



Data Analysis of Toronto Crime - 2016

This paper focuses on detailed analysis of the crime events those occurred in the City of Toronto in the year 2016.

```
#import the required packages
from pyspark import SparkContext
from pyspark.sql import SQLContext
import pandas as pd

#read the csv data file
rawdata_crime2016 = sqlContext.read.format("com.databricks.spark.csv").options(header='true').option("inferSchema", "true").load("/FileStore/tables/MCI_2016.csv")

#Lets view the sample data
display(rawdata_crime2016.limit(5))
```

X	Y	Index_	event_unique_id	occurrencedate	reporteddate	premisetype	ucr_code	ucr_ext	offence	reportedyear	r		
-79.58380727	43.74681421	5301	GO-2016321215	2016-02-16T14:00:00.000+0000	2016-02-19T15:00:00.000+0000	Other	1430	100	Assault	2016	February	19	50
-79.46574586000001	43.66966535000001	5302	GO-2016343685	2016-02-27T09:10:00.000+0000	2016-02-27T09:28:00.000+0000	Apartment	1430	100	Assault	2016	February	27	58
-79.20497568	43.78260125	5303	GO-2016344019	2016-02-27T10:45:00.000+0000	2016-02-27T12:04:00.000+0000	Other	1430	100	Assault	2016	February	27	58
-79.27951692	43.70321859	5304	GO-2016343554	2016-02-27T08:36:00.000+0000	2016-02-27T08:36:00.000+0000	Apartment	1430	100	Assault	2016	February	27	58



```
#Let's check the number of rows on the datasets
rawdata_crime2016.count()
```

```
Out[5]: 32612
```

```
#Checking the statistical view of the dataset
display(rawdata_crime2016.describe())
```

summary ▼	X ▼	Y ▼	Index_ ▼	event_unique_id ▼	premisetype ▼	ucr_code ▼	ucr_ext ▼	offence ▼	reportedyear ▼			
count	32612	32612	32612	32612	32612	32612	32612	32612	32612	32612	32612	32612
mean	-79.39571004298898	43.70777205885616	16373.986232061818	null	null	1677.9926714092971	143.40245921746597	null	2016.0	null	15.832883601128419	185.631
stddev	0.1054783620635244	0.05244439509894676	9534.304059636062	null	null	317.3225559703621	51.18191280063978	null	0.0	null	8.794617039916382	104.294
min	-79.63524142	43.58709195	1	GO-20161000017	Apartment	1410	100	Administering Noxious Thing	2016	April	1	1
max	-79.1220565	43.84620382	34755	GO-2016999983	Outside	2135	230	Use Firearm / Immit Commit Off	2016	September	31	366



From the above statistical report, it can be seen that 5 records doesn't have occurenceday,occurrencemonth and occurrenceyear value. Let's pull those 5 records and get more details.

```
#getting records if Null
display(rawdata_crime2016.where(rawdata_crime2016.occurrencemonth.isNull()))
```

X ▼	Y ▼	Index_ ▼	event_unique_id ▼	occurrencedate ▼	reporteddate ▼	premisetype ▼	ucr_code ▼	ucr_ext ▼	offence ▼	reportedyear ▼	reportedmonth			
-79.4272318	43.79296399	2830	GO-2016988968	1974-01-01T21:00:00.000+0000	2016-06-07T16:50:00.000+0000	House	1430	100	Assault	2016	June	7	159	Tues
-79.31765241	43.66889046	20178	GO-20161667005	1996-01-01T05:00:00.000+0000	2016-09-19T18:20:00.000+0000	Other	2135	210	Theft Of Motor Vehicle	2016	September	19	263	Monc
-79.31765241	43.66889046	20179	GO-20161667005	1996-01-01T05:00:00.000+0000	2016-09-19T18:20:00.000+0000	Other	2135	210	Theft Of Motor Vehicle	2016	September	19	263	Monc
-79.42659008	43.65151427	34254	GO-2016203004	1992-12-04T05:04:00.000+0000	2016-02-02T10:50:00.000+0000	Other	2130	210	Theft Over	2016	February	3	34	Wedn



Data Cleaning - As it can be seen, those 5 records were occurred in the past, not in 2016 but was reported in 2016. Hence, let's create a new dataframe removing those 5 records and only keeping those records occurred in 2016

```
#create new dataframe crime2016
crime2016=rawdata_crime2016.where(rawdata_crime2016.occurrencemonth.isNotNull() & (rawdata_crime2016.occurrenceyear==2016))
```

```
#verify the total records in the datasets
crime2016.count()
```

```
Out[9]: 32102
```

Getting the distinct values for Major Crime Indicator column

```
crime2016.select('MCI').distinct().show()
```

