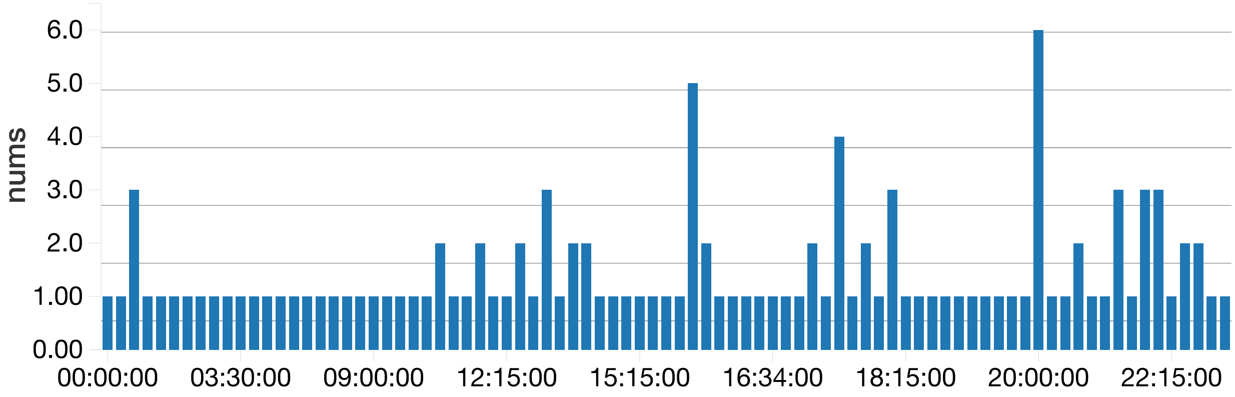
```
crimeYearMonth = spark.sql("SELECT Year(CrimeDate) as year, Month(CrimeDate) as month, count(*) as
nums FROM Baltimore_crime_table GROUP BY 1, 2 ORDER BY 1, 2")
display(crimeYearMonth)
```



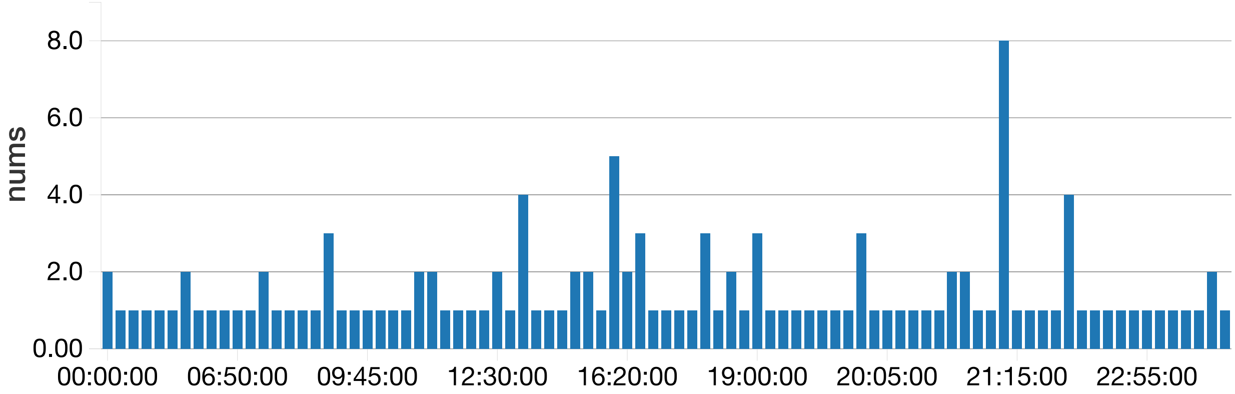
```
# from this table, the crime numbers are lowerst in December of each year, it means that criminal
may stay at home more frequently than any other month because of the incoming Christmars Day. So
for many Brick-and-mortar store, the owner can open store longer than usual before Christmars Day
to get more profit.
```

Analysis the number of crime w.r.t the hour in certian day like 2013/12/15, 2014/12/15, 2015/12/15. Then, give your travel suggestion to visit Baltimore.

