Oakland Crime Statistics 2011 to 2016

数据预处理

首先对数据进行预处理，将时间的单词

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import seaborn as sns
import json
import ast
import datetime as dt
from dateutil import tz
from dateutil import parser
import matplotlib.pyplot as plt
import time

# Input data files are available in the "../input/" directory.
# For example, running this (by clicking run or pressing Shift+Enter) will list the files in the input directory

import os
print('\n'.join(os.listdir("./oakland-crime-statistics-2011-to-2016/")))
```

```
socrata_metadata_records-for-2011.json
socrata_metadata_records-for-2016.json
socrata_metadata_records-for-2014.json
socrata_metadata_records-for-2015.json
socrata_metadata_records-for-2012.json
records-for-2016.csv
records-for-2014.csv
records-for-2015.csv
records-for-2011.csv
records-for-2012.csv
socrata_metadata_records-for-2013.json
records-for-2013.csv
```

```
intervals = (
    ('weeks', 604800),  # 60 * 60 * 24 * 7
    ('days', 86400),    # 60 * 60 * 24
    ('hours', 3600),    # 60 * 60
    ('minutes', 60),
    ('seconds', 1),
    )


def get_nlargest_incident_id(n, df):
    return df.groupby(by="Incident Type Id",sort=True, as_index=False).count().nlargest(n, 'Create Time')["Incident Type Id"].values
def get_nlargest_area_id(n, df):
    return df.groupby(by="Area Id",sort=True, as_index=False).count().nlargest(n, 'Create Time')["Area Id"].values
def display_time(seconds, granularity=10):
    result = []

    for name, count in intervals:
        value = seconds // count
        if value:
            seconds -= value * count
            if value == 1:
                name = name.rstrip('s')
            result.append("{} {}".format(value, name))
    return ', '.join(result[:granularity])

def map_x(x):
    if x.hour < 6:
        return "00AM-6AM"
    if x.hour < 12 and x.hour > 6:
        return "6AM-12PM"
    if x.hour >= 12 and x.hour<18:
        return "12PM-6PM"
    if x.hour > 18:
        return "6PM-00AM"

def prep_data(df):
    df['Create Time'] = df['Create Time'].fillna(df['Closed Time'])
    df['Closed Time'] = df['Closed Time'].fillna(df['Create Time'])
    df["time_between_creation_and_closed_seconds"] = df.apply(lambda x: abs((parser.parse(x["Closed Time"]) - parser.parse(x["Create
    df["time_of_the_day"] = df["Create Time"].map(lambda x:map_x(parser.parse(x)))
    df.replace(r'', np.nan, regex=True, inplace=True)
    df["Area Id"].fillna(-1, inplace=True)
    df["Beat"].fillna("Unknown", inplace=True)
    df["Priority"].fillna("-1", inplace=True)
    df["Priority"].astype(int)
    df.drop(["Agency", "Event Number"], inplace=True, axis=1)
    df["day_of_the_month"] = df["Create Time"].apply(lambda x: parser.parse(x).day)
    df["day_of_the_week"] = df["day_of_the_month"].apply(lambda x: (x % 7) + 1)
    df["month_of_the_year"] = df["Create Time"].apply(lambda x: parser.parse(x).month)
    return df
```

```
# 五数分布
```

```
def fiveNumber(nums):
        # 五数概括 Minimum（最小值）、Q1、Median（中位数、）、Q3、Maximum（最大值）
    Minimum = min(nums)
    Maximum = max(nums)
    Q1 = np.percentile(nums, 25)
    Median = np.median(nums)
    Q3 = np.percentile(nums, 75)

    IQR = Q3-Q1
    lower_limit = Q1-1.5*IQR  # 下限值
    upper_limit = Q3+1.5*IQR  # 上限值

    return Minimum, Q1, Median, Q3, Maximum, lower_limit, upper_limit


def printfiveNumber(fivenumber):
    print("+++++++++++++++")
    print(f"Min = {fivenumber[0]}")
    print(f"Q1  = {fivenumber[1]}")
    print(f"Median = {fivenumber[2]}")
    print(f"Q3 = {fivenumber[3]}")
    print(f"Max = {fivenumber[4]}")
    print(f"lower_limit = {fivenumber[5]}")
    print(f"upper_limit = {fivenumber[6]}")
    print("+++++++++++++++")
```

导入数据

- 2011年数据

```
crimes_2011 = pd.read_csv("./oakland-crime-statistics-2011-to-2016/records-for-2011.csv")
crimes_2011.drop(index=[180015], inplace=True)
crimes_2011 = prep_data(crimes_2011)
crimes_2011.rename(index=str, columns={"Location": "address"}, inplace=True)
crimes_2011["Priority"].replace(0.0, 1.0, inplace=True)
crimes_2011["Priority"] = crimes_2011["Priority"].astype(int)
crimes_2011.head(2)
```

	Create Time	address	Area Id	Beat	Priority	Incident Type Id	Incident Type Description	Closed Time	time_between_creation_and_
0	2011-01-01T00:00:00.000	ST&SAN PABLO AV	1.0	06X	1	PDOA	POSSIBLE DEAD PERSON	2011-01-01T00:28:17.000	1697
1	2011-01-01T00:01:11.000	ST&HANNAH ST	1.0	07X	1	415GS	415 GUNSHOTS	2011-01-01T01:12:56.000	4305

- 2012年数据

```
crimes_2012 = pd.read_csv("./oakland-crime-statistics-2011-to-2016/records-for-2012.csv")
crimes_2012.dropna(thresh=9, inplace=True)
crimes_2012 = prep_data(crimes_2012)
crimes_2012.rename(index=str, columns={"Location ": "address"}, inplace=True)
crimes_2012["Area Id"] = crimes_2013["Area Id"].astype(int)
crimes_2012["Priority"].replace(0.0, 1.0, inplace=True)
crimes_2012["Priority"] = crimes_2013["Priority"].astype(int)
crimes_2012.head(2)
```

	Create Time	Area Id	Beat	Priority	Incident Type Id	Incident Type Description	Closed Time	Location 1	Zip Codes	time_betweer

	Create Time	Area Id	Beat	Priority	Incident Type Id	Incident Type Description	Closed Time	Location 1	Zip Codes	time_betweer
0	2012-01-01T00:00:25.000	2.0	32Y	1.0	415GS	415 GUNSHOTS	2012-01-01T00:40:27.000	{'human_address': '{"address": "OLIVE ST", "ci...	NaN	2402
1	2012-01-01T00:00:27.000	2.0	30Y	2.0	415GS	415 GUNSHOTS	2012-01-01T01:34:31.000	{'human_address': '{"address": "AV&MACARTHUR B...	NaN	5644

- 2013年数据

```
crimes_2013 = pd.read_csv("./oakland-crime-statistics-2011-to-2016/records-for-2013.csv")
crimes_2013.dropna(thresh=9, inplace=True)
crimes_2013 = prep_data(crimes_2013)
crimes_2013.rename(index=str, columns={"Location ": "address"}, inplace=True)
crimes_2013["Area Id"] = crimes_2013["Area Id"].astype(int)
crimes_2013["Priority"].replace(0.0, 1.0, inplace=True)
crimes_2013["Priority"] = crimes_2013["Priority"].astype(int)
crimes_2013.head(2)
```

	Create Time	address	Area Id	Beat	Priority	Incident Type Id	Incident Type Description	Closed Time	time_between_creation_and_clos
0	2013-01-01T00:00:00.000	D ST	2	33X	1	415GS	415 GUNSHOTS	2013-01-01T00:47:51.000	2871
1	2013-01-01T00:00:05.000	ARTHUR ST	2	30X	2	415GS	415 GUNSHOTS	2013-01-01T01:30:58.000	5453

- 2014年数据

```
crimes_2014 = pd.read_csv("./oakland-crime-statistics-2011-to-2016/records-for-2014.csv")
crimes_2014.dropna(thresh=9, inplace=True)
crimes_2014 = prep_data(crimes_2014)
crimes_2014.rename(index=str, columns={"Location ": "address"}, inplace=True)
crimes_2014["Area Id"] = crimes_2013["Area Id"].astype(int)
crimes_2014["Priority"].replace(0.0, 1.0, inplace=True)
crimes_2014["Priority"] = crimes_2013["Priority"].astype(int)
crimes_2014.head(2)
```

	Create Time	Area Id	Beat	Priority	Incident Type Id	Incident Type Description	Closed Time	Location 1	Zip Codes	time_betwe
0	2014-01-01T00:00:00.000	2.0	02X	1.0	415GS	415 GUNSHOTS	2014-01-01T03:22:08.000	{'human_address': '{"address": "LINDEN ST", "c...	NaN	12128
1	2014-01-01T00:00:00.000	2.0	26Y	2.0	415GS	415 GUNSHOTS	2014-01-01T02:56:31.000	{'human_address': '{"address": "AV&INTERNATION...	NaN	10591

- 2015年数据

