This notebook explores Chicago Crime public dataset (bigquery-public-data.chicago_crime.crime)

```
query = """
SELECT count(arrest) FROM `bigquery-public-data.chicago_crime.crime` where arrest IS true
"""
```

```
# Call BigQuery and examine in dataframe
import google.datalab.bigquery as bq
df = bq.Query(query + " LIMIT 100").execute().result().to_dataframe()
```

```
print("There were " + str(df.at[0,"f0_"]) + " arrests in Chicago")
```

```
There were 1874936 arrests in Chicago
```

Chicago coordinates are: latitude 41.8781° N, longitude 87.6298° W

```
#example row
```

I create a table with ~ 1/5 of data : SELECT * FROM bigquery-public-data.chicago_crime.crime where MOD(unique_key, 5) = 0

```
query = """
SELECT * FROM `ml-sme-223918.bqml_tutorial_us.chicago_crime_subset`
"""
```

```
import google.datalab.bigquery as bq
df = bq.Query(query + " LIMIT 10000").execute().result().to_dataframe()
```

```
df.describe()
```

	unique_key	beat	district	ward	community_area	x_coordinate	y_coordi
count	1.000000e+04	10000.000000	10000.000000	9094.000000	9093.000000	9.880000e+03	9.880000e
mean	6.019879e+06	957.668400	8.809300	25.222784	37.324645	1.169884e+06	1.862125e
std	2.959119e+06	624.775001	5.129304	12.797437	17.621808	1.009025e+04	3.898737e
min	6.400000e+02	512.000000	5.000000	2.000000	3.000000	1.145015e+06	1.818775e
25%	3.354052e+06	522.000000	5.000000	9.000000	30.000000	1.162321e+06	1.828462e
50%	5.864905e+06	531.000000	5.000000	25.000000	49.000000	1.173140e+06	1.835508e
75%	8.407878e+06	1033.000000	10.000000	34.000000	53.000000	1.178061e+06	1.888487e
max	1.152740e+07	2323.000000	19.000000	48.000000	56.000000	1.188194e+06	1.932093e

I observe that latitude is between (41.658132, 41.969159) and longitude is between (-87.743523, -87.586439)

Also I see that year is between 2001 and 2018

```
df.head()
```

	unique_key	case_number	date	block	iucr	primary_type	description	location_descript
0	3045	HL177967	2005-02-12 20:47:00	007XX E 103RD ST	0110	HOMICIDE	FIRST DEGREE MURDER	RETAIL STORE
1	3205	HL435664	2005-06-21 21:28:00	103XX S INDIANA AVE	0110	HOMICIDE	FIRST DEGREE MURDER	STREET
2	20900	HW295447	2013-05-29 15:11:00	000XX W 107TH ST	0110	HOMICIDE	FIRST DEGREE MURDER	STREET
3	1710710	G513455	2001-08-27 23:55:00	104XX S STATE ST	0265	CRIM SEXUAL ASSAULT	AGGRAVATED: OTHER	RESIDENCE
4	11363170	JB327133	2018-06-29 00:44:13	002XX W 104TH ST	0281	CRIM SEXUAL ASSAULT	NON-AGGRAVATED	RESIDENCE

