MutualEvaluationHomework_1

2021 年 3 月 30 日

[135]:
```
import pandas as pd
import matplotlib.pyplot as plt
from scipy.spatial.distance import pdist
from math import ceil
import numpy as np
%matplotlib inline

print("项目 GitHub 网址: https://github.com/Xiemixue/
 ↪DataMining_MutualEvaluationAssignment_1")

print("Wine Reviews 数据集的标称属性有: country, province, variety, winery,␣
 ↪designation, region_1, region_2\n")

print("Oakland Crime Statistics 2011 to 2016s 数据集的标称属性为: Location, Area␣
 ↪id, Incident Type id, Incident Type Descripe, EventNumber (其中 2012 和 2014
的属性为 Location 1)")
print("(注: 由于这个犯罪数据集并没有严格意义上的数值属性, 因此无法对于数值属性进行五
数概括, 也没有绘制其盒图和直方图)")
```

```
项目 GitHub 网址: https://github.com/Xiemixue/
 ↪DataMining_MutualEvaluationAssignment_1
Wine Reviews 数据集的标称属性有: country, province, variety, winery, designation,␣
 ↪region_1,region_2

Oakland Crime Statistics 2011 to 2016s 数据集的标称属性为: Location, Area id,␣
 ↪Incident
Type id, Incident Type Descripe, EventNumber (其中 2012 和 2014 的属性为 Location␣
 ↪1)
```

(注：由于这个犯罪数据集并没有严格意义上的数值属性，因此无法对于数值属性进行五数概括，也没有绘制其盒图和直方图)

[44]:
```
# 频数聚合

import pandas as pd
import matplotlib.pyplot as plt
from scipy.spatial.distance import pdist
from math import ceil
import numpy as np
%matplotlib inline

data1_wine_review = pd.read_csv("winemag-data_first150k.csv", encoding="utf-8")[
    ["country", "province", "variety", "winery", "designation",␣
 ↪"region_1","region_2" ]]
data2_wine_review = pd.read_csv("winemag-data-130k-v2.csv", encoding="utf-8")[
    ["country", "province", "variety", "winery", "designation",␣
 ↪"region_1","region_2" ]]
data_all_wine = [data1_wine_review, data2_wine_review]
wine_name_list = ["winemag-data_first150k.csv", "winemag-data-130k-v2.csv"]
i = 0
for datax in data_all_wine:
    print("\n" + wine_name_list[i] + "的频数聚合分析:")
    data1_number = datax
    data1_number = data1_number.dropna(axis=0,how='all')

    data1_country=data1_number["country"].value_counts(sort=True)
    data1_country=data1_country.head(5)
    data1_country_name=data1_country.index.tolist()
    data1_country_num=data1_country.values

    data1_province=data1_number["province"].value_counts(sort=True)
    data1_province=data1_province.head(5)
    data1_province_name=data1_province.index.tolist()
    data1_province_num=data1_province.values

    data1_variety=data1_number["variety"].value_counts(sort=True)
```

```
data1_variety=data1_variety.head(5)
data1_variety_name=data1_variety.index.tolist()
data1_variety_num=data1_variety.values

data1_winery=data1_number["winery"].value_counts(sort=True)
data1_winery=data1_winery.head(5)
data1_winery_name=data1_winery.index.tolist()
data1_winery_num=data1_winery.values

data1_designation=data1_number["designation"].value_counts(sort=True)
data1_designation=data1_designation.head(5)
data1_designation_name=data1_designation.index.tolist()
data1_designation_num=data1_designation.values

data1_region_1=data1_number["region_1"].value_counts(sort=True)
data1_region_1=data1_region_1.head(5)
data1_region_1_name=data1_region_1.index.tolist()
data1_region_1_num=data1_region_1.values

data1_region_2=data1_number["region_2"].value_counts(sort=True)
data1_region_2=data1_region_2.head(5)
data1_region_2_name=data1_region_2.index.tolist()
data1_region_2_num=data1_region_2.values

index=np.arange(5)

plt.figure(figsize=(15,10))
ax1 = plt.subplot(2,2,1)
ax2 = plt.subplot(2,2,2)

plt.sca(ax1)
plt.bar(index,data1_country_num, 0.5, label="num")
plt.xticks(index,data1_country_name)
for a,b in zip(index,data1_country_num):
    plt.text(a, b+0.05, '%.0f' % b, ha='center', va= 'bottom',fontsize=11)
plt.title("country")
```

```
plt.sca(ax2)
plt.bar(index,data1_province_num, 0.5, label="num")
plt.xticks(index,data1_province_name)
for a,b in zip(index,data1_province_num):
    plt.text(a, b+0.05, '%.0f' % b, ha='center', va= 'bottom',fontsize=11)
plt.title("province")

plt.figure(figsize=(20,10))
ax3 = plt.subplot(2,2,1)
ax4 = plt.subplot(2,2,2)

plt.sca(ax3)
plt.bar(index,data1_variety_num, 0.5, label="num")
plt.xticks(index,data1_variety_name)
for a,b in zip(index,data1_variety_num):
    plt.text(a, b+0.05, '%.0f' % b, ha='center', va= 'bottom',fontsize=11)
plt.title("variety")

plt.sca(ax4)
plt.bar(index,data1_winery_num, 0.5, label="num")
plt.xticks(index,data1_winery_name)
for a,b in zip(index,data1_winery_num):
    plt.text(a, b+0.05, '%.0f' % b, ha='center', va= 'bottom',fontsize=11)
plt.title("winery")

plt.figure(figsize=(5,5))
plt.bar(index,data1_designation_num, 0.5, label="num")
plt.xticks(index,data1_designation_name)
for a,b in zip(index,data1_designation_num):
    plt.text(a, b+0.05, '%.0f' % b, ha='center', va= 'bottom',fontsize=11)
plt.title("designation")

plt.figure(figsize=(20,10))
ax5 = plt.subplot(2,2,1)
ax6 = plt.subplot(2,2,2)
```