```
crimes_2015 = pd.read_csv("./oakland-crime-statistics-2011-to-2016/records-for-2015.csv")
crimes_2015.dropna(thresh=9, inplace=True)
crimes_2015 = prep_data(crimes_2015)
crimes_2015.rename(index=str, columns={"Location ": "address"}, inplace=True)
crimes_2015["Area Id"] = crimes_2013["Area Id"].astype(int)
crimes_2015["Priority"].replace(0.0, 1.0, inplace=True)
crimes_2015["Priority"] = crimes_2013["Priority"].astype(int)
crimes_2015.head(2)
```

	Create Time	Location	Area Id	Beat	Priority	Incident Type Id	Incident Type Description	Closed Time	time_between_creation_and_
0	2015-01-01T00:01:59.000	S ELMHURST AV	2.0	31Y	1.0	415	DISTURBING THE PEACE	2015-01-01T06:23:08.000	22869
1	2015-01-01T00:02:02.000	AV&D ST	2.0	32X	2.0	415GS	415 GUNSHOTS	2015-01-01T01:44:40.000	6158

- 2016年数据

```
crimes_2016 = pd.read_csv("./oakland-crime-statistics-2011-to-2016/records-for-2016.csv")
crimes_2016.dropna(thresh=9, inplace=True)
crimes_2016 = prep_data(crimes_2016)
crimes_2016.rename(index=str, columns={"Location": "address"}, inplace=True)
crimes_2016["Priority"] = crimes_2016["Priority"].astype(int)
crimes_2016.head(2)
```

	Create Time	address	Area Id	Beat	Priority	Incident Type Id	Incident Type Description	Closed Time	time_between_creation_and
0	2016-01-01T00:00:57.000	ST&MARKET ST	P1	05X	2	415GS	415 GUNSHOTS	2016-01-01T00:32:30.000	1893
1	2016-01-01T00:01:25.000	AV&HAMILTON ST	P3	26Y	2	415GS	415 GUNSHOTS	2016-01-01T00:48:23.000	2818

数据可视化和摘要

犯罪持续时间五数分布

```
print("五数分析")
for i, crime_year in enumerate(crimes_list):
    five = fiveNumber(crime_year[crime_year["Priority"] == 1]["time_between_creation_and_closed_seconds"])
    print(str(2011+i)+"time_between_creation_and_closed_seconds")
    printfiveNumber(five)
```

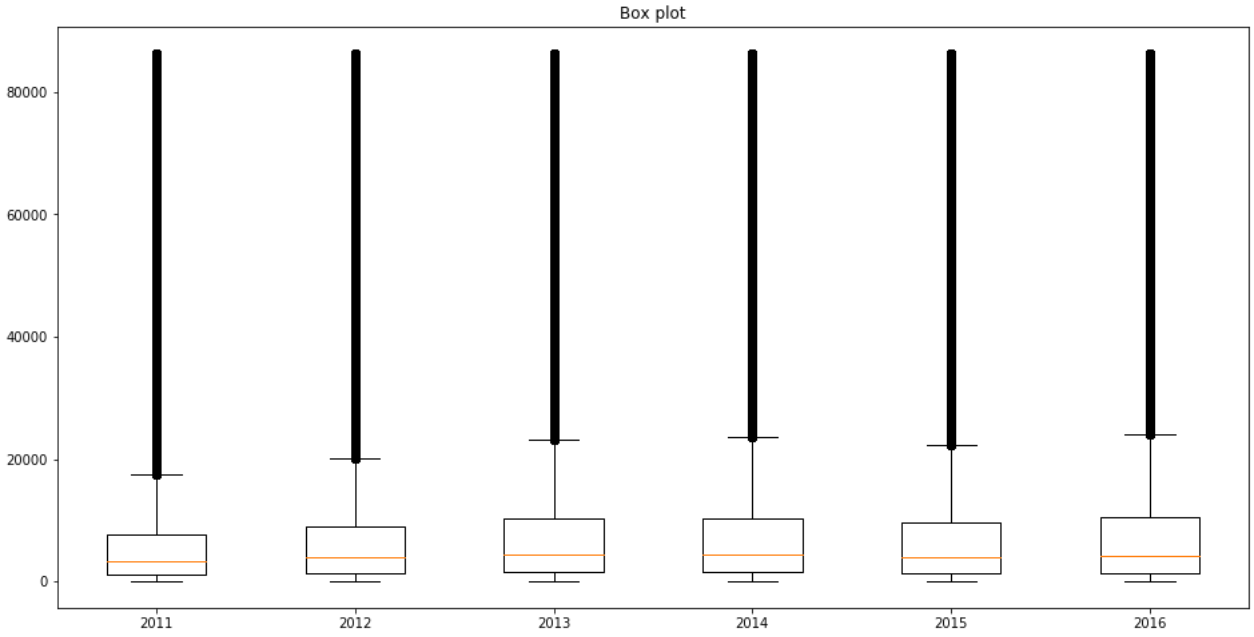
五数分析
2011time_between_creation_and_closed_seconds
+++++++++++++++
Min = 0
Q1 = 1379.0
Median = 3393.0
Q3 = 7047.0
Max = 86241
lower_limit = -7123.0
upper_limit = 15549.0
+++++++++++++++
2012time_between_creation_and_closed_seconds
+++++++++++++++
Min = 0
Q1 = 1396.25
Median = 3890.0
Q3 = 8941.0
Max = 86343
lower_limit = -9920.875
upper_limit = 20258.125
+++++++++++++++
2013time_between_creation_and_closed_seconds
+++++++++++++++
Min = 0
Q1 = 1635.0
Median = 4024.0
Q3 = 8136.0
Max = 86388
lower_limit = -8116.5
upper_limit = 17887.5
+++++++++++++++
2014time_between_creation_and_closed_seconds
+++++++++++++++
Min = 0
Q1 = 1500.0
Median = 4304.0
Q3 = 10436.0
Max = 86399
lower_limit = -11904.0
upper_limit = 23840.0
+++++++++++++++
2015time_between_creation_and_closed_seconds
+++++++++++++++
Min = 0
Q1 = 1313.0
Median = 3976.5
Q3 = 9604.25
Max = 86399
lower_limit = -11123.875
upper_limit = 22041.125
+++++++++++++++
2016time_between_creation_and_closed_seconds
+++++++++++++++
Min = 0
Q1 = 1541.5
Median = 3830.0
Q3 = 8049.0
Max = 86399
lower_limit = -8219.75
upper_limit = 17810.25
+++++++++++++++

犯罪处理时间随时间的变化

```
def box_plot(all_data):
#     all_data = np.array(all_data)
    fig = plt.figure(figsize=(16,8))

    plt.boxplot(all_data,notch=False, sym='o',vert=True)   # vertical box aligmnent  # vertical box aligmnent
    year_list = []
    for i in range(2011,2017):
        year_list.append(str(i))
    plt.xticks([i for i in range(1,7)],year_list)
#     plt.xlabel('measurement x')
    t = plt.title('Box plot')
    plt.show()


box = []
for crime_year in crimes_list:
    plt.subplots_adjust(left=0, right=2.5, top=3, bottom=1)
    box.append(crime_year["time_between_creation_and_closed_seconds"])
# box = box.transpose()
print(np.shape(box))
#         plt.boxplot(box,notch=False, sym='o',vert=True)   # vertical box aligmnent  # vertical box aligmnent
#         plot.show()