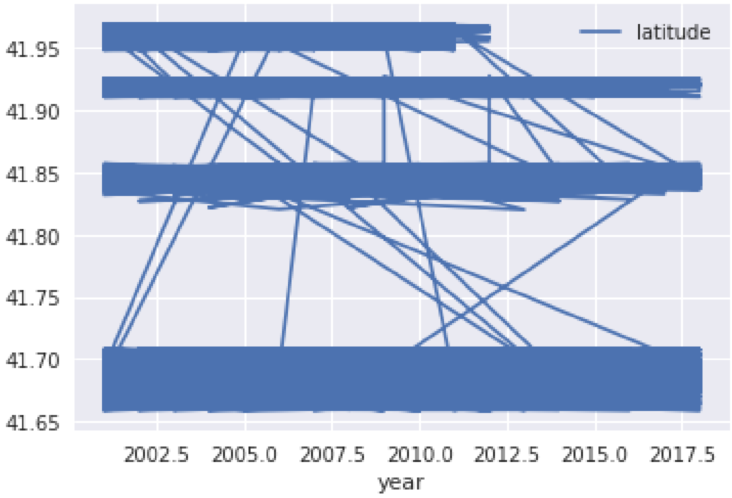
5 rows × 22 columns

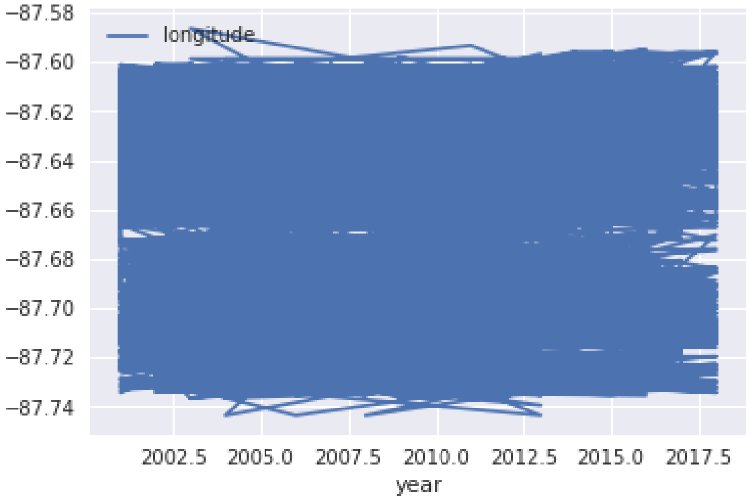
I see in BigQuery: Table size 271.74 MB

Number of rows 1,353,959

```
df.plot(x='year', y='latitude')
df.plot(x='year', y='longitude')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f8bfc245250>
```





I see a lot of crime activity happening between 41.66 : 41.71 latitude in Chicago.

```
print(df['location'][0])
```

```
(41.707455731, -87.605637491)
```

```
print(df['location'])
```

```
for x in df['location'][0]:
  print(x)
```

```
(
4
1
.
7
0
7
4
5
5
7
3
1
,
 
-
8
7
.
6
0
5
6
3
7
4
9
1
)
```

```
print(df['location'][0].find(","))
```

```
13
```

so it's a string

```
import copy
df2=copy.deepcopy(df)
```

https://docs.python.org/2/library/copy.html (https://docs.python.org/2/library/copy.html)

Now, considering first element is (41.707455731, -87.605637491)

```
print(df2['location'][0][8:13])
```

```
55731
```

```
print(len(df2['location'][0]))
```

```
29
```

```
print(df2['location'][0][23:28])
```

```
37491
```

```
df2['location'][0][8:13]="00000"
print(df2['location'][0])
```

```
TypeErrorTraceback (most recent call last)
<ipython-input-25-6dbf6f697f44> in <module>()
----> 1 df2['location'][0][8:13]="00000"
      2 print(df2['location'][0])

TypeError: 'newstr' object does not support item assignment
```

```
for index, row in df2.iterrows():
    print(row)
    print(row['location'])
    break
```

```
unique_key                                  3045
case_number                             HL177967
date                         2005-02-12 20:47:00
block                            007XX E 103RD ST
iucr                                        0110
primary_type                            HOMICIDE
description                 FIRST DEGREE MURDER
location_description                RETAIL STORE
arrest                                      True
domestic                                   False
beat                                         512
district                                       5
ward                                           9
community_area                                50
fbi_code                                     01A
x_coordinate                         1.18295e+06
y_coordinate                         1.83683e+06
year                                        2005
updated_on                   2015-08-17 15:03:40
latitude                                 41.7075
longitude                               -87.6056
location        (41.707455731, -87.605637491)
Name: 0, dtype: object
(41.707455731, -87.605637491)
```

```
for index, row in df2.iterrows():
    print(row['location'])
    tmp = row['location'][0:8] + "00000" + row['location'][13:23] + "00000)"
    print(tmp)
    print(row['location'])
    row['location'] = tmp
    print(row['location'])
    break
```

```
(41.707455731, -87.605637491)
(41.707400000, -87.605600000)
(41.707455731, -87.605637491)
(41.707400000, -87.605600000)
```

```
for index, row in df2.iterrows():
  try:
    tmp = row['location'][0:8] + "00000" + row['location'][13:23] + "00000)"
    row['location'] = tmp
  except TypeError:
    print(row)
print(df2.head())
```

So there are rows for which there is no location set. Need to clean it up.

```
(df2[df2["location"] != False]).head()
#df2.head()
```

```
NameErrorTraceback (most recent call last)
<ipython-input-1-74cf8e19ee0e> in <module>()
----> 1 (df2[df2["location"] != False]).head()
      2 #df2.head()

NameError: name 'df2' is not defined
```

```
#checking if there still are any rows with no location data set
for index, row in df2.iterrows():
  try:
    tmp = row['location'][0:8]
  except TypeError:
    print(row['location'])
```