	MCI
Assault	
Robbery	
Theft Over	
Break and Enter	
Auto Theft	

Data Analysis through charts

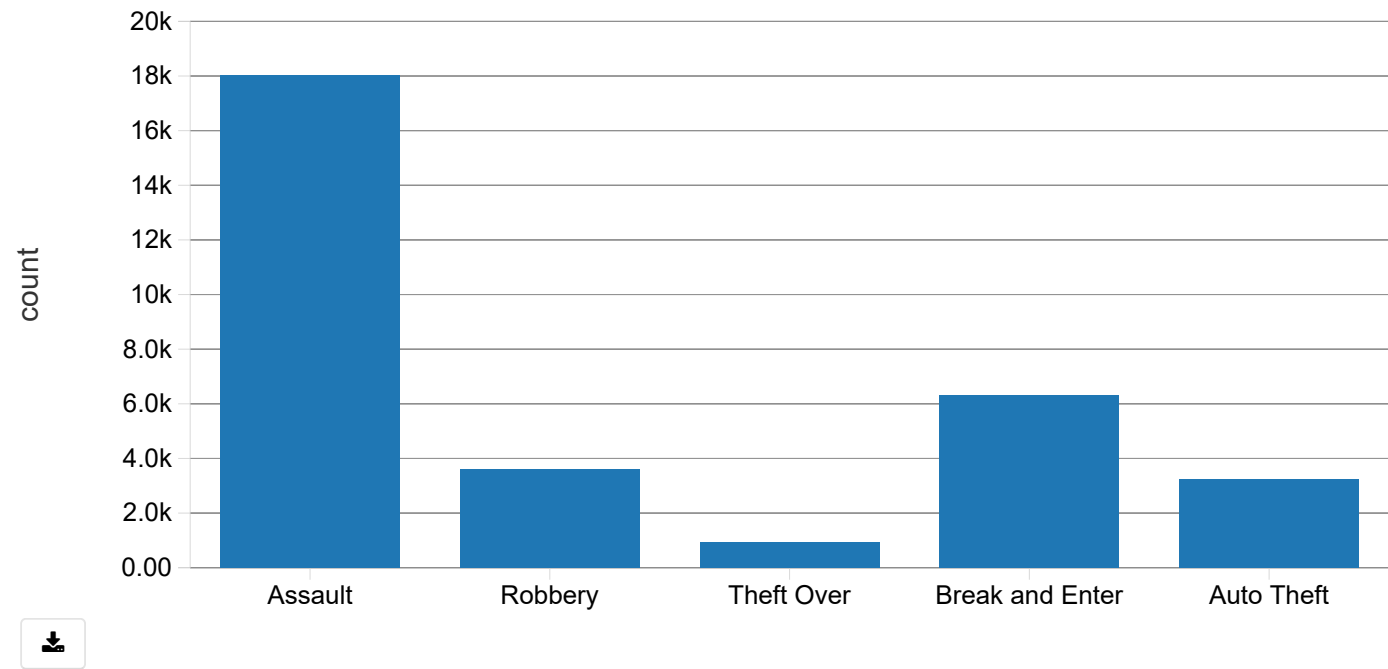
Number of occurrence for each MCI and plot the graph to represent

```
# importing required package
import pandas as pd
import matplotlib.pyplot as plt
#%matplotlib inline
import requests
requests.packages.urllib3.disable_warnings()
```

1.1 Below is the graphical represent of each Major Crime for the year 2016. Grouping by the Major Crime Indicator and displaying with bar chart

Conclusion: It can be concluded that Assault has the highest occurrence compared to other crimes. And theft is the lowest.

```
#Group by the Major Crime Indicator and displaying with bar chart
crime_by_type = crime2016.select('MCI').groupby('MCI').count()
display(crime_by_type)
```



1.2 Crime occurence as per the Month.

Now let's check the crime for each month in the year 2016.

```
crime_byMonth = crime2016.select('occurrencemonth').groupby('occurrencemonth').count().orderBy('occurrencemonth')
display(crime_byMonth)
```

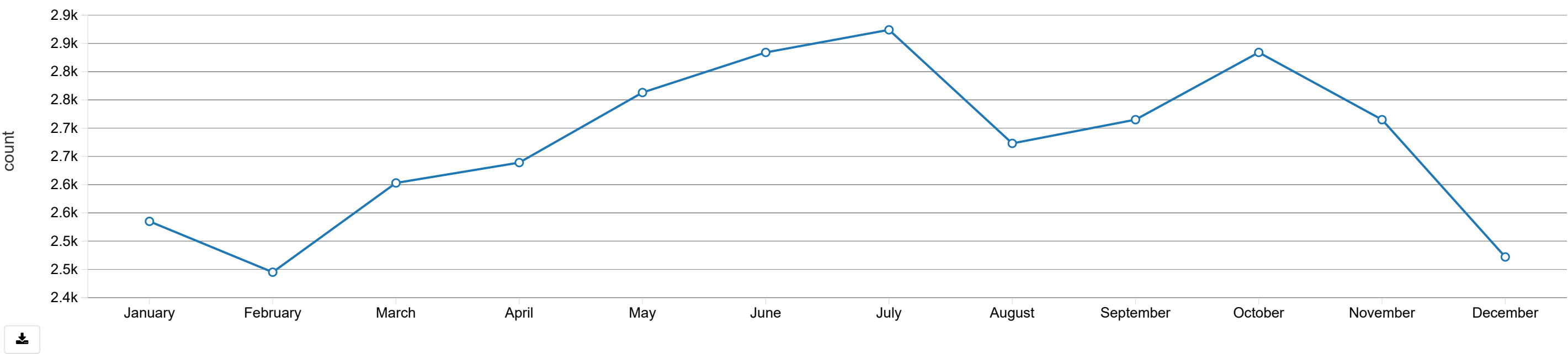
occurrencemonth	count
April	2639
August	2673
December	2472
February	2445
January	2535
July	2874
June	2834
March	2603
May	2763

From the above results, it can be seen that the month order is not as per the calendar month. So we need to create a new dataframe, by adding new column 'month_num' with the numerical value for each month.

```
from pyspark.sql.functions import *

crime_byMonth_sort = crime_byMonth.withColumn("month_num", when(crime_byMonth.occurrencemonth == 'January','01').when(crime_byMonth.occurrencemonth ==
'February','02').when(crime_byMonth.occurrencemonth == 'March','03').when(crime_byMonth.occurrencemonth == 'April','04').when(crime_byMonth.occurrencemonth ==
'May','05').when(crime_byMonth.occurrencemonth == 'June','06').when(crime_byMonth.occurrencemonth == 'July','07').when(crime_byMonth.occurrencemonth ==
'August','08').when(crime_byMonth.occurrencemonth == 'September','09').when(crime_byMonth.occurrencemonth == 'October','10').when(crime_byMonth.occurrencemonth ==
'November','11').when(crime_byMonth.occurrencemonth == 'December','12').otherwise(crime_byMonth.occurrencemonth))
```

```
display(crime_byMonth_sort.select('occurrencemonth', 'count').orderBy('month_num'))
```



Conclusion : Crime rate is low from October to December and Jan- Feb, may be because of the winter. From the above graph, it is clear that there was slight slope down in February and then it keeps on increasing untill July. In August it came down a bit and went up in October and decline toward the end of the year.