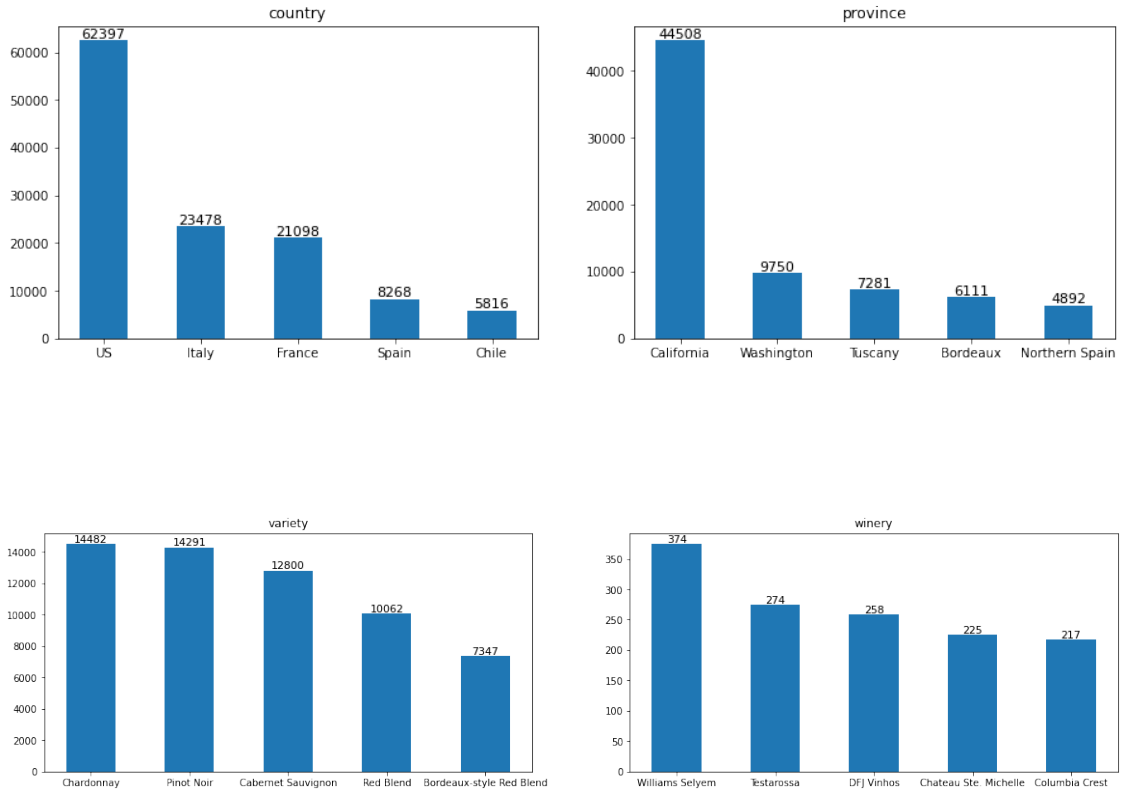
```
    plt.sca(ax5)
    plt.bar(index,data1_region_1_num, 0.5, label="num")
    plt.xticks(index,data1_region_1_name)
    for a,b in zip(index,data1_region_1_num):
        plt.text(a, b+0.05, '%.0f' % b, ha='center', va= 'bottom',fontsize=11)
    plt.title("region_1")

    plt.sca(ax6)
    plt.bar(index,data1_region_2_num, 0.5, label="num")
    plt.xticks(index,data1_region_2_name)
    for a,b in zip(index,data1_region_2_num):
        plt.text(a, b+0.05, '%.0f' % b, ha='center', va= 'bottom',fontsize=11)
    plt.title("region_2")

    plt.show()
    i+=1



data1 = pd.read_csv("records-for-2011.csv",encoding="utf-8")
data2 = pd.read_csv("records-for-2012.csv",encoding="utf-8")
data3 = pd.read_csv("records-for-2013.csv",encoding="utf-8")
data4 = pd.read_csv("records-for-2014.csv",encoding="utf-8")
data5 = pd.read_csv("records-for-2015.csv",encoding="utf-8")
data6 = pd.read_csv("records-for-2016.csv",encoding="utf-8")
data_all_crime = [data1, data2, data3, data4, data5, data6]

year=2011
for datax in data_all_crime:
    data1_number=datax
    data1_number=data1_number.dropna(axis=0,how='all')
    print("\nrecords-for-" + str(year)+".csv 的频数聚合分析:")
    data1_location_g=[]
    if(year==2012 or year==2014):
        data1_location=data1_number["Location 1"].value_counts(sort=True)
```

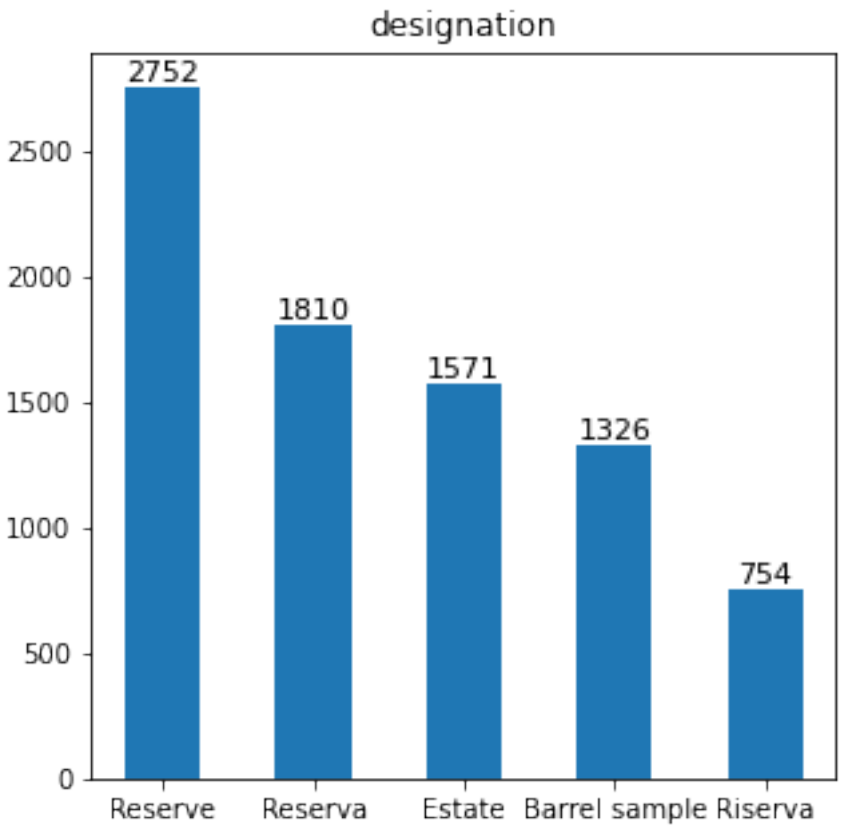
```
        data1_location=data1_location.head(5)
        data1_location_name=data1_location.index.tolist()
        data1_location_num=data1_location.values
        for data_g in data1_location_name:
            data1_location_g.append(data_g.split('"')[3])
        data1_location_name=data1_location_g
    elif(year==2013):
        data1_location=data1_number["Location "].value_counts(sort=True)
        data1_location=data1_location.head(5)
        data1_location_name=data1_location.index.tolist()
        data1_location_num=data1_location.values
    else:
        data1_location=data1_number["Location"].value_counts(sort=True)
        data1_location=data1_location.head(5)
        data1_location_name=data1_location.index.tolist()
        data1_location_num=data1_location.values

    data1_area=data1_number["Area Id"].value_counts(sort=True)
    data1_area=data1_area.head(2)
    data1_area_name=data1_area.index.tolist()
    data1_area_num=data1_area.values

    data1_id=data1_number["Incident Type Id"].value_counts(sort=True)
    data1_id=data1_id.head(5)
    data1_id_name=data1_id.index.tolist()
    data1_id_num=data1_id.values

    data1_Description=data1_number["Incident Type Description"].
↪value_counts(sort=True)
    data1_Description=data1_Description.head(5)
    data1_Description_name=data1_Description.index.tolist()
    data1_Description_num=data1_Description.values

    data1_Event=data1_number["Event Number"].value_counts(sort=True)
    data1_Event=data1_Event.head(5)
    data1_Event_name=data1_Event.index.tolist()
```