No rows with empty coordinates left (good) but also no change in location (bad).

```
#df2.head()
df3=copy.deepcopy(df)
df3.head()
```

	unique_key	case_number	date	block	iucr	primary_type	description	location_descript
0	3045	HL177967	2005-02-12 20:47:00	007XX E 103RD ST	0110	HOMICIDE	FIRST DEGREE MURDER	RETAIL STORE
1	3205	HL435664	2005-06-21 21:28:00	103XX S INDIANA AVE	0110	HOMICIDE	FIRST DEGREE MURDER	STREET
2	20900	HW295447	2013-05-29 15:11:00	000XX W 107TH ST	0110	HOMICIDE	FIRST DEGREE MURDER	STREET
3	1710710	G513455	2001-08-27 23:55:00	104XX S STATE ST	0265	CRIM SEXUAL ASSAULT	AGGRAVATED: OTHER	RESIDENCE
4	11363170	JB327133	2018-06-29 00:44:13	002XX W 104TH ST	0281	CRIM SEXUAL ASSAULT	NON-AGGRAVATED	RESIDENCE

5 rows × 22 columns

```
print((df3[df3["location"] != False]).shape[0])
print((df3[df3["location"] == False]).shape[0])
print((df3[df3["location"].notnull()]).shape[0])
```

```
10000
0
9880
```

```
#this is how to filter rows with None in location
df3 = df3[df3["location"].notnull()]
print(df3.shape[0])
```

```
9880
```

```
#let's really change the location
for index, row in df3.iterrows():
  try:
    #print("index="+index)
    tmp = row['location'][0:7] + "000000" + row['location'][12:23] + "000000)"
    print("tmp="+tmp)
    df3.set_value(index, 'location', tmp)
    #break
  except TypeError:
    print("TypeError in:" + row)
#print(df3.head())
```

```
print(df3.head())
```

```
   unique_key case_number                date                 block  iucr  \
0        3045    HL177967 2005-02-12 20:47:00     007XX E 103RD ST  0110
1        3205    HL435664 2005-06-21 21:28:00  103XX S INDIANA AVE  0110
2       20900    HW295447 2013-05-29 15:11:00     000XX W 107TH ST  0110
3     1710710     G513455 2001-08-27 23:55:00     104XX S STATE ST  0265
4    11363170    JB327133 2018-06-29 00:44:13     002XX W 104TH ST  0281

          primary_type          description location_description  arrest  \
0             HOMICIDE  FIRST DEGREE MURDER         RETAIL STORE    True
1             HOMICIDE  FIRST DEGREE MURDER               STREET    True
2             HOMICIDE  FIRST DEGREE MURDER               STREET    True
3  CRIM SEXUAL ASSAULT    AGGRAVATED: OTHER            RESIDENCE   False
4  CRIM SEXUAL ASSAULT       NON-AGGRAVATED            RESIDENCE   False

   domestic              ...                ward  community_area  fbi_code  \
0     False              ...                 9.0            50.0       01A
1     False              ...                 9.0            49.0       01A
2     False              ...                34.0            49.0       01A
3     False              ...                 NaN             NaN        02
4     False              ...                34.0            49.0        02

   x_coordinate y_coordinate  year          updated_on   latitude  longitude  \
0     1182951.0    1836828.0  2005 2015-08-17 15:03:40  41.707456 -87.605637
1     1179414.0    1836239.0  2005 2015-08-17 15:03:40  41.705921 -87.618608
2     1177693.0    1834013.0  2013 2015-08-17 15:03:40  41.699851 -87.624977
3     1178136.0    1835744.0  2001 2015-08-17 15:03:40  41.704591 -87.623303
4     1176571.0    1835979.0  2018 2018-07-06 15:55:18  41.705272 -87.629026

                        location
0  (41.7070000000, -87.6056000000)
1  (41.7050000000, -87.6186000000)
2  (41.6990000000, -87.6249000000)
3  (41.7040000000 -87.62330000000)
4  (41.7050000000, -87.6290000000)

[5 rows x 22 columns]
```

```
#let's plot the crime area
#first, sum up crime # in same location
df4 = df3.groupby('location').count()
df4.head()
#df4.plot(x='location', y='count', logy=True, kind='bar');
```

	unique_key	case_number	date	block	iucr	primary_type	description	location_descrip
location								
(41.6580000000, -87.6340000000)	2	2	2	2	2	2	2	2
(41.6580000000, -87.6357000000)	1	1	1	1	1	1	1	1
(41.6580000000, -87.6380000000)	1	1	1	1	1	1	1	1
(41.6580000000, -87.6393000000)	1	1	1	1	1	1	1	1
(41.6580000000, -87.6404000000)	3	3	3	3	3	3	3	3