1.3 Crime occurence as per the Neighbourhood

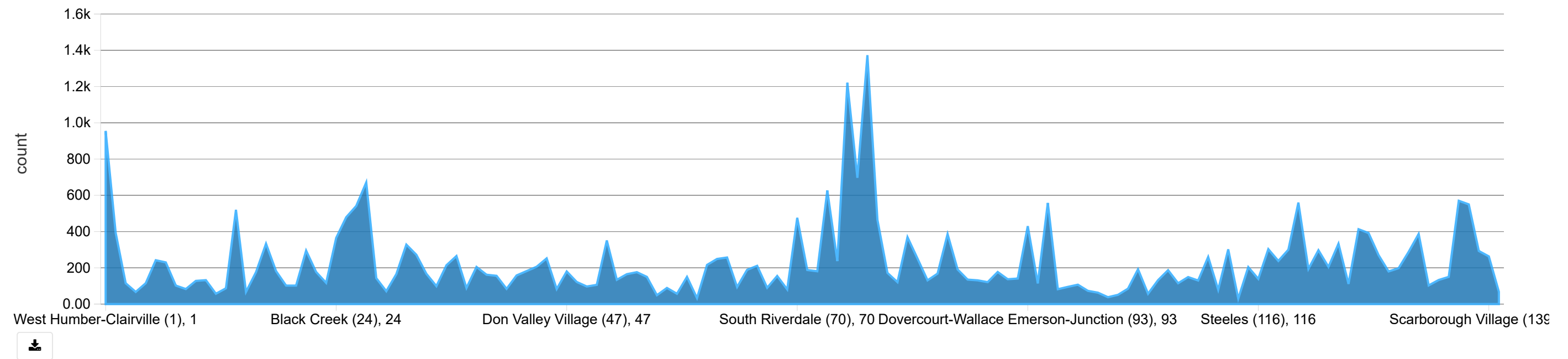
Let's find distinct Neighbourhood. From the result it's seen that, there are total 140 different sub region of Toronto

```
display(crime2016.select('Hood_ID','Neighbourhood').distinct().orderBy('Hood_ID'))
```

Hood_ID	Neighbourhood
1	West Humber-Clairville (1)
2	Mount Olive-Silverstone-Jamestown (2)
3	Thistletown-Beaumond Heights (3)
4	Rexdale-Kipling (4)
5	Elms-Old Rexdale (5)
6	Kingsview Village-The Westway (6)
7	Willowridge-Martingrove-Richview (7)
8	Humber Heights-Westmount (8)
9	Edenbridge-Humber Valley (9)

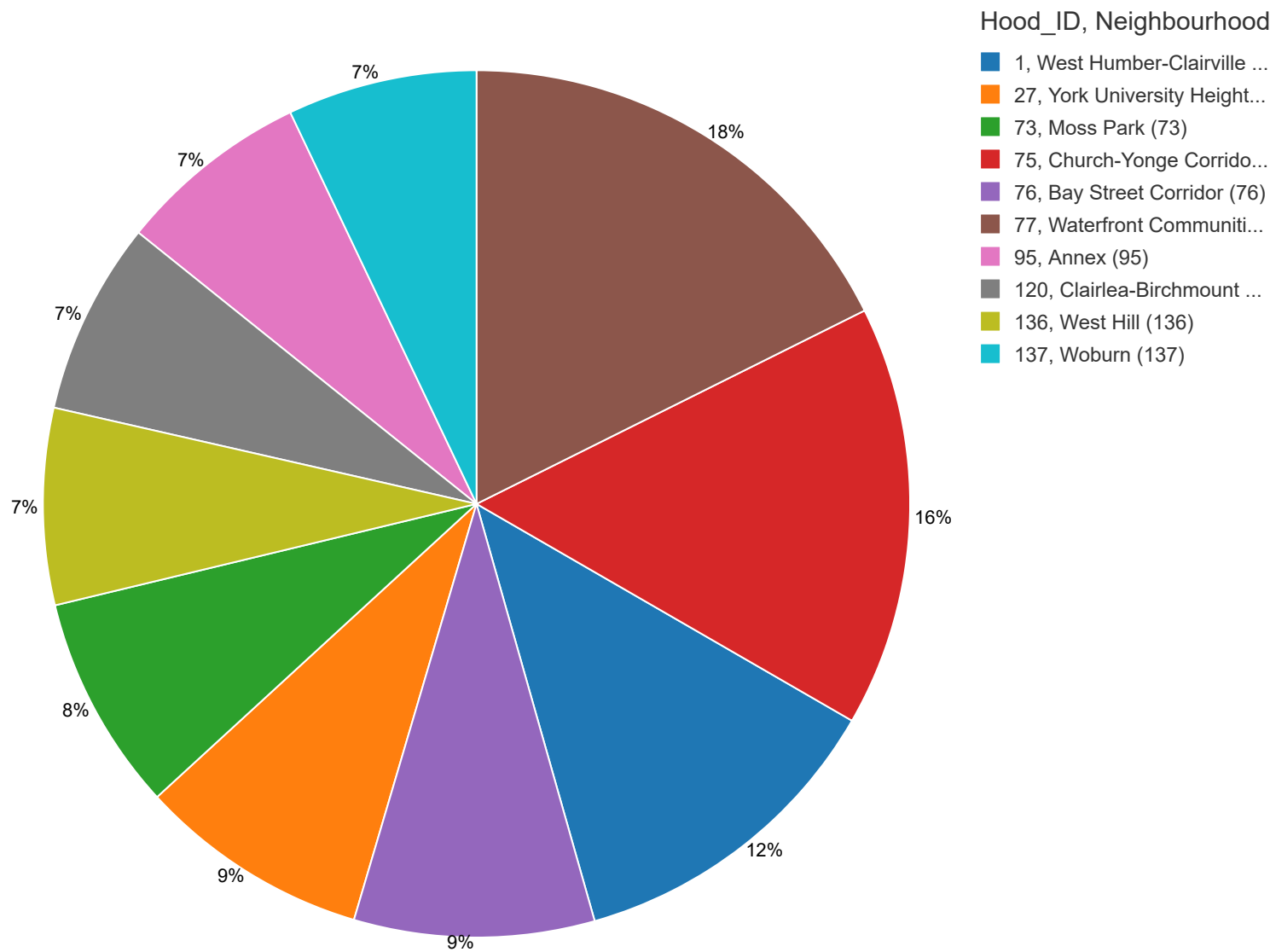
Plot a graph for each subregion, the crime occurence count

```
crime_byregion = crime2016.select('Neighbourhood','Hood_ID').groupBy('Neighbourhood','Hood_ID').count().orderBy('Hood_ID')
display(crime_byregion)
```