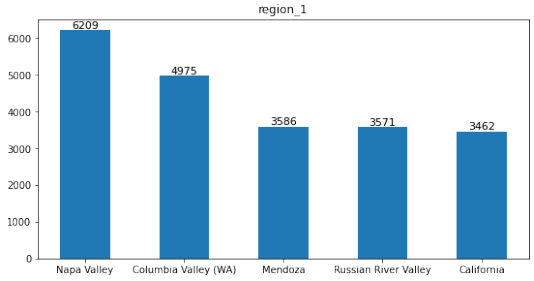
```
data1_Event_num=data1_Event.values

index=np.arange(5)
index_id=np.arange(2)
plt.figure(figsize=(10, 5))
plt.bar(index,data1_location_num, 0.5, label="num")
plt.xticks(index,data1_location_name)
for a,b in zip(index,data1_location_num):
    plt.text(a, b+0.05, '%.0f' % b, ha='center', va= 'bottom',fontsize=11)
plt.title("Location")

plt.figure(figsize=(10,10))
ax1 = plt.subplot(2,2,1)
ax2 = plt.subplot(2,2,2)

plt.sca(ax1)
plt.bar(index_id,data1_area_num, 0.5, label="num")
plt.xticks(index_id,data1_area_name)
for a,b in zip(index_id,data1_area_num):
    plt.text(a, b+0.05, '%.0f' % b, ha='center', va= 'bottom',fontsize=11)
plt.title("Area Id")

plt.sca(ax2)
plt.bar(index,data1_id_num, 0.5, label="num")
plt.xticks(index,data1_id_name)
for a,b in zip(index,data1_id_num):
    plt.text(a, b+0.05, '%.0f' % b, ha='center', va= 'bottom',fontsize=11)
plt.title("Incident Type Id")

plt.figure(figsize=(20,10))
ax3 = plt.subplot(2,2,1)
ax4 = plt.subplot(2,2,2)
plt.sca(ax3)
plt.bar(index,data1_Description_num, 0.5, label="num")
plt.xticks(index,data1_Description_name)
for a,b in zip(index,data1_Description_num):
```

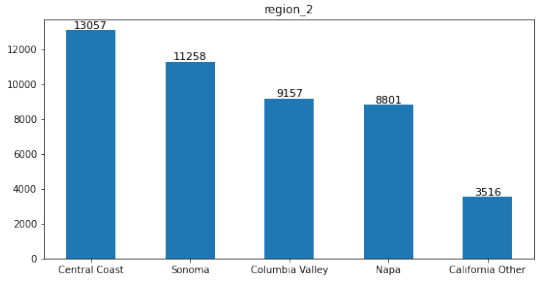
```
        plt.text(a, b+0.05, '%.0f' % b, ha='center', va= 'bottom',fontsize=11)
    plt.title("Incident Type Description")

    plt.sca(ax4)
    plt.bar(index,data1_Event_num, 0.5, label="num")
    plt.xticks(index,data1_Event_name)
    for a,b in zip(index,data1_Event_num):
        plt.text(a, b+0.05, '%.0f' % b, ha='center', va= 'bottom',fontsize=11)
    plt.title("Event Number")

    plt.show()
    year+=1
```

winemag-data_first150k.csv 的频数聚合分析:

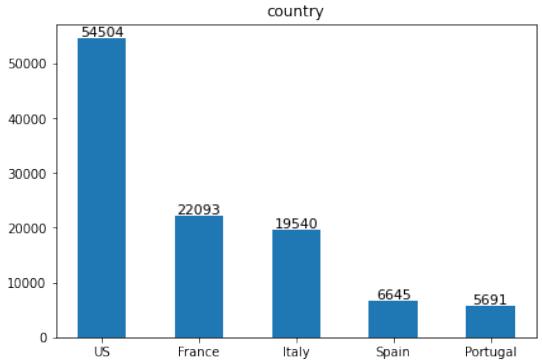


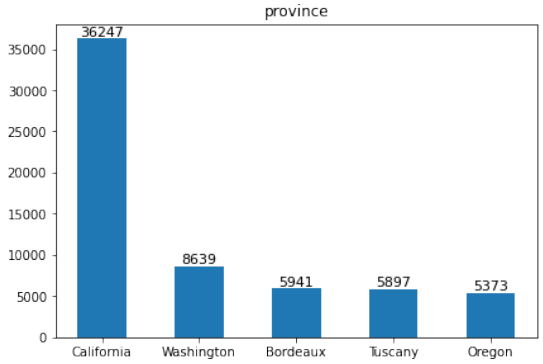


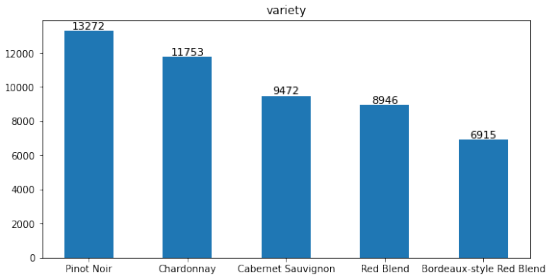


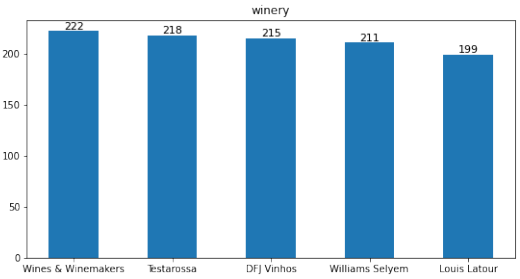


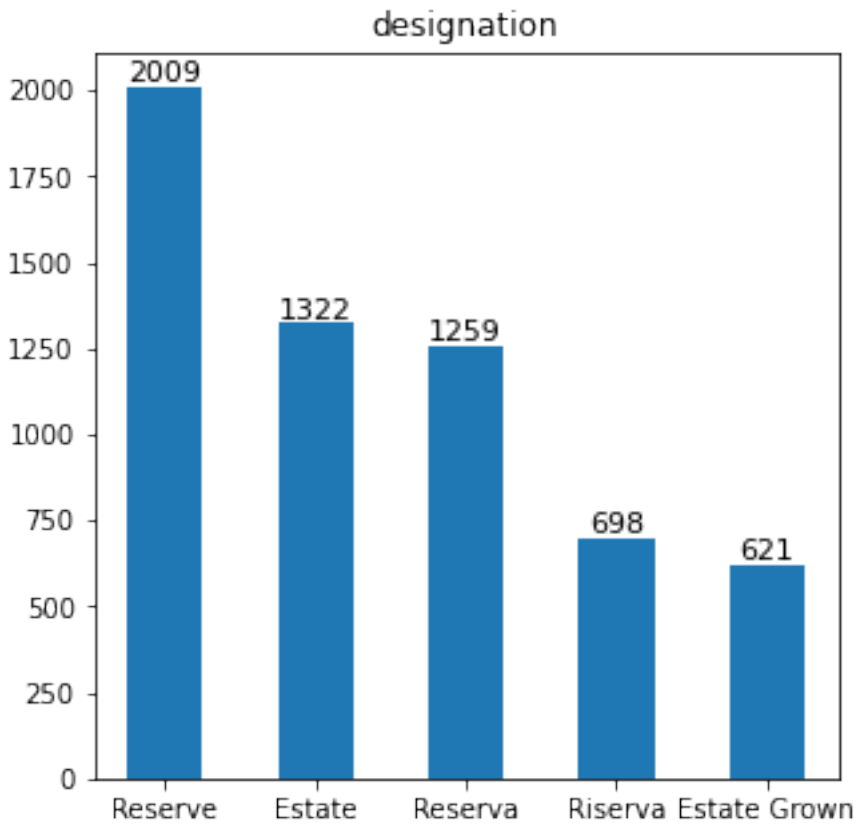
winemag-data-130k-v2.csv 的频数聚合分析:

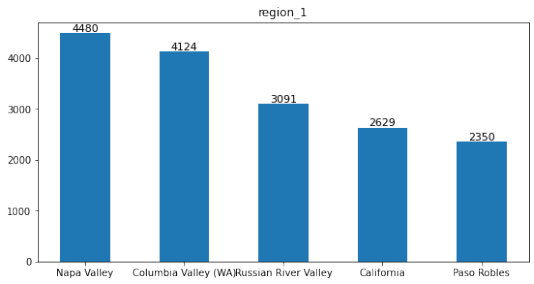

country
54504
22093
19540
6645
5691
US
France
Italy
Spain
Portugal

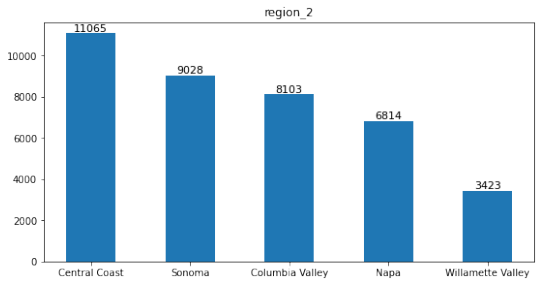

province
36247
8639
5941
5897
5373
California
Washington
Bordeaux
Tuscany
Oregon


variety
13272
11753
9472
8946
6915
Pinot Noir
Chardonnay
Cabernet Sauvignon
Red Blend
Bordeaux-style Red Blend

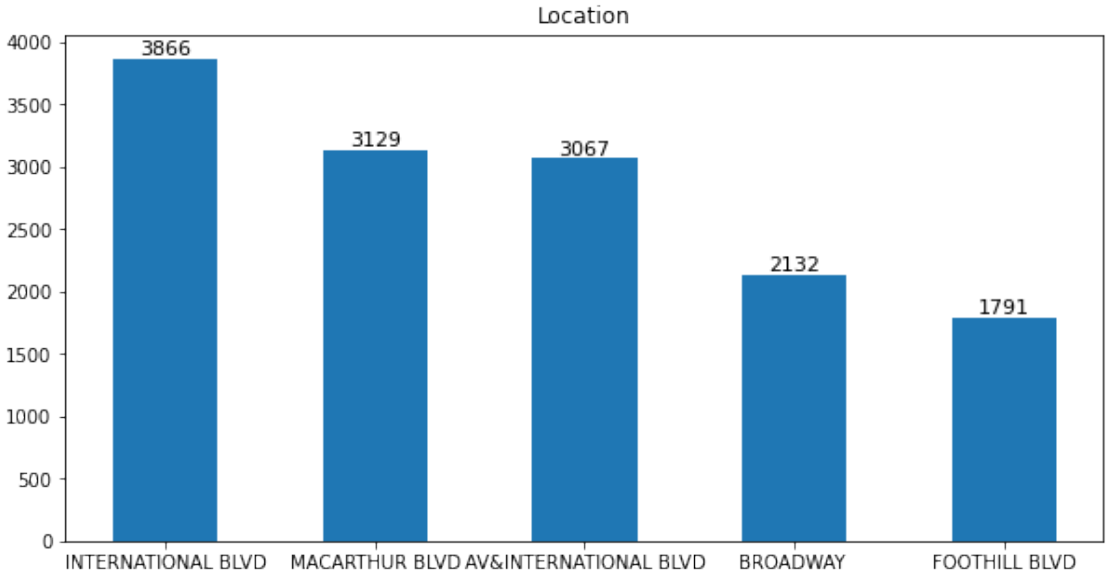

winery
222
218
215
211
199
Wines & Winemakers
Testarossa
DFJ Vinhos
Williams Selyem
Louis Latour