box_plot(box)
```

```
(6,)
```

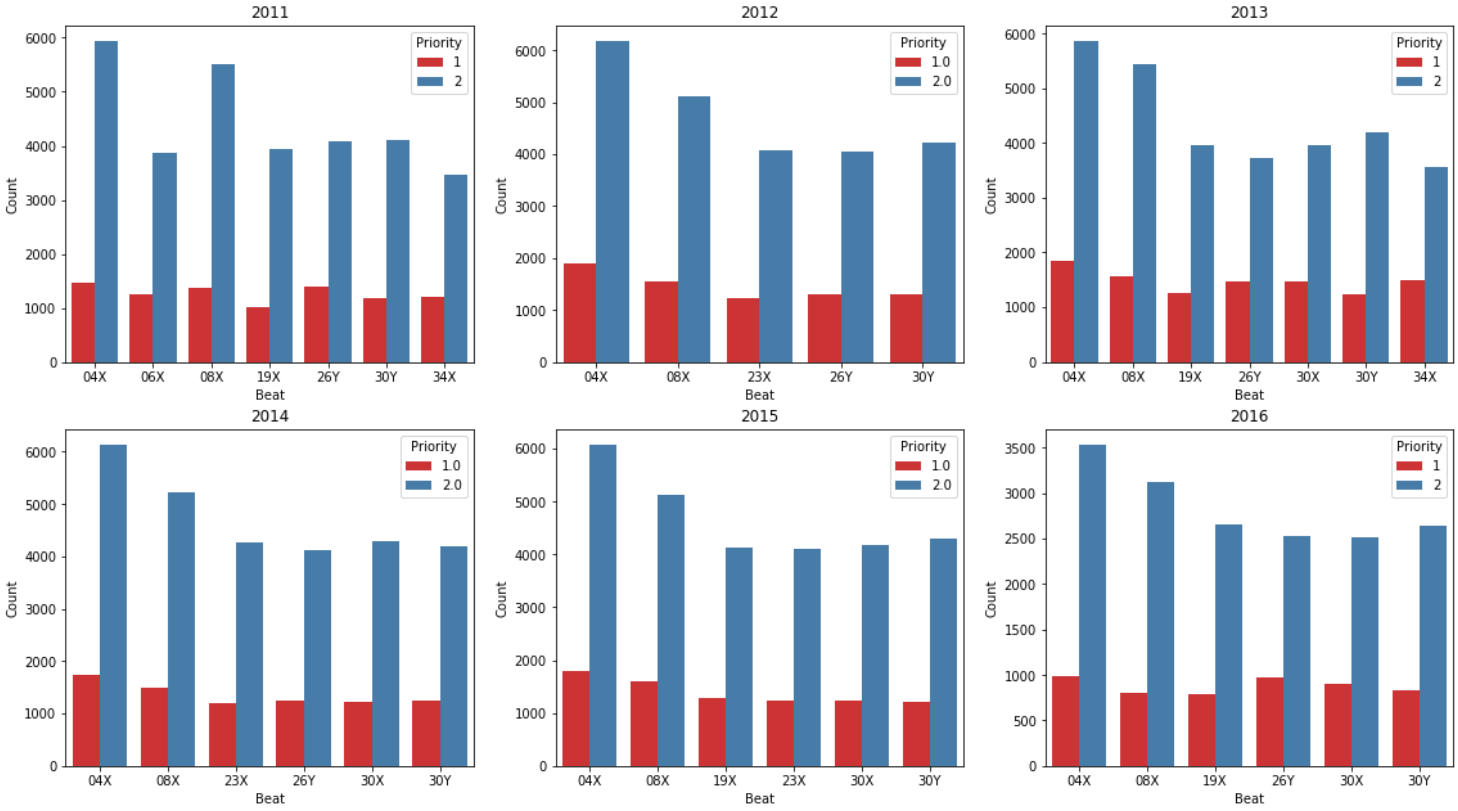
```
<Figure size 432x288 with 0 Axes>
```



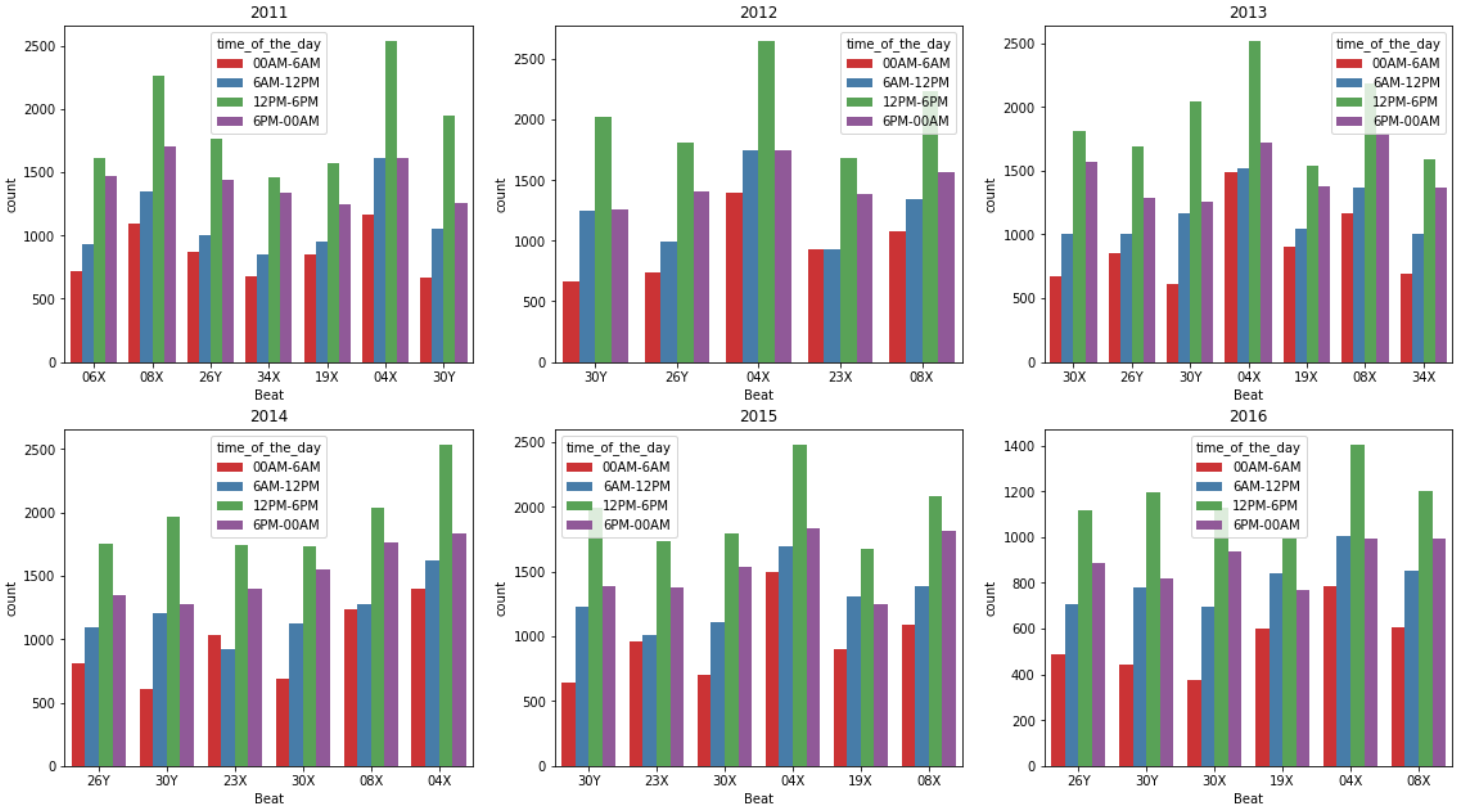
beats的频数统计

```
fig, ax = plt.subplots(nrows=2, ncols=3)
plt.subplots_adjust(left=0, right=2.5, top=3, bottom=1)
i = 0
for row in ax:
    for col in row:
        col.set_title(str(2011 + i))
        #sns.countplot(data=crimes_list[i].loc[crimes_list[i]['Incident Type Id'].isin(nlargest[i])], x="Incident Type Id", hue="Pric
        temp = crimes_list[i].groupby(by=["Beat", "Priority"],sort=True, as_index=False).count().rename(index=str, columns={"Create T
        beats_prio_1 = list(temp[temp["Priority"] == 1].nlargest(5, "Count")["Beat"].values)
        beats_prio_2 = list(temp[temp["Priority"] == 2].nlargest(5, "Count")["Beat"].values)
        print("Year " + str(2011 +i ) +":\n")
        print("The Beats With the Most Reports (Priority 1, Decending Order): {} \nThe Beats With the Most Reports (Priority 2, Decer
        print("Common Beats: {}".format(str(list(set(beats_prio_1) & set(beats_prio_2)))))
        sns.barplot(data=temp[temp["Beat"].isin(beats_prio_1 + beats_prio_2)], x="Beat", y="Count", hue="Priority",palette="Set1", ax
        print("==========================================================================================\n")
        i += 1
```

```
Year 2011:

The Beats With the Most Reports (Priority 1, Decending Order): ['04X', '26Y', '08X', '06X', '34X']
The Beats With the Most Reports (Priority 2, Decending Order): ['04X', '08X', '30Y', '26Y', '19X']
Unique Beats: ['04X', '26Y', '30Y', '19X', '06X', '34X', '08X']
Common Beats: ['04X', '08X', '26Y']
==========================================================================================

Year 2012:

The Beats With the Most Reports (Priority 1, Decending Order): ['04X', '08X', '26Y', '30Y', '23X']
The Beats With the Most Reports (Priority 2, Decending Order): ['04X', '08X', '30Y', '23X', '26Y']
Unique Beats: ['04X', '26Y', '30Y', '23X', '08X']
Common Beats: ['04X', '30Y', '26Y', '23X', '08X']
==========================================================================================

Year 2013:

The Beats With the Most Reports (Priority 1, Decending Order): ['04X', '08X', '34X', '26Y', '30X']
The Beats With the Most Reports (Priority 2, Decending Order): ['04X', '08X', '30Y', '30X', '19X']
Unique Beats: ['04X', '30X', '26Y', '30Y', '19X', '34X', '08X']
Common Beats: ['04X', '30X', '08X']
==========================================================================================

Year 2014:

The Beats With the Most Reports (Priority 1, Decending Order): ['04X', '08X', '30Y', '26Y', '30X']
The Beats With the Most Reports (Priority 2, Decending Order): ['04X', '08X', '30X', '23X', '30Y']
Unique Beats: ['04X', '30X', '30Y', '26Y', '23X', '08X']
Common Beats: ['04X', '30X', '08X', '30Y']
==========================================================================================

Year 2015:

The Beats With the Most Reports (Priority 1, Decending Order): ['04X', '08X', '19X', '23X', '30X']
The Beats With the Most Reports (Priority 2, Decending Order): ['04X', '08X', '30Y', '30X', '19X']
Unique Beats: ['04X', '30X', '30Y', '19X', '23X', '08X']
Common Beats: ['04X', '30X', '08X', '19X']
==========================================================================================

Year 2016:

The Beats With the Most Reports (Priority 1, Decending Order): ['04X', '26Y', '30X', '30Y', '08X']
The Beats With the Most Reports (Priority 2, Decending Order): ['04X', '08X', '19X', '30Y', '26Y']
Unique Beats: ['04X', '30X', '26Y', '30Y', '19X', '08X']
Common Beats: ['04X', '08X', '30Y', '26Y']
==========================================================================================
```




```
fig, ax = plt.subplots(nrows=2, ncols=3)
plt.subplots_adjust(left=0, right=2.5, top=3, bottom=1)
i = 0
for row in ax:
    for col in row:
        col.set_title(str(2011 + i))
        temp = crimes_list[i].groupby(by=["Beat", "Priority"],sort=True, as_index=False).count().rename(index=str, columns={"Create T
        beats_prio_1 = list(temp[temp["Priority"] == 1].nlargest(5, "Count")["Beat"].values)
        beats_prio_2 = list(temp[temp["Priority"] == 2].nlargest(5, "Count")["Beat"].values)
        sns.countplot(data=crimes_list[i][crimes_list[i]["Beat"].isin(beats_prio_1 + beats_prio_2)], x="Beat", hue="time_of_the_day",
        i += 1
```



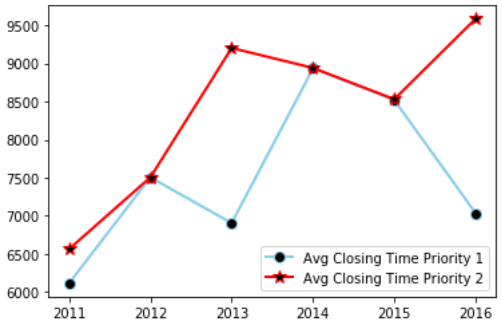

```
for i, x in enumerate(crimes_list):
    x["Year"] = 2011 + i
combined = crimes_2011
for x in range(1,len(crimes_list)):
    combined = combined.append(crimes_list[x], ignore_index=True)
combined.tail(5)
```

```
/usr/local/lib/python3.7/site-packages/pandas/core/frame.py:7123: FutureWarning: Sorting because non-concatenation axis is not aligne
of pandas will change to not sort by default.

To accept the future behavior, pass 'sort=False'.

To retain the current behavior and silence the warning, pass 'sort=True'.

  sort=sort,
```

	Area Id	Beat	Closed Time	Create Time	Incident Type Description	Incident Type Id	Location	Location 1	Priority	Year	C
1044499	P1	02Y	2016-08-01T00:36:46.000	2016-07-31T23:43:51.000	DRUNK ON THE STREET	922	NaN	NaN	2.0	2016	N
1044500	P1	02Y	2016-07-31T23:58:03.000	2016-07-31T23:45:50.000	415 GUNSHOTS	415GS	NaN	NaN	2.0	2016	N
1044501	P3	26Y	2016-08-01T00:08:00.000	2016-07-31T23:50:54.000	DISTURBANCE-NEIGHBOR	415N	NaN	NaN	2.0	2016	N
1044502	P2	19X	2016-08-01T01:33:31.000	2016-07-31T23:56:29.000	SUSPICIOUS PERSON	912	NaN	NaN	2.0	2016	N
1044503	P3	29X	2016-08-01T00:16:16.000	2016-07-31T23:57:31.000	415 FAMILY	415	NaN	NaN	2.0	2016	N

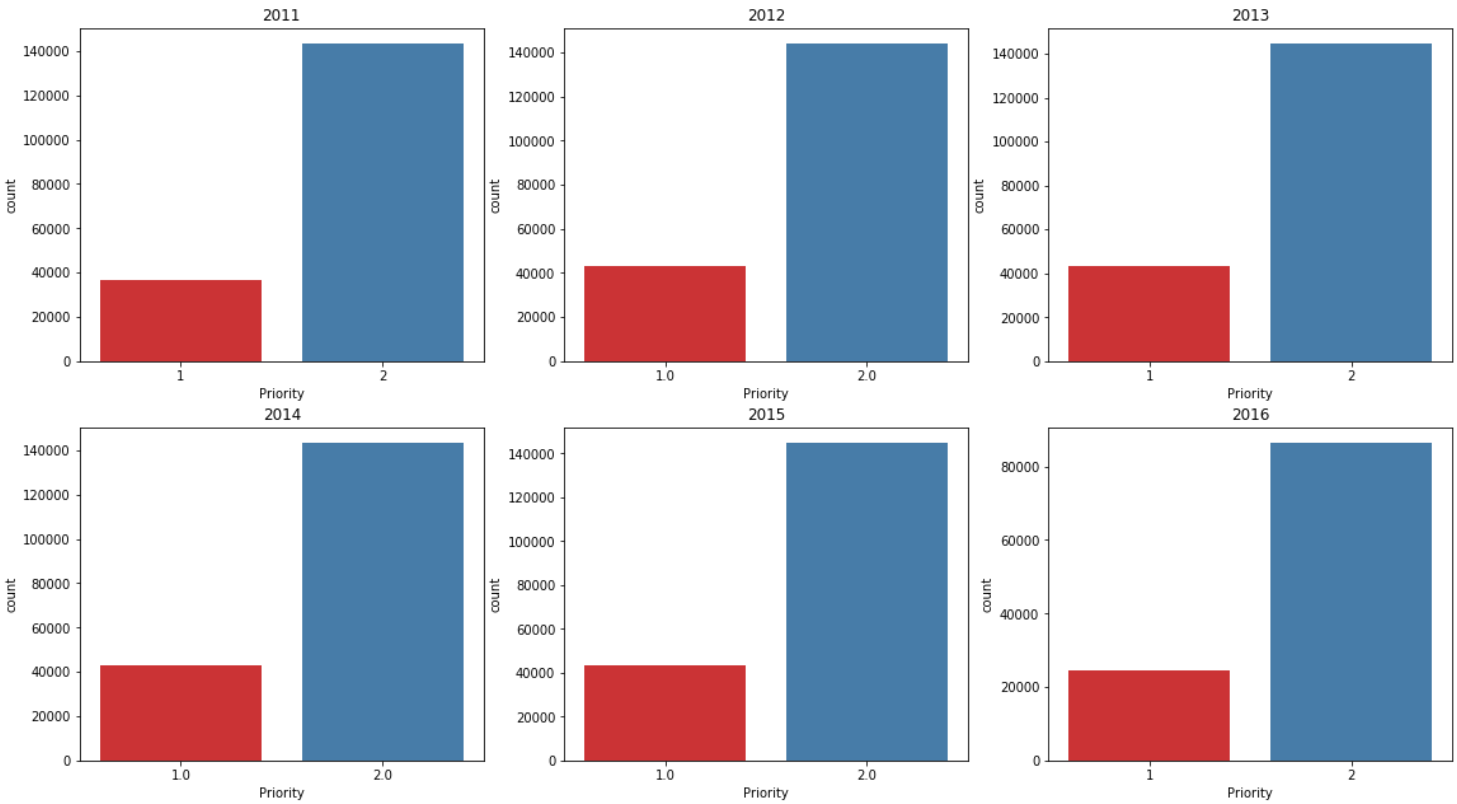
```
temp = combined.groupby(by=["Year", "Priority"]).mean()
prio_1 = temp.loc[list(zip(range(2011,2017),[1.0] * 6))]["time_between_creation_and_closed_seconds"]
prio_2 = temp.loc[list(zip(range(2011,2017),[2.0] * 6))]["time_between_creation_and_closed_seconds"]
plt.plot(range(2011, 2017),prio_1, marker='o', markerfacecolor='black', markersize=8, color='skyblue', linewidth=2, label="Avg Closir
plt.plot(range(2011, 2017), prio_2, marker='*',color="red", markersize=10, markerfacecolor='black', linewidth=2, label="Avg Closing T
plt.legend()
```