1.4 Crime Occurence for 10 most affected neighbourhood

```
crime_byregion_top10 = crime_byregion.select('Neighbourhood', 'Hood_ID', 'count').orderBy(desc('count')).limit(10)
display(crime_byregion_top10.orderBy('Hood_ID'))
```



```
crime_byregion_percent=crime_byregion_top10.withColumn('Percent', when( crime_byregion_top10['count'].cast('int') != 0, crime_byregion_top10['count']*100/32102) ).orderBy('count')
display(crime_byregion_percent.orderBy(desc('Percent')))
```

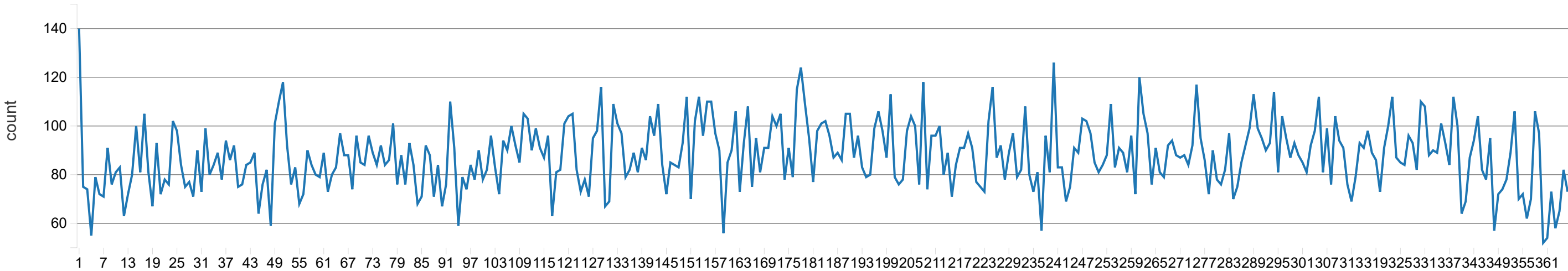
Neighbourhood	Hood_ID	count	Percent	
Waterfront Communities-The Island (77)	77		1373	4.276992087720391
Church-Yonge Corridor (75)	75		1221	3.8035013394804063
West Humber-Clairville (1)	1		955	2.9748925300604325
Bay Street Corridor (76)	76		697	2.1712042863373
York University Heights (27)	27		671	2.0902124478225654
Moss Park (73)	73		627	1.9531493364899384
West Hill (136)	136		570	1.7755903058999438
Clairlea-Birchmount (120)	120		560	1.7444395987788923
Annex (95)	95		558	1.738209457354682
Woburn (137)	137		550	1.7132888916578406



1.5 Crime Trend of the Daily Base

Let's find the graph on the daily basis

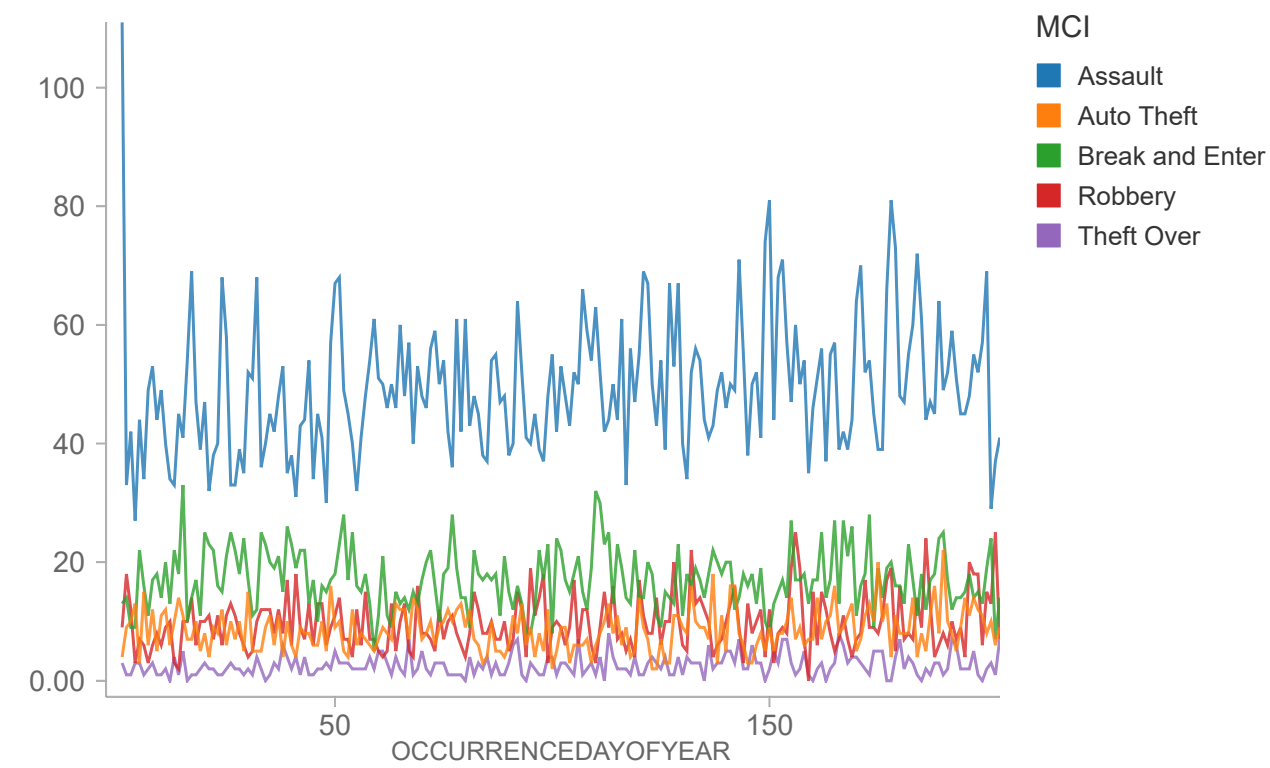
```
display(crime2016.where((crime2016.occurrencedayofyear == 1) & (crime2016.occurrenceyear==2016)))
crime_year_trend = crime2016.select('occurrencedayofyear').where((crime2016.occurrenceyear==2016)).groupby('occurrencedayofyear').count().orderBy('occurrencedayofyear')
display(crime_year_trend)
```



1.6 Crime Trend of the year for each crime type

Finding the crime on the daily basis for each major crime indicator.

```
crime_year_trend_mci = crime2016.select('occurrencedayofyear','MCI').where((crime2016.occurrenceyear==2016)).groupby('occurrencedayofyear','MCI').count().orderBy('occurrencedayofyear')
display(crime_year_trend_mci)
```

Aggregated in the backend.