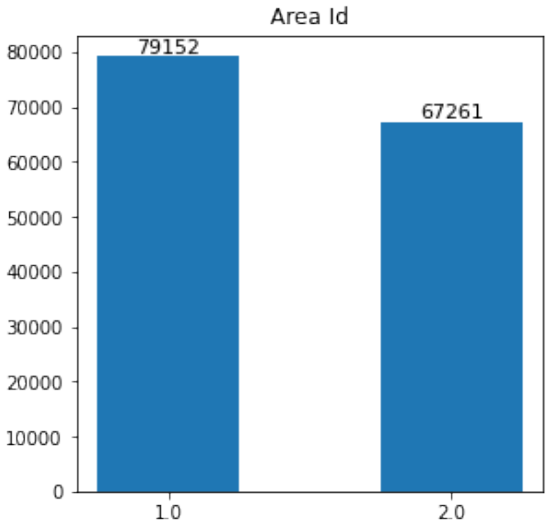


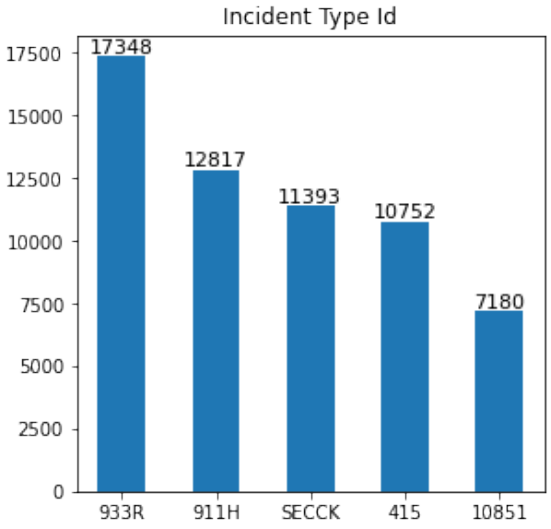


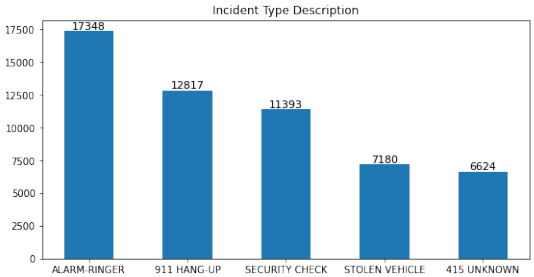


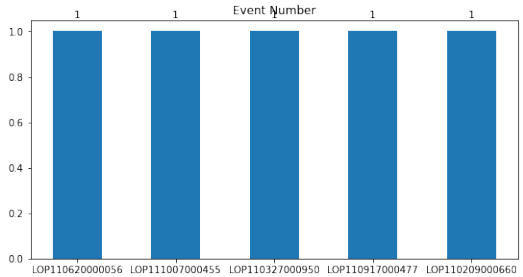
records-for-2011.csv 的频数聚合分析:


Location
4000
3500
3000
2500
2000
1500
1000
500
0
3866
3129
3067
2132
1791
INTERNATIONAL BLVD
MACARTHUR BLVD
AV&INTERNATIONAL BLVD
BROADWAY
FOOTHILL BLVD

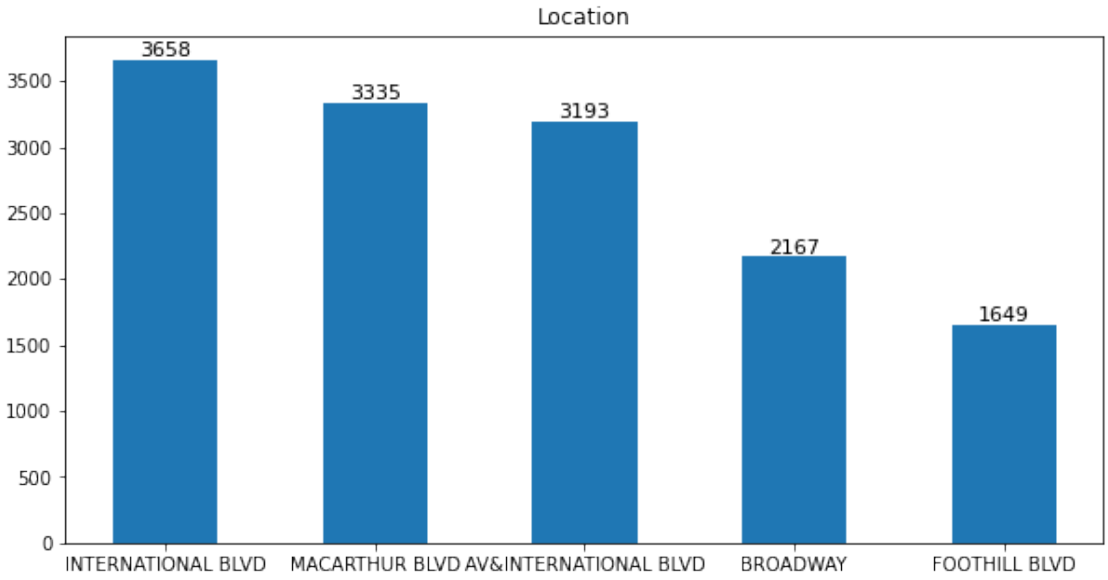

Area Id
80000
70000
60000
50000
40000
30000
20000
10000
0
79152
67261
1.0
2.0

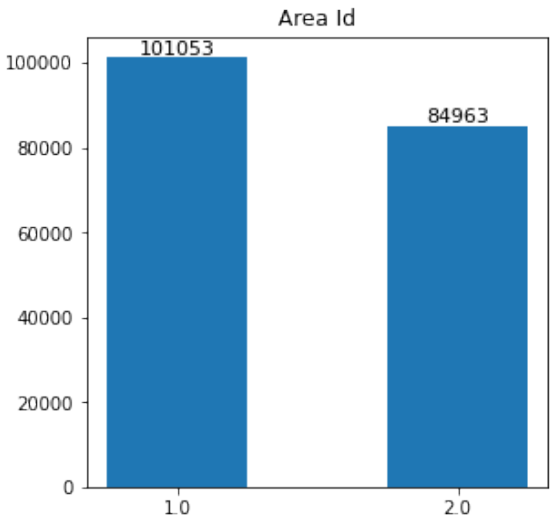

Incident Type Id
17500
15000
12500
10000
7500
5000
2500
0
17348
12817
11393
10752
7180
933R
911H
SECCK
415
10851