```
<matplotlib.legend.Legend at 0x13b6df6d0>
```



每年的报警数量

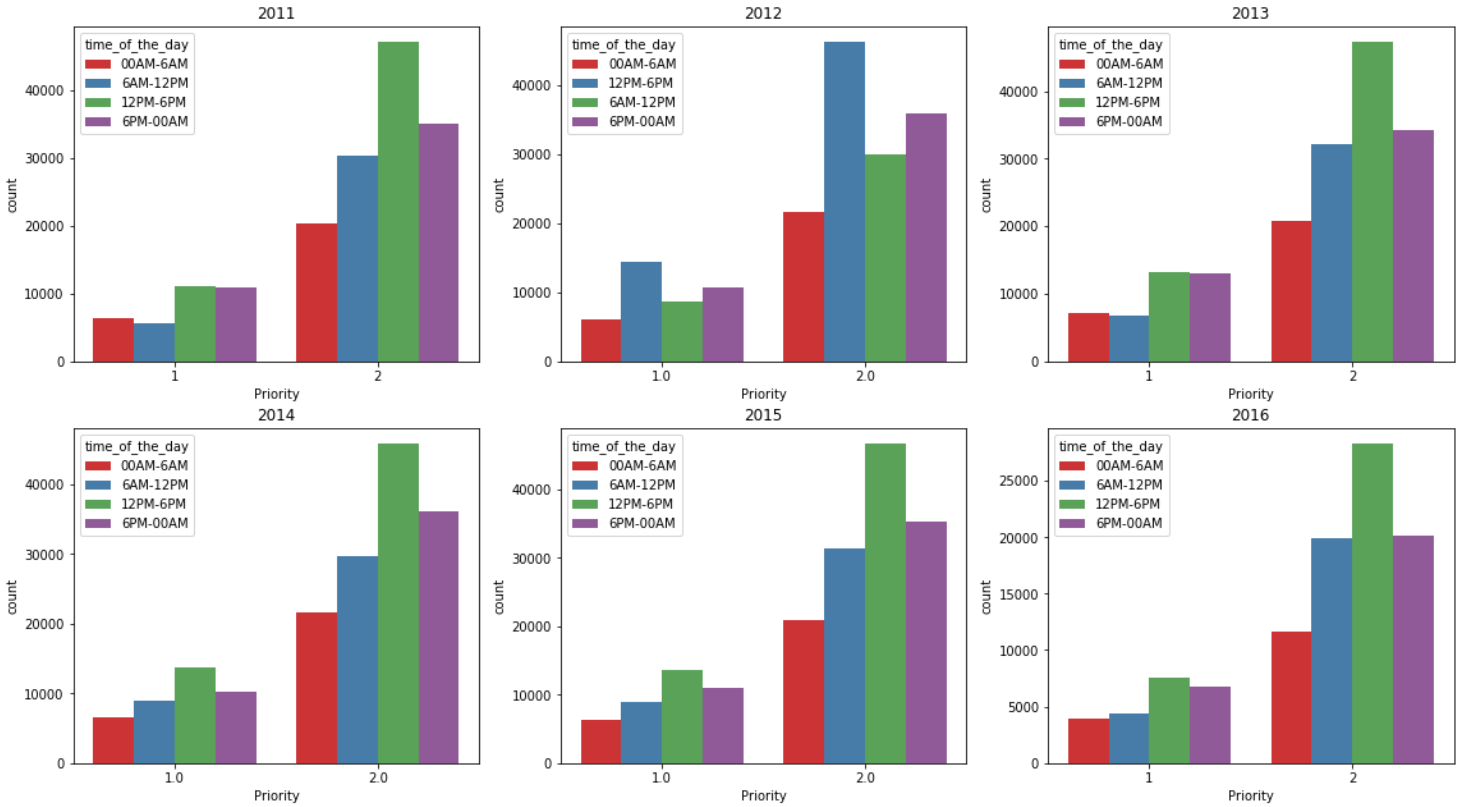
```
fig, ax = plt.subplots(nrows=2, ncols=3)
plt.subplots_adjust(left=0, right=2.5, top=3, bottom=1)
crimes_list = [crimes_2011, crimes_2012, crimes_2013, crimes_2014, crimes_2015, crimes_2016]
i = 0
for row in ax:
    for col in row:
        col.set_title(str(2011 + i))
        sns.countplot(data=crimes_list[i], x="Priority", ax=col, palette="Set1")
        i+=1
```



每年犯罪持续时间盒图

每年报警的时间段分布

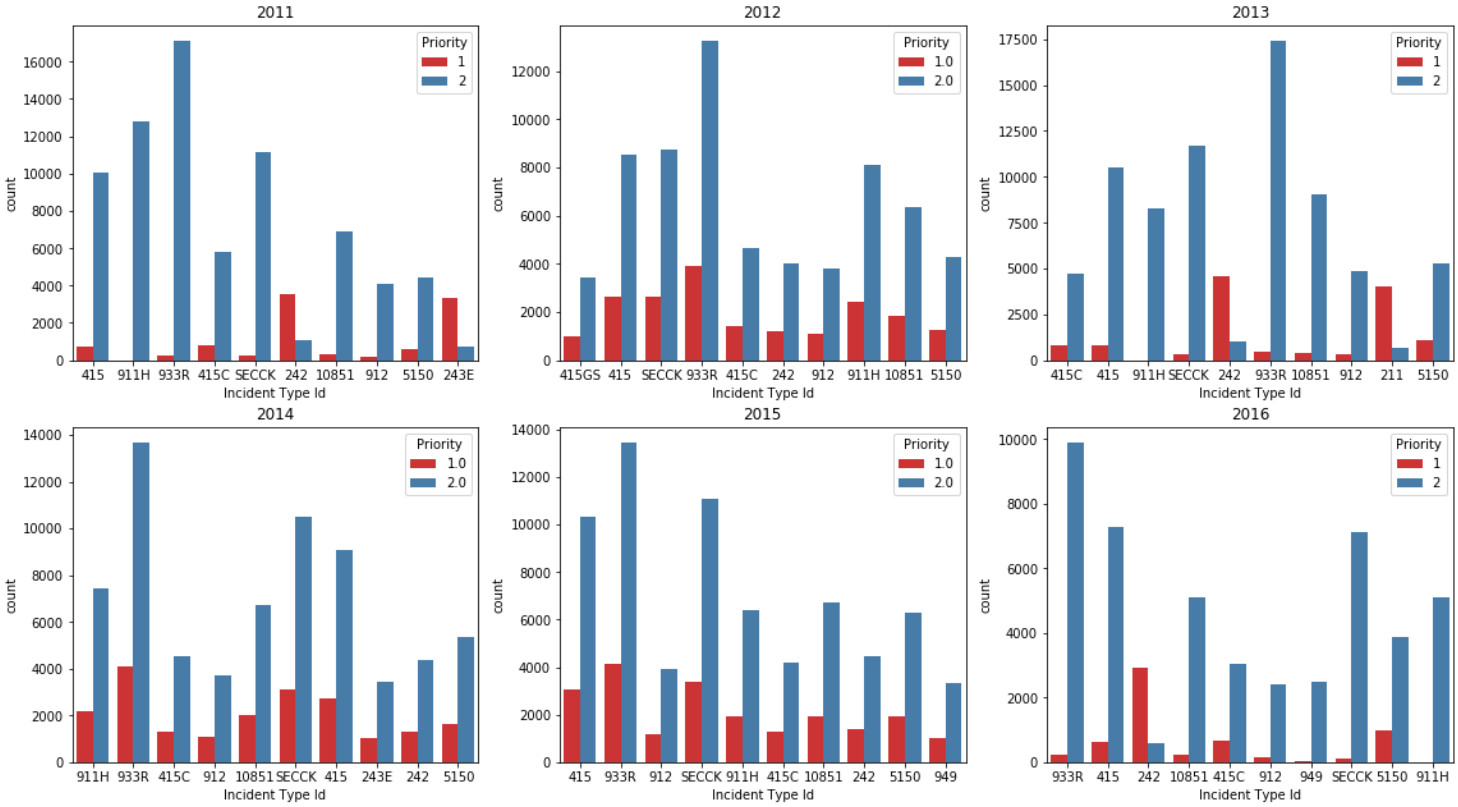
```
fig, ax = plt.subplots(nrows=2, ncols=3)
plt.subplots_adjust(left=0, right=2.5, top=3, bottom=1)
i = 0
for row in ax:
    for col in row:
        col.set_title(str(2011 + i))
        sns.countplot(data=crimes_list[i], x="Priority", hue="time_of_the_day", palette="Set1", ax=col)
        i+=1
```



从上述的图中可以看出优先级为1的案件少，优先级为2的案件多，案件的高发时间段为12PM-6PM

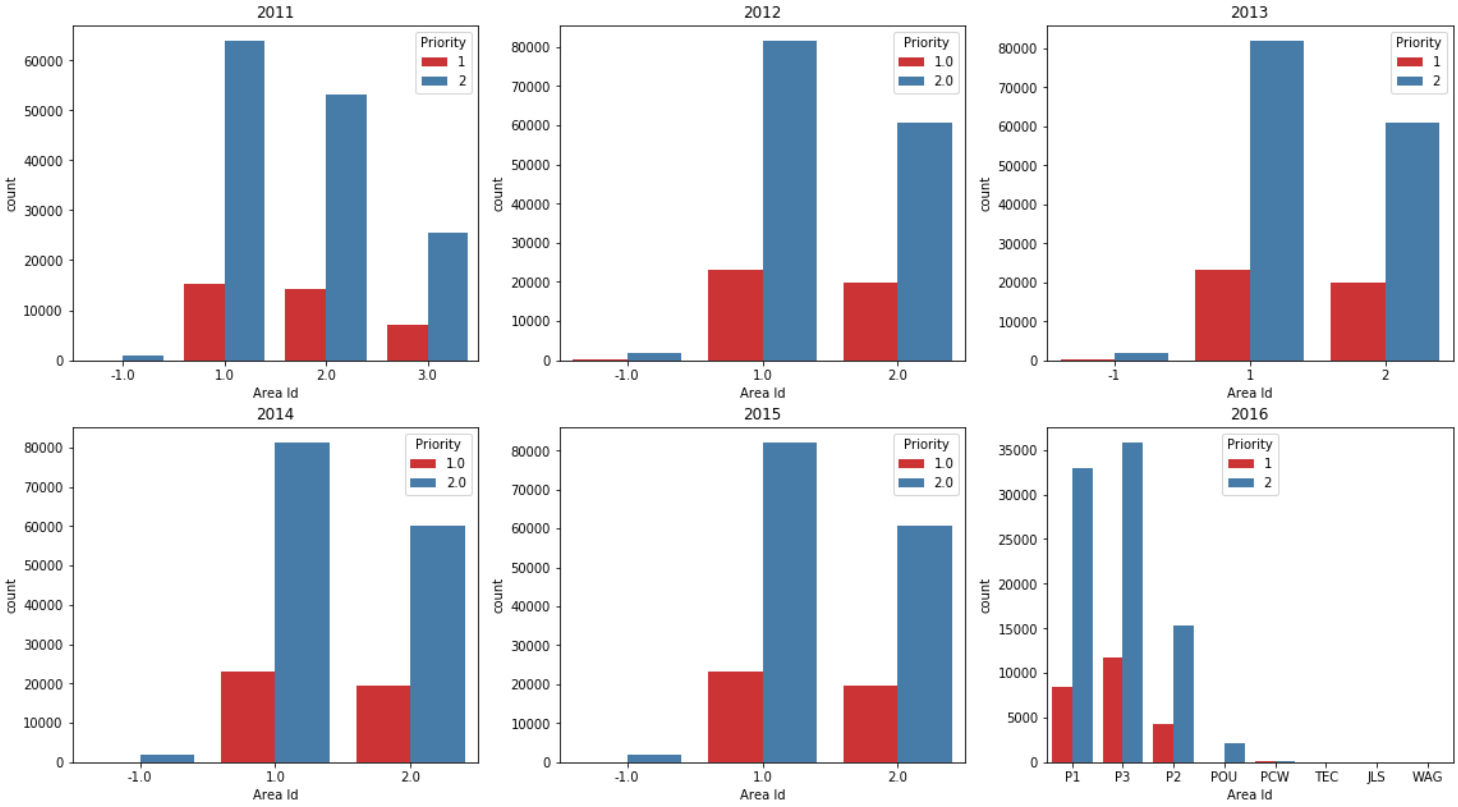
```
fig, ax = plt.subplots(nrows=2, ncols=3)
plt.subplots_adjust(left=0, right=2.5, top=3, bottom=1)
i = 0
nlargest = [set(get_nlargest_incident_id(10, x)) for x in crimes_list]
print("From 2011 to 2016 Top 10 Common Incident Types are: {}".format(str(set.intersection(*nlargest))))
for row in ax:
    for col in row:
        col.set_title(str(2011 + i))
        sns.countplot(data=crimes_list[i].loc[crimes_list[i]['Incident Type Id'].isin(nlargest[i])], x="Incident Type Id", hue="Prior
        i += 1
```