5 rows × 21 columns

```
df5=df4.sort_values(by='case_number', ascending=False)
df5.head()
```

	unique_key	case_number	date	block	iucr	primary_type	description	location_descrip
location								
(41.7050000000, -87.6009000000)	40	40	40	40	40	40	40	40
(41.6920000000, -87.6043000000)	39	39	39	39	39	39	39	39
(41.7070000000, -87.6018000000)	39	39	39	39	39	39	39	39
(41.9640000000, -87.6547000000)	33	33	33	33	33	33	33	33
(41.8490000000, -87.7088000000)	33	33	33	33	33	33	33	33

5 rows × 21 columns

```
df5.plot(x='location', y='case_number', kind='bar')
```

```
KeyErrorTraceback (most recent call last)
<ipython-input-39-58b3f5d49b01> in <module>()
----> 1 df5.plot(x='location', y='case_number', kind='bar')

/usr/local/envs/py2env/lib/python2.7/site-packages/pandas/plotting/_core.pyc in __call__(self, x, y, kin
d, ax, subplots, sharex, sharey, layout, figsize, use_index, title, grid, legend, style, logx, logy, loglog, xt
icks, yticks, xlim, ylim, rot, fontsize, colormap, table, yerr, xerr, secondary_y, sort_columns, **kwds)
   2675                         fontsize=fontsize, colormap=colormap, table=table,
   2676                         yerr=yerr, xerr=xerr, secondary_y=secondary_y,
-> 2677                         sort_columns=sort_columns, **kwds)
   2678     __call__.__doc__ = plot_frame.__doc__
   2679 

/usr/local/envs/py2env/lib/python2.7/site-packages/pandas/plotting/_core.pyc in plot_frame(data, x, y,
 kind, ax, subplots, sharex, sharey, layout, figsize, use_index, title, grid, legend, style, logx, logy, loglo
g, xticks, yticks, xlim, ylim, rot, fontsize, colormap, table, yerr, xerr, secondary_y, sort_columns, **kw
ds)
   1900                  yerr=yerr, xerr=xerr,
   1901                  secondary_y=secondary_y, sort_columns=sort_columns,
-> 1902                  **kwds)
   1903 
   1904 

/usr/local/envs/py2env/lib/python2.7/site-packages/pandas/plotting/_core.pyc in _plot(data, x, y, subpl
ots, ax, kind, **kwds)
   1707             if is_integer(x) and not data.columns.holds_integer():
   1708                 x = data.columns[x]
-> 1709             data = data.set_index(x)
   1710 
   1711         if y is not None:

/usr/local/envs/py2env/lib/python2.7/site-packages/pandas/core/frame.pyc in set_index(self, keys, dro
p, append, inplace, verify_integrity)
   3144                 names.append(None)
   3145             else:
-> 3146                 level = frame[col]._values
   3147                 names.append(col)
   3148                 if drop:

/usr/local/envs/py2env/lib/python2.7/site-packages/pandas/core/frame.pyc in __getitem__(self, key)
   2137             return self._getitem_multilevel(key)
   2138         else:
-> 2139             return self._getitem_column(key)
   2140 
   2141     def _getitem_column(self, key):

/usr/local/envs/py2env/lib/python2.7/site-packages/pandas/core/frame.pyc in _getitem_column(self, ke
y)
```

```
   2144         # get column
   2145         if self.columns.is_unique:
-> 2146             return self._get_item_cache(key)
   2147 
   2148         # duplicate columns & possible reduce dimensionality

/usr/local/envs/py2env/lib/python2.7/site-packages/pandas/core/generic.pyc in _get_item_cache(self, item)
   1840         res = cache.get(item)
   1841         if res is None:
-> 1842             values = self._data.get(item)
   1843             res = self._box_item_values(item, values)
   1844             cache[item] = res

/usr/local/envs/py2env/lib/python2.7/site-packages/pandas/core/internals.pyc in get(self, item, fastpath)
   3841 
   3842             if not isna(item):
-> 3843                 loc = self.items.get_loc(item)
   3844             else:
   3845                 indexer = np.arange(len(self.items))[isna(self.items)]

/usr/local/envs/py2env/lib/python2.7/site-packages/pandas/core/indexes/base.pyc in get_loc(self, key, method, tolerance)
   2525                 return self._engine.get_loc(key)
   2526             except KeyError:
-> 2527                 return self._engine.get_loc(self._maybe_cast_indexer(key))
   2528 
   2529         indexer = self.get_indexer([key], method=method, tolerance=tolerance)

pandas/_libs/index.pyx in pandas._libs.index.IndexEngine.get_loc()

pandas/_libs/index.pyx in pandas._libs.index.IndexEngine.get_loc()

pandas/_libs/hashtable_class_helper.pxi in pandas._libs.hashtable.PyObjectHashTable.get_item()

pandas/_libs/hashtable_class_helper.pxi in pandas._libs.hashtable.PyObjectHashTable.get_item()

KeyError: 'location'
```