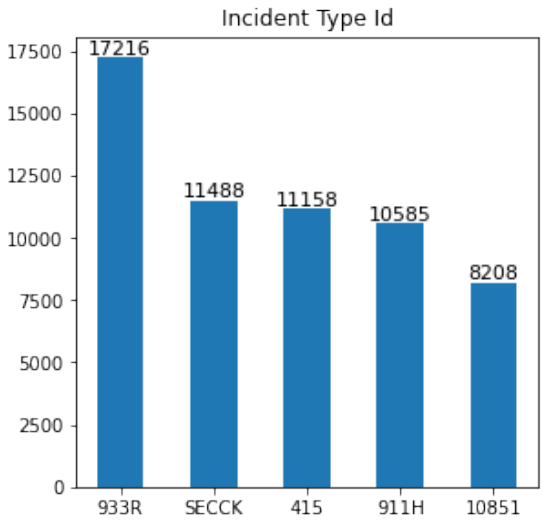

Incident Type Description
17500
15000
12500
10000
7500
5000
2500
0
17348
12817
11393
7180
6624
ALARM-RINGER
911 HANG-UP
SECURITY CHECK
STOLEN VEHICLE
415 UNKNOWN

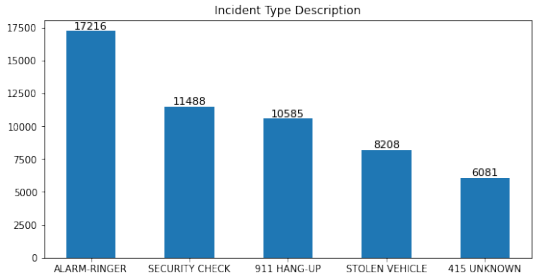

Event Number
1.0
0.8
0.6
0.4
0.2
0.0
1
1
1
1
1
LOP110620000056
LOP111007000455
LOP110327000950
LOP110917000477
LOP110209000660