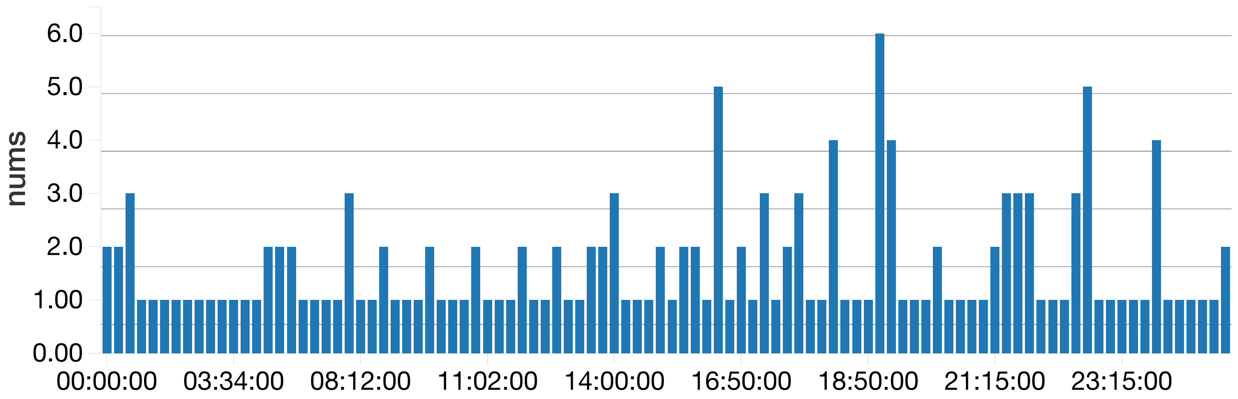
```
crime_nums_2013= spark.sql("SELECT Crime_Time, count (*) as nums FROM Baltimore_crime_table where
CrimeDate in (to_date('12/15/2013','MM/dd/yyyy')) group by 1 order by 1")
display(crime_nums_2013)
```



```
crime_nums_2014= spark.sql("SELECT Crime_Time, count (*) as nums FROM Baltimore_crime_table where
CrimeDate in (to_date('12/15/2014','MM/dd/yyyy')) group by 1 order by 1")
display(crime_nums_2014)
```



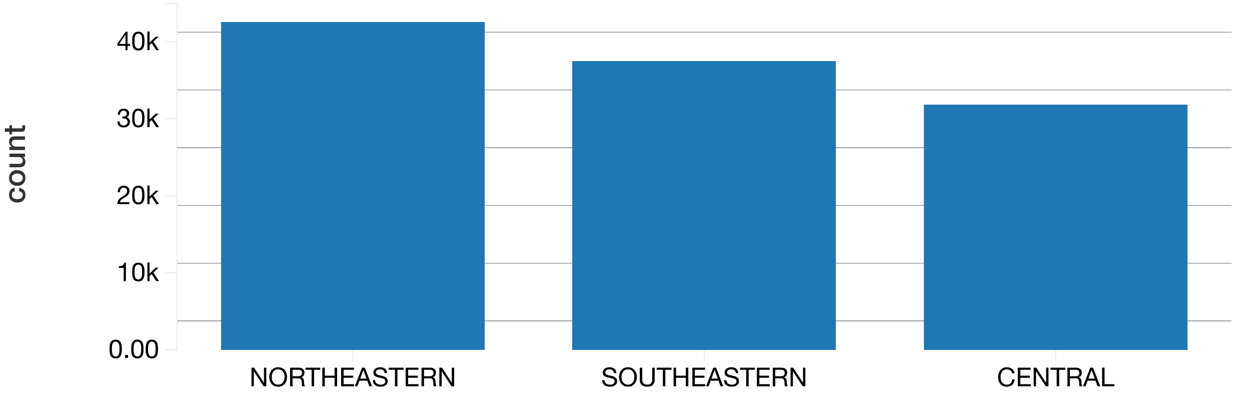
```
crime_nums_2015= spark.sql("SELECT Crime_Time, count (*) as nums FROM Baltimore_crime_table where
CrimeDate in (to_date('12/15/2015','MM/dd/yyyy')) group by 1 order by 1")
display(crime_nums_2015)
```



```
# from the visualization analysis, we can see that crime numbers are higher than any other time
from 19:00 to 22:00. So visitors should avoid of being there from 19:00 to 22.00, and the total
crime numbers increase from 2013-2015.
```