```
From 2011 to 2016 Top 10 Common Incident Types are: {'10851', '911H', 'SECCK', '242', '933R', '5150', '415', '912', '415C'}
```

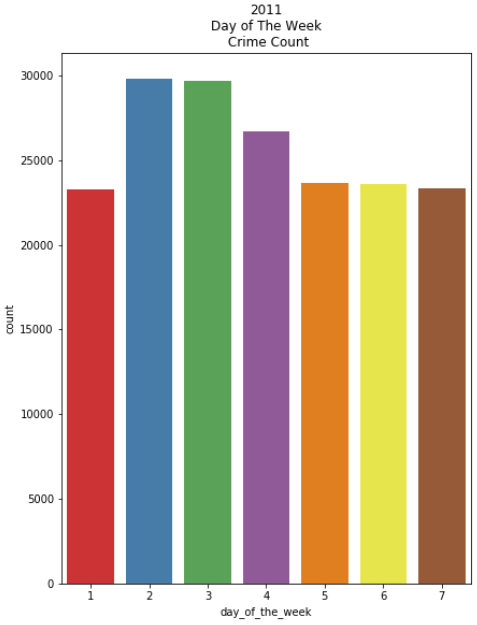


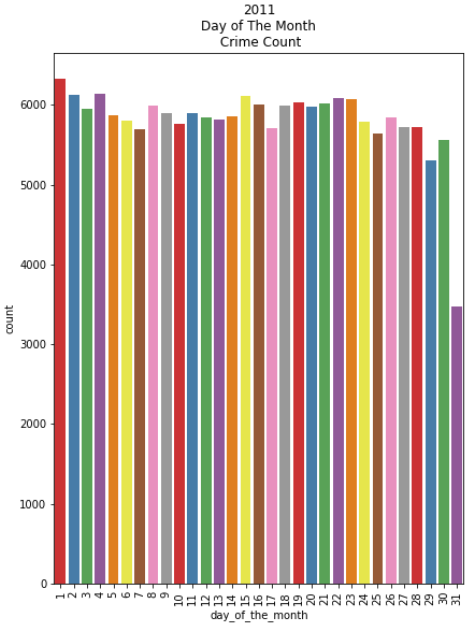