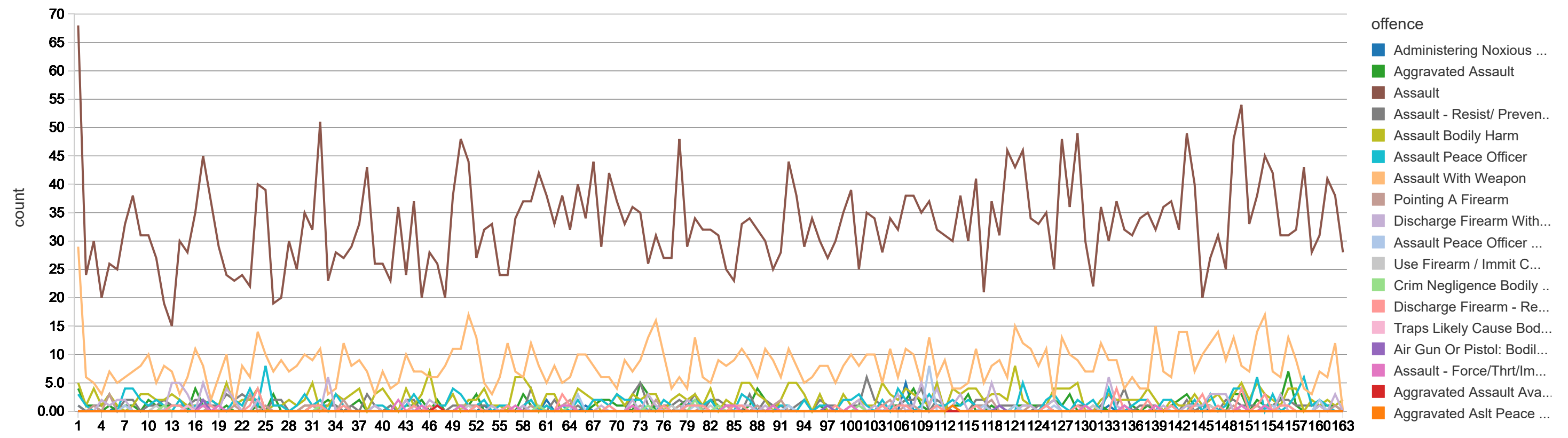
Showing the first 1000 rows.



Crime details for each offence under Assault

Checking the details for each offence under the MCI type Assault, as Assault shares more number of crime events.

```
crime_year_trend_assault = crime2016.select('occurrencedayofyear','offence').where((crime2016.MCI == 'Assault')).groupby('occurrencedayofyear','offence').count().orderBy('occurrencedayofyear','offence')
display(crime_year_trend_assault)
```

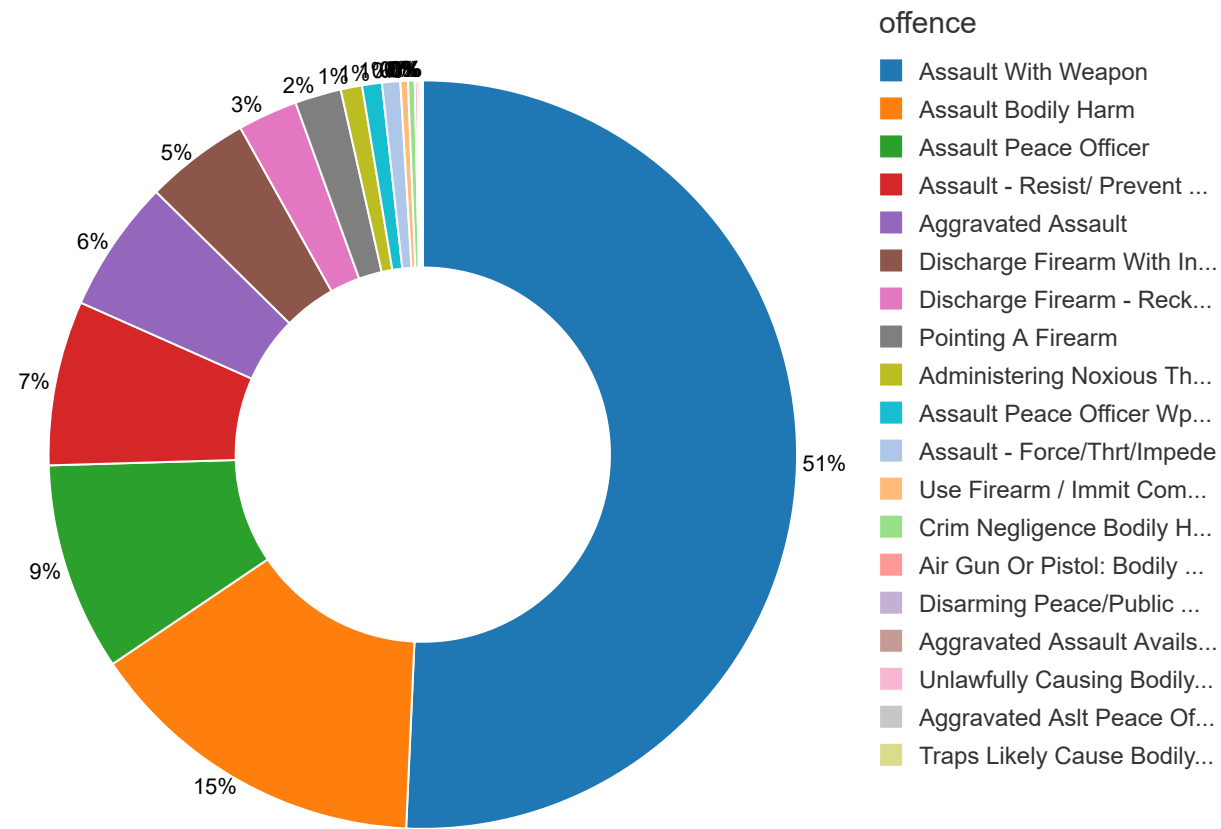


Aggregated in the backend.
Showing the first 1000 rows.