Find out the top-3 danger disrict

```
top3_crime_number = spark.sql("SELECT District, count(*) as count From Baltimore_crime_table Group
by 1 order by 2 DESC limit 3")
display(top3_crime_number)
```



For different category of crime, find the percentage of crime type. Based on the output, give your hints to adjust the policy.

```
crime_type = X["Inside/Outside"]
count = 0
for cnt in crime_type:
  if cnt == "I":
    count += 1
print("The percentage of Inside crime type:" + str(count/len(crime_type)))
print("The percentage of Outside crime type:" + str(1 - count/len(crime_type)))
```

```
The percentage of Inside crime type:0.5035996800284419
The percentage of Outside crime type:0.4964003199715581
```

```
# based on this output, the inside crime rate and outside crime rate almost the same, so the policy should balance the number of police between community and public place.
```

Conclusion.

```
# 1. This is a crime analysis project, which focus on analyzing crime trend and factors which influence the crime rate in Baltimore.
# 2. This data set comes from public resources in Washington DC, which records the crime information from 2011-2016.
# 3. This is an unstructured data set, so i need to deal with it by building data pipeline to further analyze.
# 4. I set up 3 main steps for data cleaning and exploration, data analysis, data modeling and visualization.
# 5. First of all, i use spark datasframe to create table and set up environment, and then transfer to Pandas dataframe to understand data information, slove missing value and uncommon data type information. Secondly, i use Spark SQL to analyze crime numbers with respect to different features and get some significant insights. Finally, i use Spark ML to build clustering model to visualize the results by clustering the data set.
# 6. By analyzing the data set, i draw a conclusion that visitors need to choose suitable time to go to Baltimore avoid time from 19:00 to 22:00. At the same time, i don't suggest visitors to go to the northeastern, southeastern, central street and some adjacent locations from clustering result. But for the owners of the stores, i suggset that they can open longer to make more money, and people can invite their family members to their house to a enjoy good time in December. Additionally, downtown is safer than any other places in Baltimore.
```