```
fig, ax = plt.subplots(nrows=2, ncols=3)
plt.subplots_adjust(left=0, right=2.5, top=3, bottom=1)
i = 0
area_nlargest = [set(get_nlargest_area_id(10, x)) for x in crimes_list]
print("From 2011 to 2016 Top 10 Common Area Id are: {}".format(str(set.intersection(*area_nlargest))))
for row in ax:
    for col in row:
        col.set_title(str(2011 + i))
        sns.countplot(data=crimes_list[i].loc[crimes_list[i]['Area Id'].isin(area_nlargest[i])], x="Area Id", hue="Priority", palette
        i += 1
```

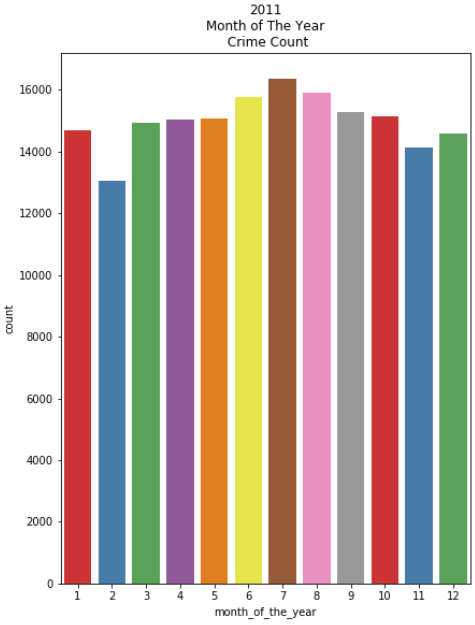
```
From 2011 to 2016 Top 10 Common Area Id are: set()
```

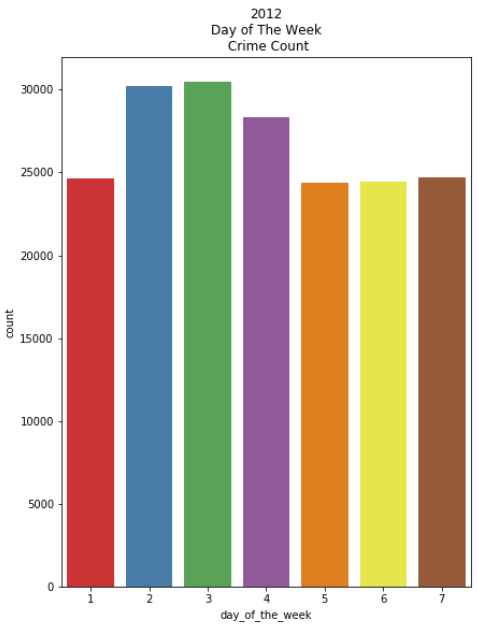


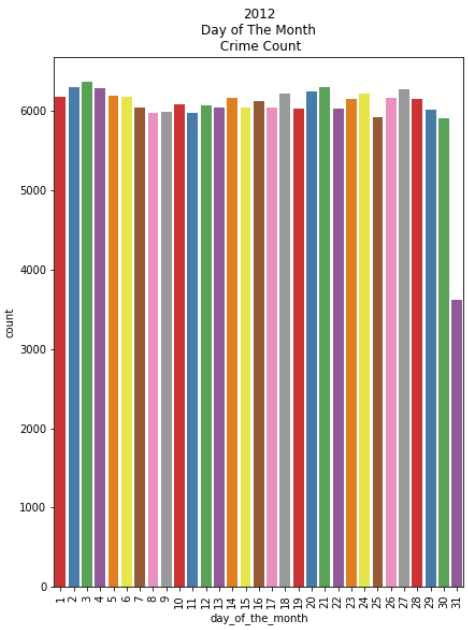

```
fig, ax = plt.subplots(nrows=6, ncols=3)
plt.subplots_adjust(left=0, right=3, top=12, bottom=0)
i_list = 0
for i, row in enumerate(ax):
    for j, col in enumerate(row):
        year_string = str(2011 + i)

        if j == 1:
            month_or_day = 'Day of The Month'
            title = year_string+ '\n' + month_or_day +'\n Crime Count'
            col.set_title(title)
            col.set_xticklabels(col.get_xticklabels(), rotation=90)
            sns.countplot(data=crimes_list[i_list], x="day_of_the_month" ,palette="Set1", ax=col)
        elif j == 2:
            month_or_day = 'Month of The Year'
            title = year_string + '\n' + month_or_day +'\n Crime Count'
            col.set_title(title)
            sns.countplot(data=crimes_list[i_list], x="month_of_the_year",palette="Set1", ax=col)
        else:
            month_or_day = 'Day of The Week'
            title = year_string+ '\n' + month_or_day +'\n Crime Count'
            col.set_title(title)
            sns.countplot(data=crimes_list[i_list], x="day_of_the_week" ,palette="Set1", ax=col)