As Assault, consist default crime sub type 'Assault' and it has major part, let's not compare it and check the granulaity for other offence under Assault.

```
crime_year_trend = crime2016.select('offence').where((crime2016.MCI == 'Assault') & (crime2016.offence <> 'Assault')).groupby('offence').count().orderBy(desc('count'),'offence')
display(crime_year_trend)
```

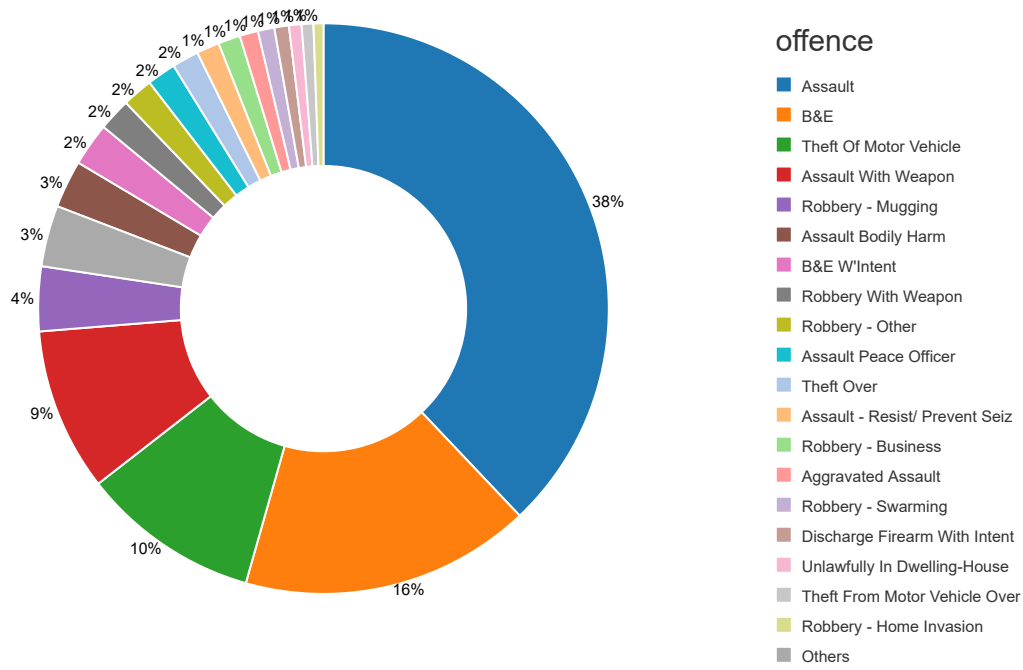


Conclusion : From the pie chart above, it's clear that Assault with weapon is highest crime rate around 51% ie 3000 event in a year, followed by body harm i.e 9% 900 events

1.7 Crime Rate for each offence

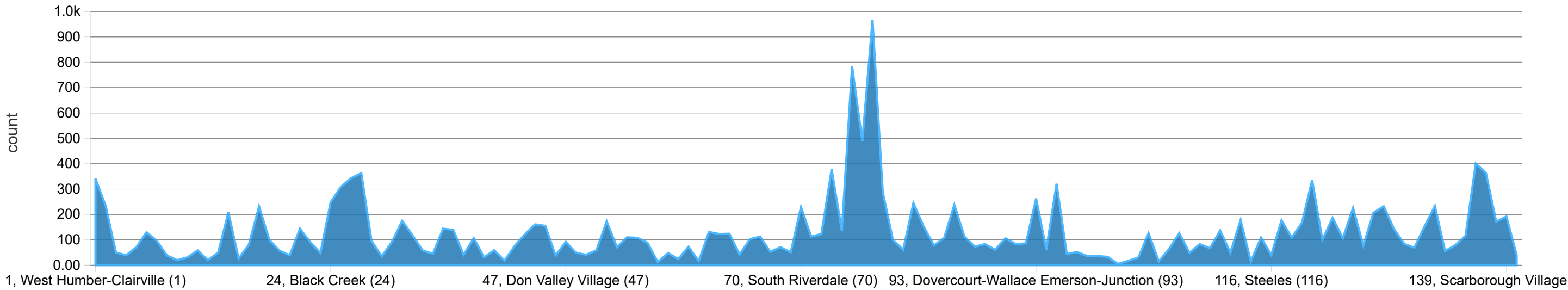
Let's remove the filte 'Assault' and find the crime rate for each offence for all the major crime type

```
crime_year_trend_offence = crime2016.select('offence').where((crime2016.occurrenceyear==2016)).groupby('offence').count().orderBy('offence')
display(crime_year_trend_offence)
```