3. Modeling

```
from pyspark.sql.types import DoubleType
changedTypedf = spark_df.withColumn("Post", spark_df["Post"].cast(DoubleType()))
changedTypedf.show()
```

```
+----------+---------+--------------------+--------------------+--------------+-----+------------+
--------------------+--------------------+---------------+----------+
| CrimeDate|CrimeCode|            Location|         Description|Inside/Outside| Post|    District|
Neighborhood|          Location 1|Total Incidents|Crime_Time|
+----------+---------+--------------------+--------------------+--------------+-----+------------+
--------------------+--------------------+---------------+----------+
|11/12/2016|       3B|   300 SAINT PAUL PL|    ROBBERY - STREET|             O|111.0|     CENTRAL|
Downtown|(39.2924100000, -...|              1|  02:35:00|
|11/12/2016|      3CF|      800 S BROADWAY|ROBBERY - COMMERCIAL|             I|213.0|SOUTHEASTERN|
Fells Point|(39.2824200000, -...|              1|  02:56:00|
|11/12/2016|       6D|    1500 PENTWOOD RD|   LARCENY FROM AUTO|             O|413.0|NORTHEASTERN|
Stonewood-Pentwoo...|(39.3480500000, -...|              1|  03:00:00|
|11/12/2016|       6D|      6600 MILTON LN|   LARCENY FROM AUTO|             O|424.0|NORTHEASTERN|
Westfield|(39.3626300000, -...|              1|  03:00:00|
|11/12/2016|       6E|  300 W BALTIMORE ST|             LARCENY|             O|111.0|     CENTRAL|
Downtown|(39.2893800000, -...|              1|  03:00:00|
|11/12/2016|       4E|   6900 MCCLEAN BLVD|      COMMON ASSAULT|             I|423.0|NORTHEASTERN|
Hamilton Hills|(39.3707000000, -...|              1|  03:00:00|
|11/12/2016|      3CO|   1700 W LOMBARD ST|ROBBERY - COMMERCIAL|             O|933.0|    SOUTHERN|
Union Square|(39.2862400000, -...|              1|  03:45:00|
```

```python
from pyspark.ml.feature import VectorAssembler
vecAssembler = VectorAssembler(inputCols=["Post"], outputCol="features")
new_df = vecAssembler.transform(changedTypedf)
new_df.show()
```

```
+----------+---------+--------------------+--------------------+--------------+-----+------------+
--------------------+--------------------+---------------+----------+--------+
| CrimeDate|CrimeCode|            Location|         Description|Inside/Outside| Post|    District|
Neighborhood|          Location 1|Total Incidents|Crime_Time|features|
+----------+---------+--------------------+--------------------+--------------+-----+------------+
--------------------+--------------------+---------------+----------+--------+
|11/12/2016|       3B|   300 SAINT PAUL PL|    ROBBERY - STREET|             O|111.0|     CENTRAL|
Downtown|(39.2924100000, -...|              1|  02:35:00| [111.0]|
|11/12/2016|      3CF|      800 S BROADWAY|ROBBERY - COMMERCIAL|             I|213.0|SOUTHEASTERN|
Fells Point|(39.2824200000, -...|              1|  02:56:00| [213.0]|
|11/12/2016|       6D|    1500 PENTWOOD RD|   LARCENY FROM AUTO|             O|413.0|NORTHEASTERN|
Stonewood-Pentwoo...|(39.3480500000, -...|              1|  03:00:00| [413.0]|
|11/12/2016|       6D|      6600 MILTON LN|   LARCENY FROM AUTO|             O|424.0|NORTHEASTERN|
Westfield|(39.3626300000, -...|              1|  03:00:00| [424.0]|
|11/12/2016|       6E|  300 W BALTIMORE ST|             LARCENY|             O|111.0|     CENTRAL|
Downtown|(39.2893800000, -...|              1|  03:00:00| [111.0]|
|11/12/2016|       4E|   6900 MCCLEAN BLVD|      COMMON ASSAULT|             I|423.0|NORTHEASTERN|
Hamilton Hills|(39.3707000000, -...|              1|  03:00:00| [423.0]|
|11/12/2016|      3CO|   1700 W LOMBARD ST|ROBBERY - COMMERCIAL|             O|933.0|    SOUTHERN|
Union Square|(39.2862400000, -...|              1|  03:45:00| [933.0]|
|11/12/2016|       6D|     0 N CONKLING ST|   LARCENY FROM AUTO|             O|223.0|SOUTHEASTERN|
```