The above is because index was set to location , need to be reset https://stackoverflow.com/questions/31167896/keyerror-in-dataframe (https://stackoverflow.com/questions/31167896/keyerror-in-dataframe)

```
df5 = df5.reset_index()
df5.head(1)
```

	index	location	unique_key	case_number	date	block	iucr	primary_type	description	locatic
0	0	(41.7050000000, -87.6009000000)	40	40	40	40	40	40	40	40

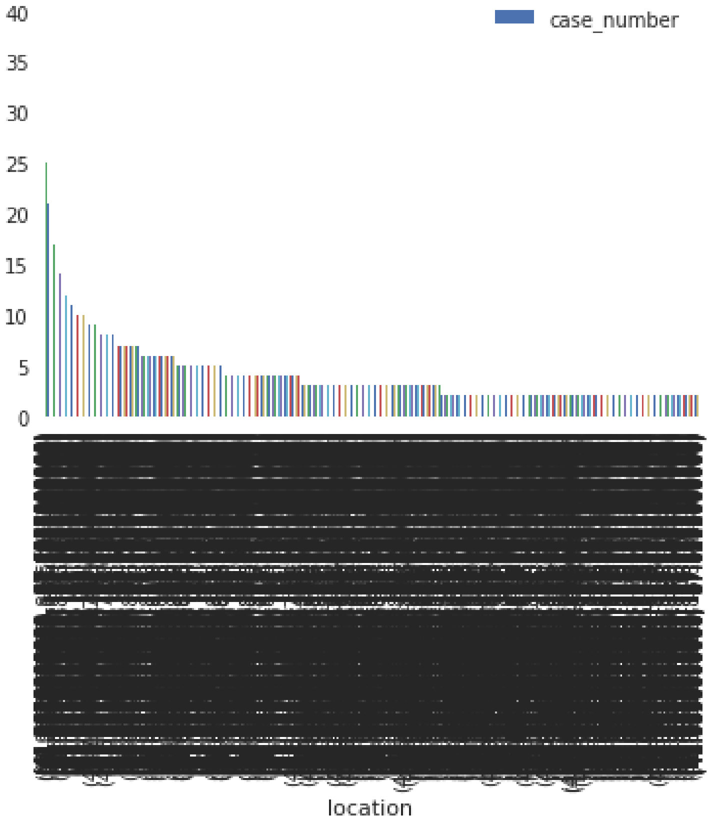
1 rows × 23 columns

```
print(df5.shape[0])
df5 = df5[df5['case_number']>1]
print(df5.shape[0])
```

```
4174
1784
```

```
df5.plot(x='location', y='case_number', kind='bar')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f9538462f90>
```



Cant' see much from the chart.
Anyway, this is the area with most crimes: https://goo.gl/maps/sG6bqFV9Xcm (https://goo.gl/maps/sG6bqFV9Xcm)
in my dataframe (**not** in Chicago - since I only took 10,000 rows from the > 1 M rows)
(after removing 6xzeros from both latitude and longitude)