    i_list += 1
```

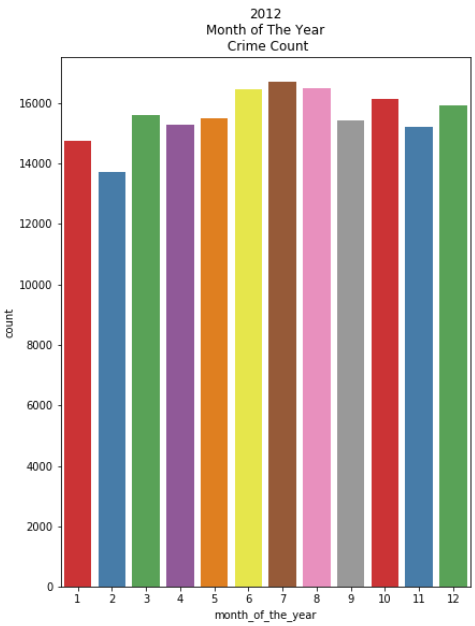

2011
Day of The Week
Crime Count
count
day_of_the_week

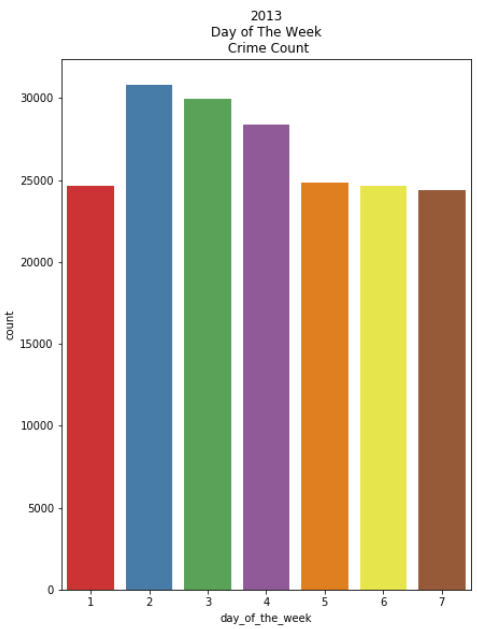

2011
Day of The Month
Crime Count
count
day_of_the_month

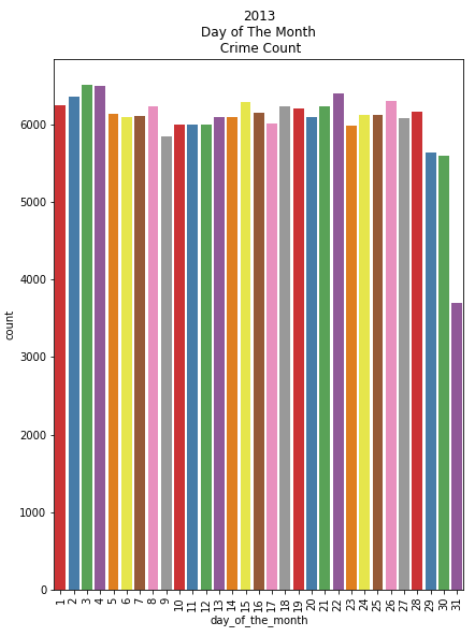

2011
Month of The Year
Crime Count
count
month_of_the_year

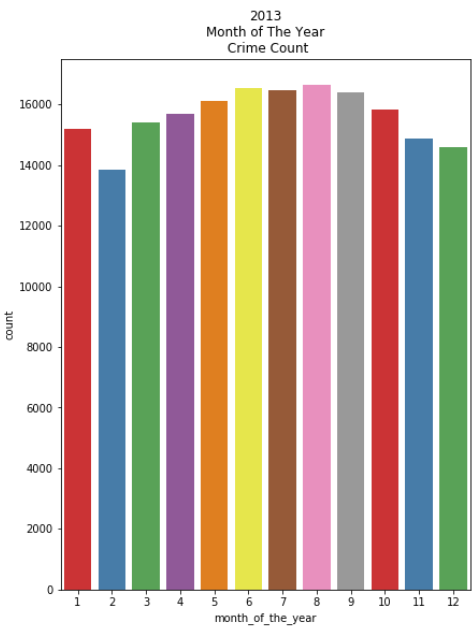

2012
Day of The Week
Crime Count
count
day_of_the_week

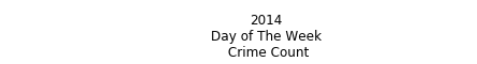

2012
Day of The Month
Crime Count
count
day_of_the_month


2012
Month of The Year
Crime Count
count
month_of_the_year


2013
Day of The Week
Crime Count
count
day_of_the_week

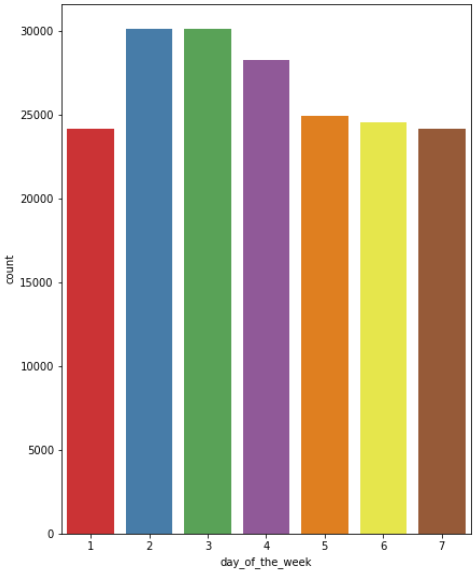

2013
Day of The Month
Crime Count
count
day_of_the_month

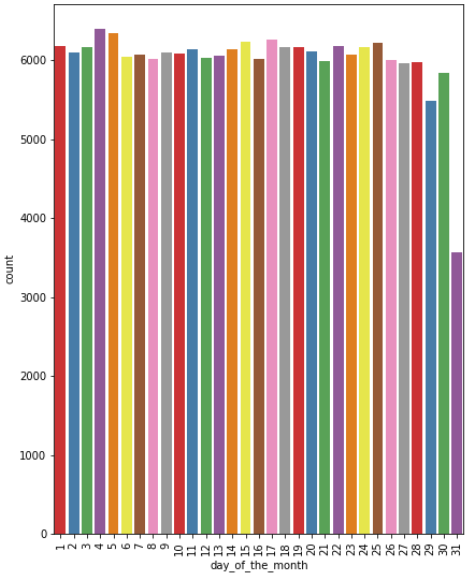

2013
Month of The Year
Crime Count
count
month_of_the_year

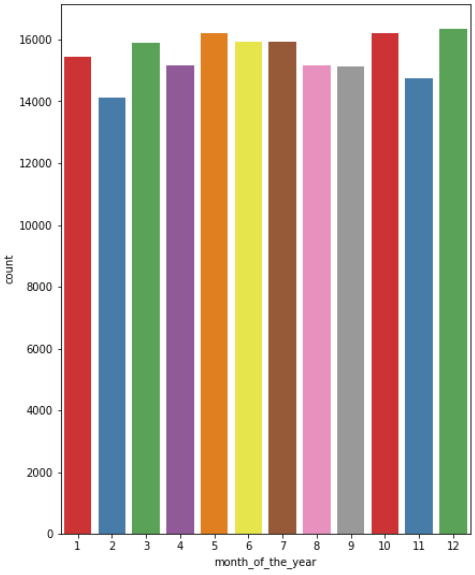

2014
Day of The Week
Crime Count


2014
Day of The Month
Crime Count

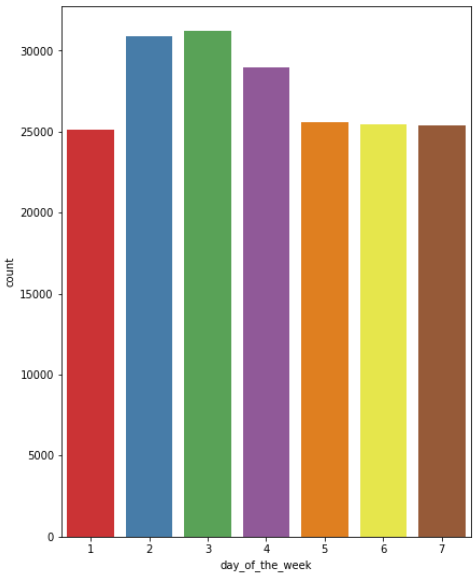

2014
Month of The Year
Crime Count



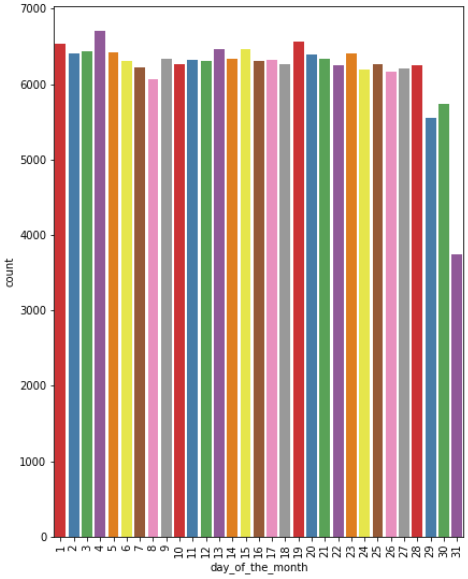




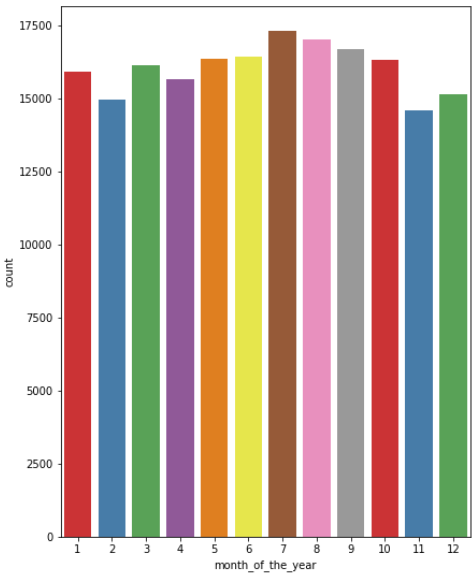
2015
Day of The Week
Crime Count



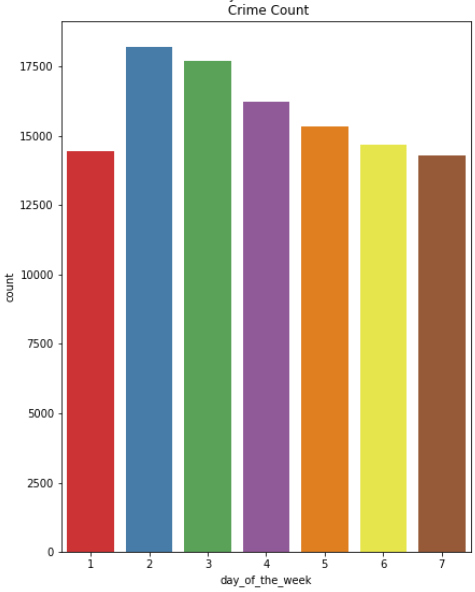
2015
Day of The Month
Crime Count



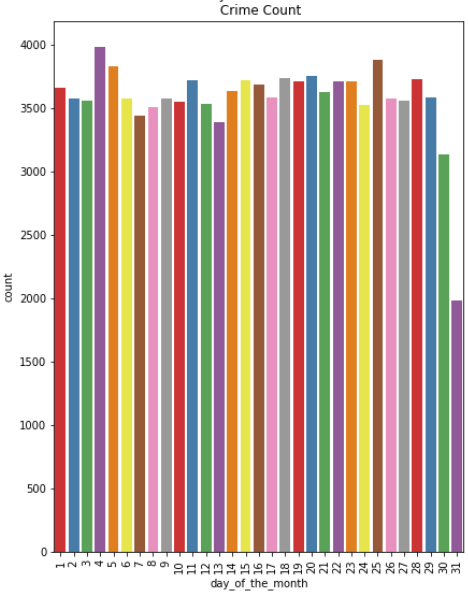
2015
Month of The Year
Crime Count



2016
Day of The Week
Crime Count



2016
Day of The Month
Crime Count



2016
Month of The Year
Crime Count