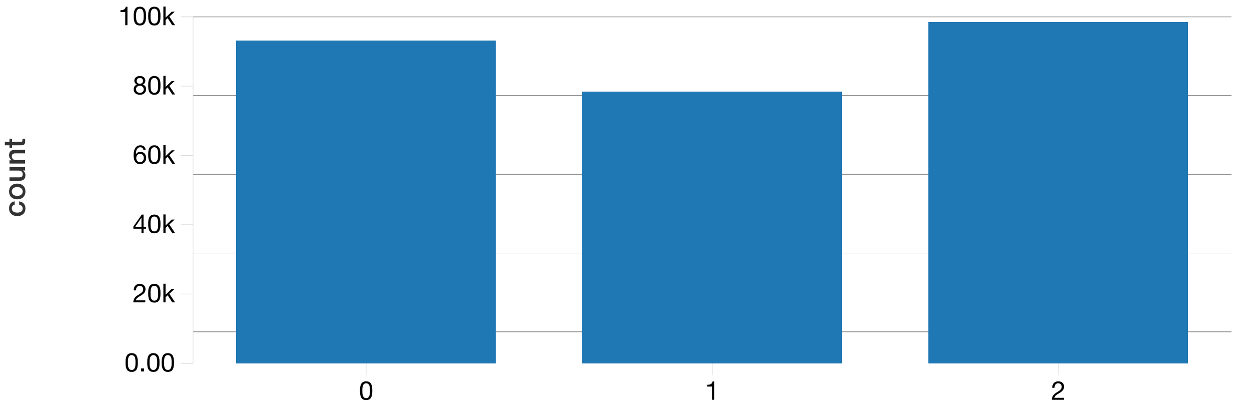
```python
from pyspark.ml.clustering import KMeans

kmeans = KMeans(k=3, seed=1)  # 3 clusters here
model = kmeans.fit(new_df.select('features'))



transformed = model.transform(new_df)
transformed.show()
```

```
+----------+---------+--------------------+--------------------+--------------+-----+------------+
--------------------+--------------------+---------------+----------+--------+----------+
| CrimeDate|CrimeCode|            Location|         Description|Inside/Outside| Post|    District|
Neighborhood|          Location 1|Total Incidents|Crime_Time|features|prediction|
+----------+---------+--------------------+--------------------+--------------+-----+------------+
--------------------+--------------------+---------------+----------+--------+----------+
|11/12/2016|       3B|   300 SAINT PAUL PL|    ROBBERY - STREET|             O|111.0|     CENTRAL|
Downtown|(39.2924100000, -...|              1|  02:35:00| [111.0]|         1|
|11/12/2016|      3CF|      800 S BROADWAY|ROBBERY - COMMERCIAL|             I|213.0|SOUTHEASTERN|
Fells Point|(39.2824200000, -...|              1|  02:56:00| [213.0]|         1|
|11/12/2016|       6D|    1500 PENTWOOD RD|   LARCENY FROM AUTO|             O|413.0|NORTHEASTERN|
Stonewood-Pentwoo...|(39.3480500000, -...|              1|  03:00:00| [413.0]|         0|
|11/12/2016|       6D|      6600 MILTON LN|   LARCENY FROM AUTO|             O|424.0|NORTHEASTERN|
Westfield|(39.3626300000, -...|              1|  03:00:00| [424.0]|         0|
|11/12/2016|       6E|  300 W BALTIMORE ST|             LARCENY|             O|111.0|     CENTRAL|
Downtown|(39.2893800000, -...|              1|  03:00:00| [111.0]|         1|
|11/12/2016|       4E|   6900 MCCLEAN BLVD|      COMMON ASSAULT|             I|423.0|NORTHEASTERN|
Hamilton Hills|(39.3707000000, -...|              1|  03:00:00| [423.0]|         0|
|11/12/2016|      3CO|   1700 W LOMBARD ST|ROBBERY - COMMERCIAL|             O|933.0|    SOUTHERN|
Union Square|(39.2862400000, -...|              1|  03:45:00| [933.0]|         2|
|11/12/2016|       6D|     0 N CONKLING ST|   LARCENY FROM AUTO|             O|223.0|SOUTHEASTERN|
```

```
# Shows the result.
centers = model.clusterCenters()
print("Cluster Centers: ")
for center in centers:
    print(center)
```

```
Cluster Centers: 
[509.9858846]
[214.80034982]
[840.37942828]
```

```
transformed.createOrReplaceTempView("New_Baltimore_crime_table")
```


100k
80k
60k
40k
20k
count