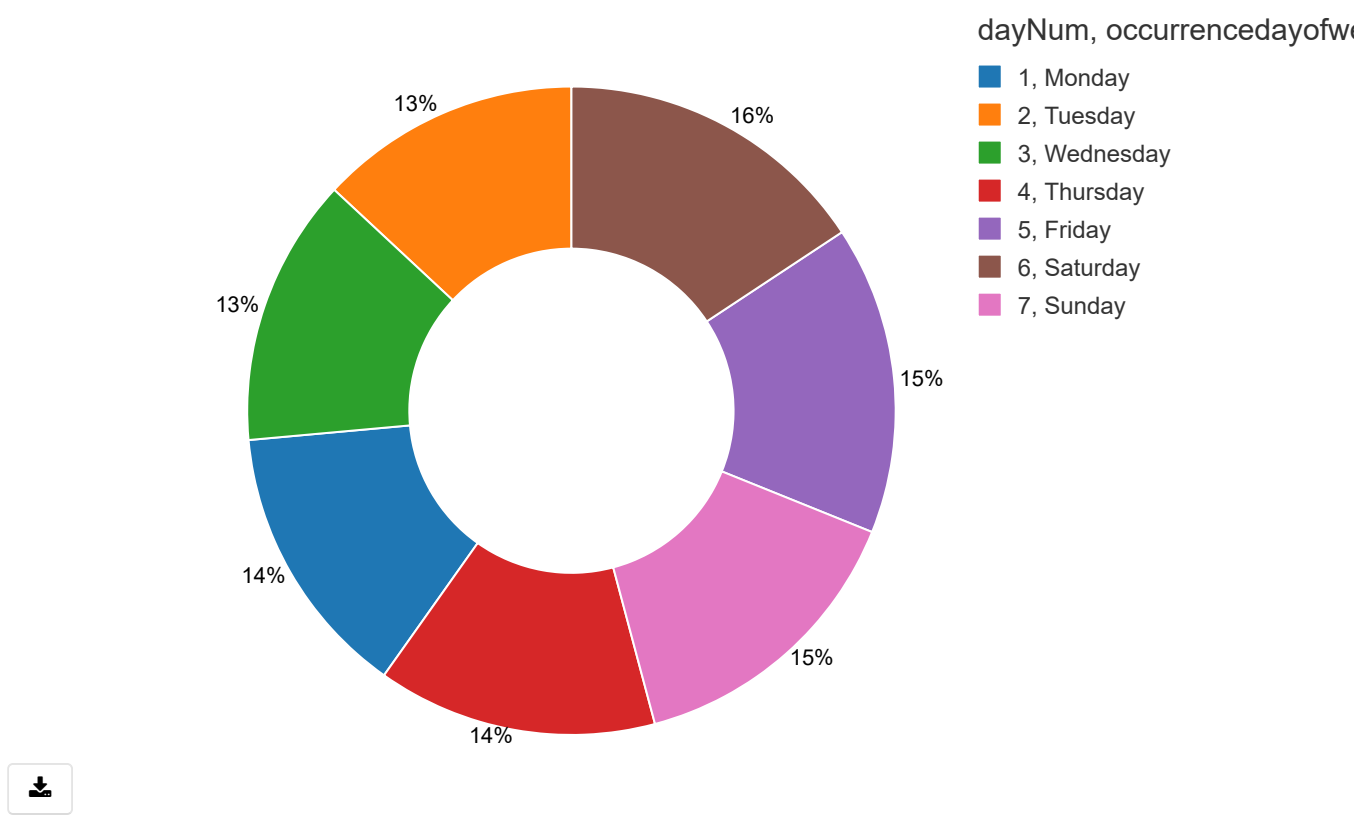
Conclusion : Seems like the major offence are Assault (38%) , Break and Enter (16%) and Theft of Motor Vehicle (10%)

```
#Check Assault as per neighbourhood
crime_byregion_assault = crime2016.select('Neighbourhood','Hood_ID').where((crime2016.MCI == 'Assault')).groupby('Neighbourhood','Hood_ID').count().orderBy('Hood_ID')
display(crime_byregion_assault)
```



1.8 Crime occurrence Day of the week

```
crime_day_of_week = crime2016.select('occurrencedayofweek').groupBy('occurrencedayofweek').count().orderBy('occurrencedayofweek')
crime_day_of_week_sort = crime_day_of_week.withColumn('dayNum',when(rtrim(crime_day_of_week.occurrencedayofweek) == 'Monday', '1').when(rtrim(crime_day_of_week.occurrencedayofweek) == 'Tuesday', '2').when(rtrim(crime_day_of_week.occurrencedayofweek) == 'Wednesday', '3').when(rtrim(crime_day_of_week.occurrencedayofweek) == 'Thursday', '4').when(rtrim(crime_day_of_week.occurrencedayofweek) == 'Friday', '5').when(rtrim(crime_day_of_week.occurrencedayofweek) == 'Saturday', '6').when(rtrim(crime_day_of_week.occurrencedayofweek) == 'Sunday', '7').otherwise('0'))
display(crime_day_of_week_sort.orderBy('dayNum'))
```

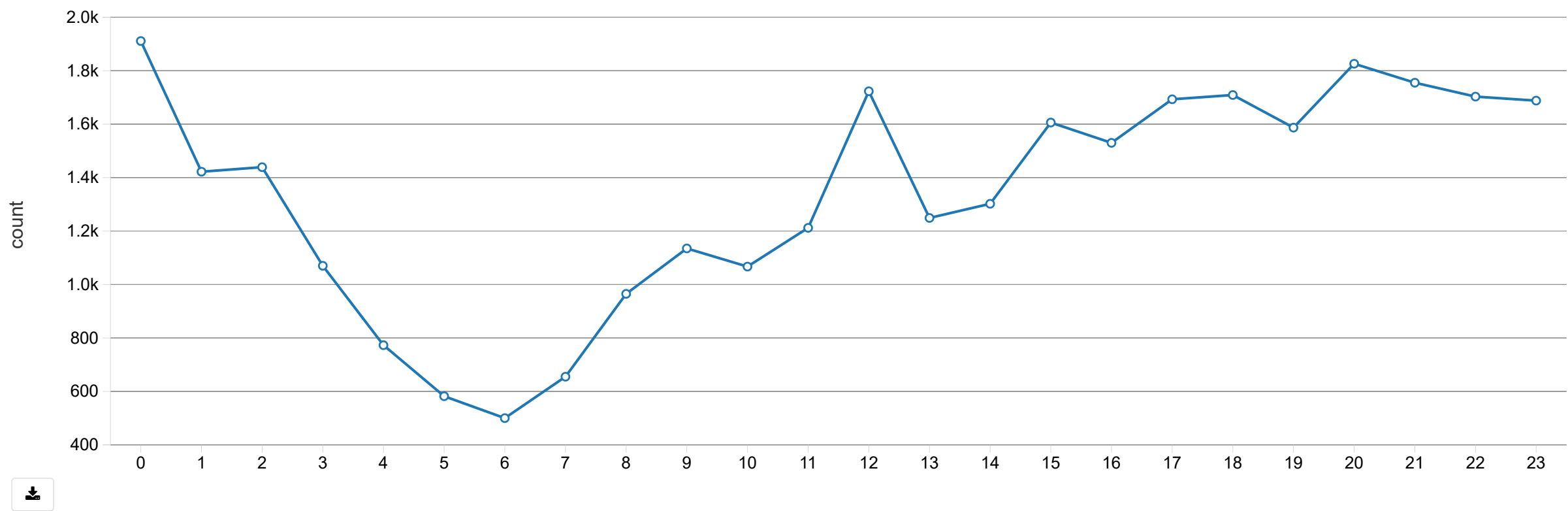


Conclusion : Seems like weekend comes with more crime. Weekend shares equal crime to the 4 days of the week, with Monday- Tuesday the lowest.