http://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.drop_duplicates.html (http://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.drop_duplicates.html)

```
#df3 has the rows that have location set
print(df3.shape[0])
df31=df3.drop_duplicates(subset="primary_type")
print(df31.shape[0])
```

```
9880
25
```

```
print(df31["primary_type"])
```

```
0                          HOMICIDE
3                CRIM SEXUAL ASSAULT
7                           ROBBERY
43                          BATTERY
174           PUBLIC PEACE VIOLATION
289                         ASSAULT
375                        STALKING
376                        BURGLARY
441                           THEFT
576              MOTOR VEHICLE THEFT
625                           ARSON
626               DECEPTIVE PRACTICE
655                  CRIMINAL DAMAGE
755                CRIMINAL TRESPASS
770                WEAPONS VIOLATION
786                     PROSTITUTION
803                      SEX OFFENSE
805                         GAMBLING
806        OFFENSE INVOLVING CHILDREN
816                       KIDNAPPING
817                        NARCOTICS
935              LIQUOR LAW VIOLATION
936                    OTHER OFFENSE
975    INTERFERENCE WITH PUBLIC OFFICER
1614                    INTIMIDATION
Name: primary_type, dtype: object
```

```
print(df31.head(1))
```

```
   unique_key case_number                date            block  iucr  \
0        3045    HL177967 2005-02-12 20:47:00  007XX E 103RD ST  0110

  primary_type          description location_description  arrest  domestic  \
0     HOMICIDE  FIRST DEGREE MURDER         RETAIL STORE    True     False

               ...                ward  community_area  fbi_code  \
0              ...                 9.0            50.0       01A

   x_coordinate y_coordinate  year          updated_on   latitude  longitude  \
0     1182951.0    1836828.0  2005 2015-08-17 15:03:40  41.707456 -87.605637

                        location
0  (41.7070000000, -87.6056000000)

[1 rows x 22 columns]
```

So things of interest: primary_type ; location_description ; arrest ; domestic ; year ; location

```
df32=df3.drop_duplicates(subset="location_description")
print(df32.shape[0])
print(df32["location_description"])
```

```
81
0                  RETAIL STORE
1                     STREET
3                   RESIDENCE
5           VEHICLE NON-COMMERCIAL
7                 HOTEL/MOTEL
8                   SIDEWALK
10                GAS STATION
13     PARKING LOT/GARAGE(NON.RESID.)
15             RESIDENCE-GARAGE
19                   TAXICAB
32             SMALL RETAIL STORE
37          SCHOOL, PUBLIC, BUILDING
44          SCHOOL, PUBLIC, GROUNDS
45         RESIDENCE PORCH/HALLWAY
53                  APARTMENT
70                    OTHER
84             VEHICLE-COMMERCIAL
88                   CTA BUS
120                   ALLEY
136                RESTAURANT
216      RESIDENTIAL YARD (FRONT/BACK)
297     POLICE FACILITY/VEH PARKING LOT
323           GROCERY FOOD STORE
327          TAVERN/LIQUOR STORE
352        CHA PARKING LOT/GROUNDS
388           CONSTRUCTION SITE
392            VACANT LOT/LAND
418              CHA APARTMENT
455               DRUG STORE
460           ABANDONED BUILDING
                 ...
2517                 HOUSE
2826     FACTORY/MANUFACTURING BUILDING
2915                CAR WASH
2994   OTHER RAILROAD PROP / TRAIN DEPOT
2996        SCHOOL, PRIVATE, BUILDING
3116                  AUTO
3148       COLLEGE/UNIVERSITY GROUNDS
3161      NURSING HOME/RETIREMENT HOME
3194     OTHER COMMERCIAL TRANSPORTATION
3427       CTA GARAGE / OTHER PROPERTY
3665            FEDERAL BUILDING
3907        HOSPITAL BUILDING/GROUNDS
3955          MEDICAL/DENTAL OFFICE
3979             CLEANING STORE
4178         JAIL / LOCK-UP FACILITY
4330              FIRE STATION
4759            APPLIANCE STORE
```

```
4824       CHA HALLWAY/STAIRWELL/ELEVATOR
4899                       VACANT LOT
5556                    DAY CARE CENTER
5880                     LAUNDRY ROOM
6101                   BOAT/WATERCRAFT
6161                    ATHLETIC CLUB
6349             SCHOOL, PRIVATE, GROUNDS
7253                    BOWLING ALLEY
7389                   ANIMAL HOSPITAL
8223                          YARD
8948                MOVIE HOUSE/THEATER
9027                          None
9151    COLLEGE/UNIVERSITY RESIDENCE HALL
Name: location_description, Length: 81, dtype: object
```

We could first test a simple ML model: given primary_type, location_description => predict arrest (Y/N).