1.9 Crime for Hour of the day

Let's find which hour of the day is highly prone to crime.

```
crime_per_hour = crime2016.select('occurrencehour').groupBy('occurrencehour').count().orderBy('occurrencehour')
display(crime_per_hour)
```



Conclusion : It can seen that as the day progresses, crime rate increases. The safest hours are from 2:00 am to 11:00 am

Comparing crime occurence with the google trend data weekly

Let's compare our crime rate data with the web search done by Toronto people during the same time period.


As such google trend data set is in weekly basis, Let's convert your crime dataset on weekly basis. Also google datset is indexed as 100 for the week with highest seach and then indexing other weeks comparatively. So we need to transform our crime dataset accordingly.

```
crime_week_year= crime2016.select('Neighbourhood',crime2016.occurrencedate)

# Adding an additional columns as week number, using the function weekofyear()
crime_week_year =crime_week_year.withColumn('weekNum',weekofyear(crime2016.occurrencedate))
crime_week_year = crime_week_year.withColumn('date',(crime2016.occurrencedate.substr(1,10)))
crime_week_year = crime_week_year.select('weekNum').groupby('weekNum').count()
crime_week_year = crime_week_year.select('weekNum','count')

#Converting the actual count as the indexed value, 700 is the max crime rate weekly. So let's divide each week count by 700 and then multpling by 100
crime_week_year =crime_week_year.withColumn('crimeIndex',((crime_week_year['count']/700 )*100).cast('int'))
display(crime_week_year)
```

weekNum	count	crimeIndex
31	613	87
53	274	39
34	599	85

28	648	92
26	695	99
27	642	91
44	657	93
12	573	81
22	700	100
47	667	95
1	523	74
52	421	60
13	561	80
6	579	82
16	661	94
3	566	80
40	572	81
20	643	91
48 	657	93

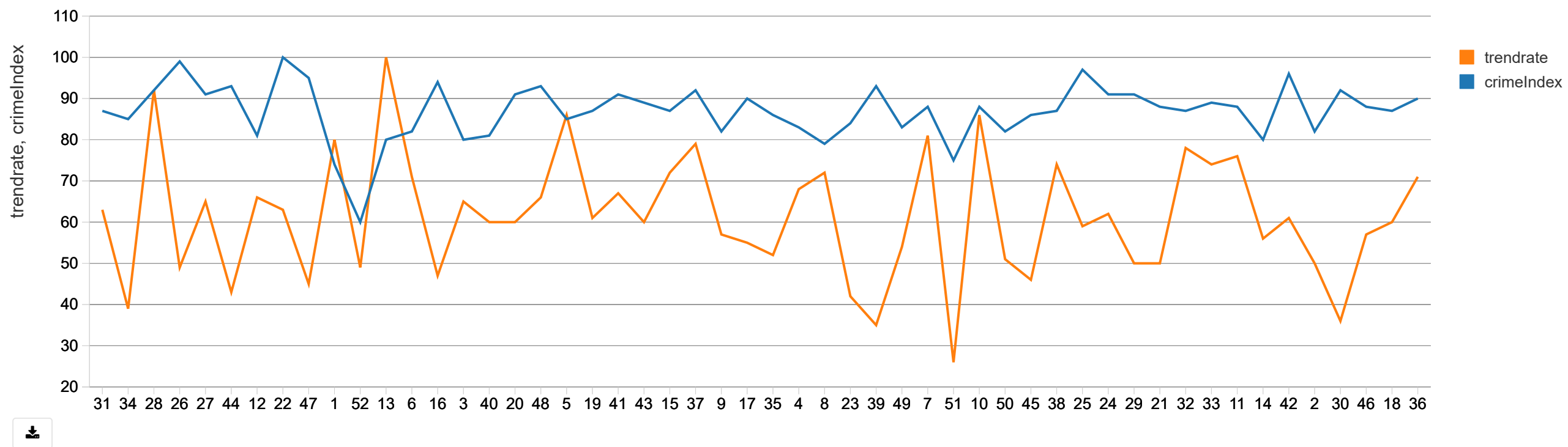
```
from pyspark.sql.functions import to_timestamp

rawdata_google = sqlContext.read.format("com.databricks.spark.csv").options(header='true').option("inferSchema", "true").load("/FileStore/tables/GoogleTrend_Toronto.csv")

# with the weekofyear, adding additional columns to get numerical week num
rawdata_google=rawdata_google.withColumn('weekNum',weekofyear(to_timestamp(rawdata_google.week,"dd/MM/yyyy"))) +1)
rawdata_google= rawdata_google.withColumn('weekNum',when(rtrim(rawdata_google.week) == '03/01/2016', 1).otherwise(rawdata_google.weekNum))
#display (rawdata_google.select('weekNum','trendrate'))

# Let's merge both dataset, by joining through the key weekNum.
merge_data =crime_week_year.join(rawdata_google,(rawdata_google.weekNum == crime_week_year.weekNum))

display(merge_data)
```



Conclusion : It seems like that is not much relation between google search and crime occurred. However, it can be seen that there was a rise in web search from week 2 to week12 and crime during that period was low. Week 16 to week 26, crime rate was at peak and google search was low.

From week 27 to week 48, It can be seen that google search is not steady, there are lots of ups and downs whereas crime rate became steady during that period.