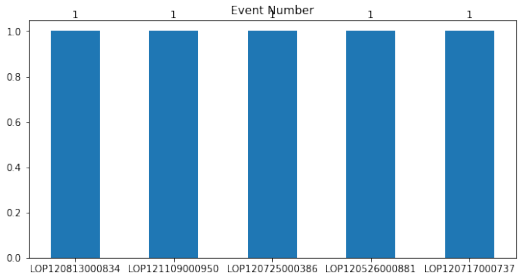
records-for-2012.csv 的频数聚合分析：



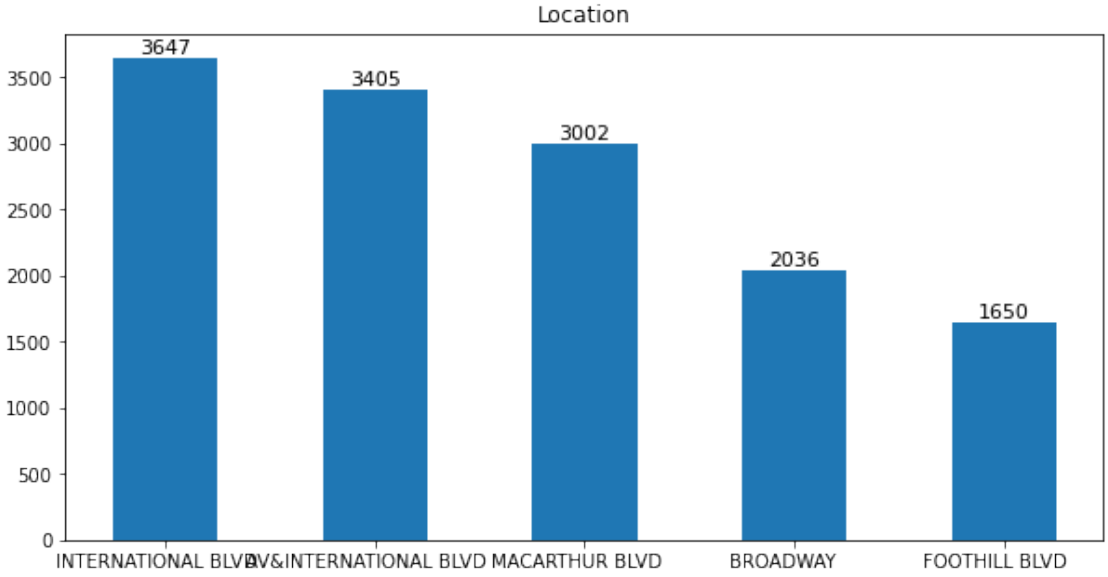


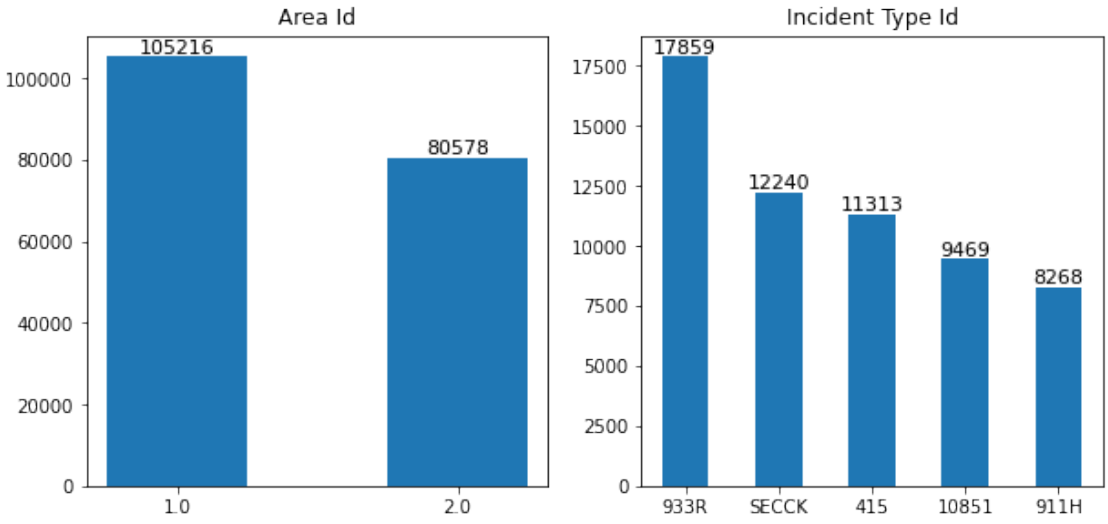


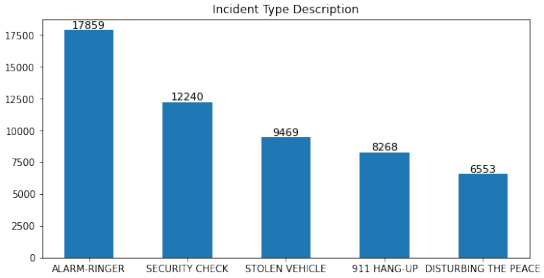


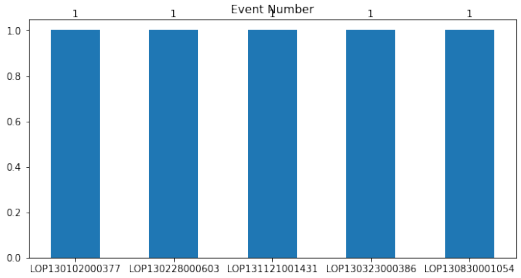


records-for-2013.csv 的频数聚合分析:

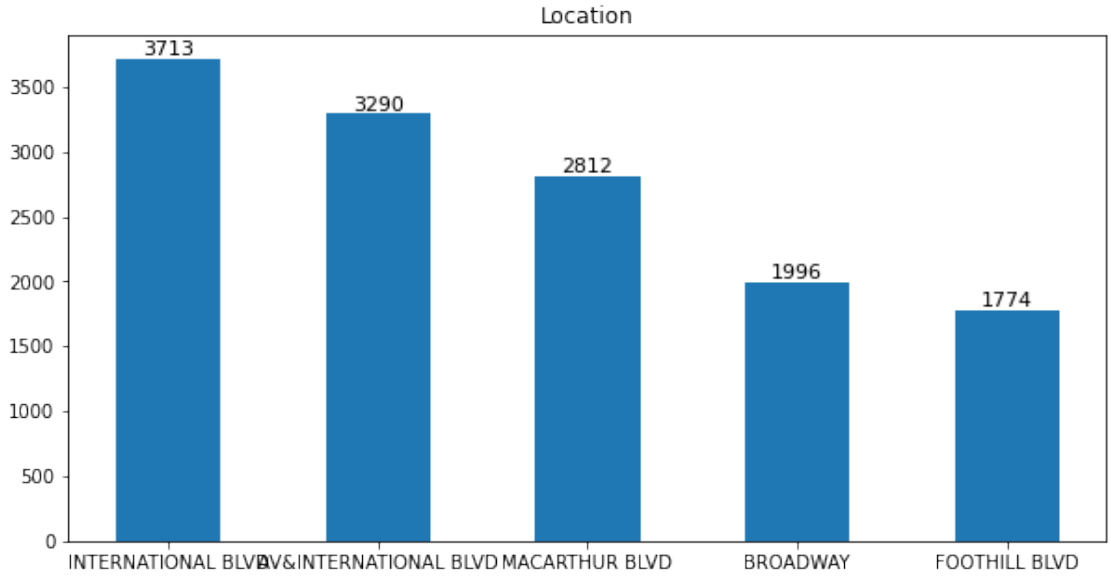


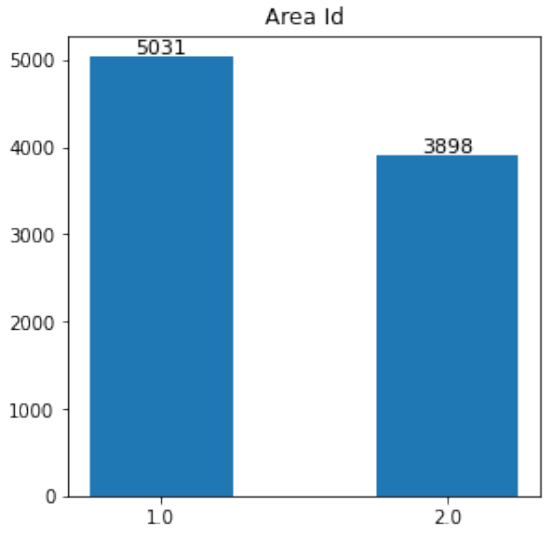


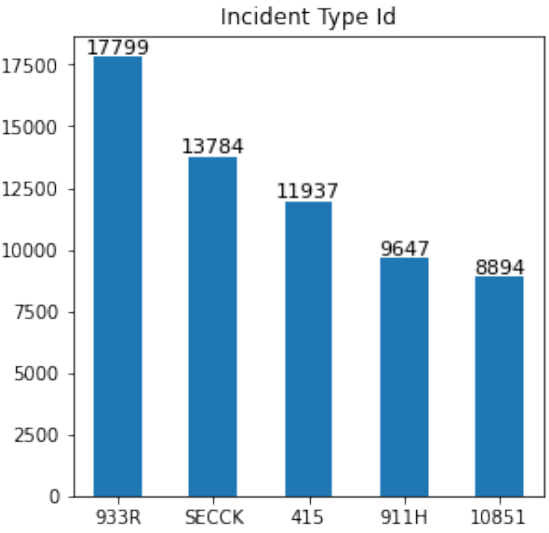


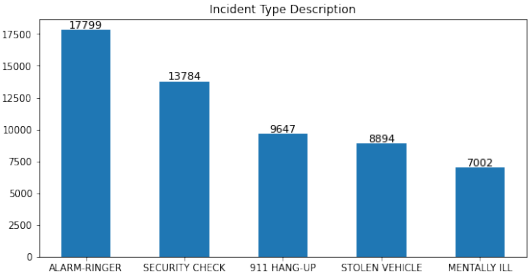


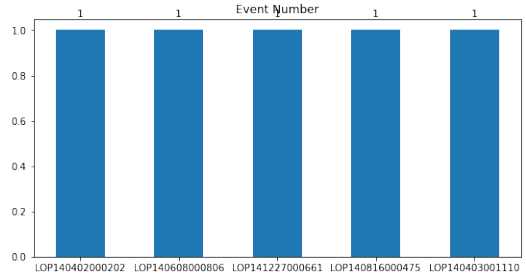
records-for-2014.csv 的频数聚合分析:


Location
3713
3290
2812
1996
1774
3500
3000
2500
2000
1500
1000
500
0
INTERNATIONAL BLVD
INTERNATIONAL BLVD
MACARTHUR BLVD
BROADWAY
FOOTHILL BLVD

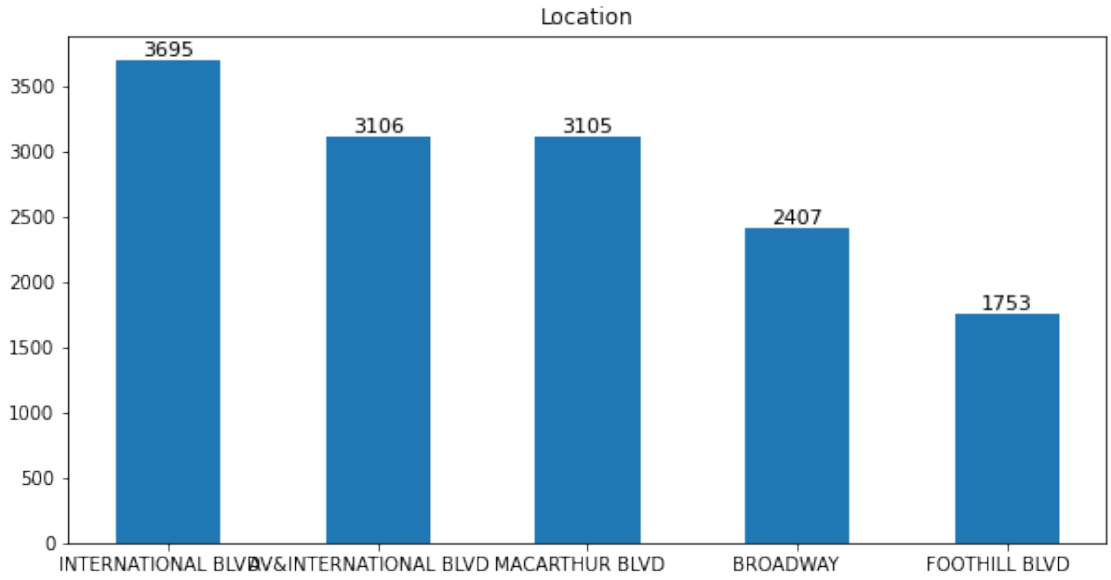

Area Id
5031
3898
5000
4000
3000
2000
1000
0
1.0
2.0

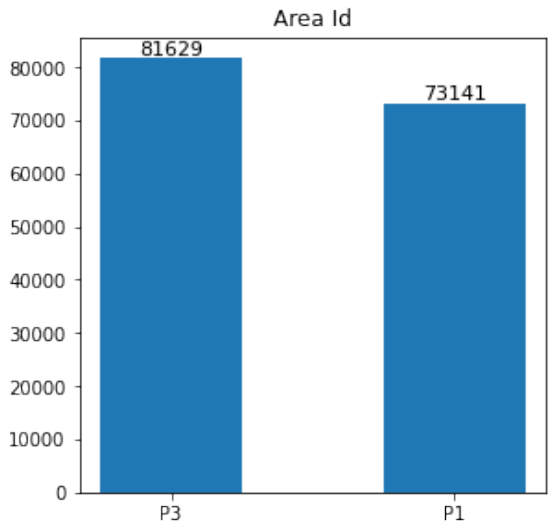

Incident Type Id
17799
13784
11937
9647
8894
17500
15000
12500
10000
7500
5000
2500
0
933R
SECCK
415
911H
10851

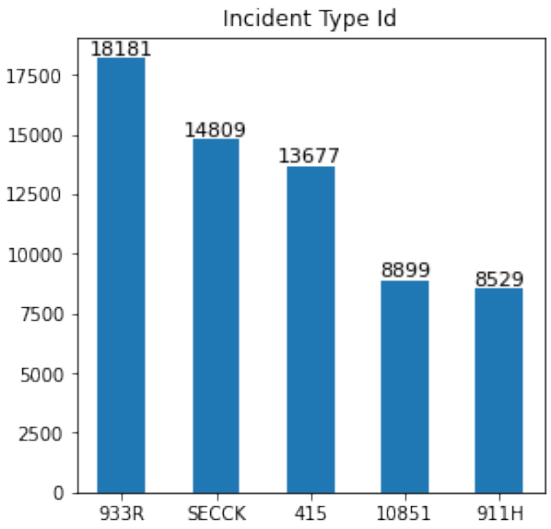

Incident Type Description
17799
13784
9647
8894
7002
17500
15000
12500
10000
7500
5000
2500
0
ALARM-RINGER
SECURITY CHECK
911 HANG-UP
STOLEN VEHICLE
MENTALLY ILL

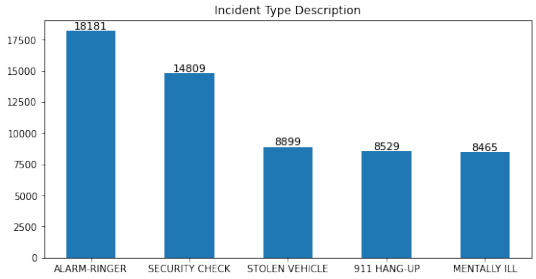

Event Number
1
1
1
1
1
1.0
0.8
0.6
0.4
0.2
0.0
LOP140402000202
LOP140608000806
LOP141227000661
LOP140816000475
LOP140403001110

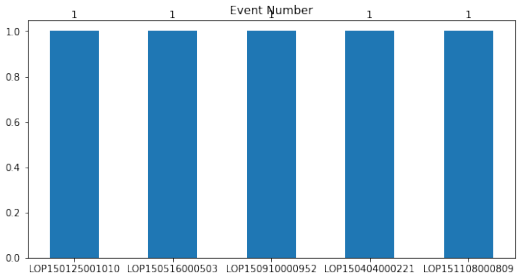
records-for-2015.csv 的频数聚合分析：



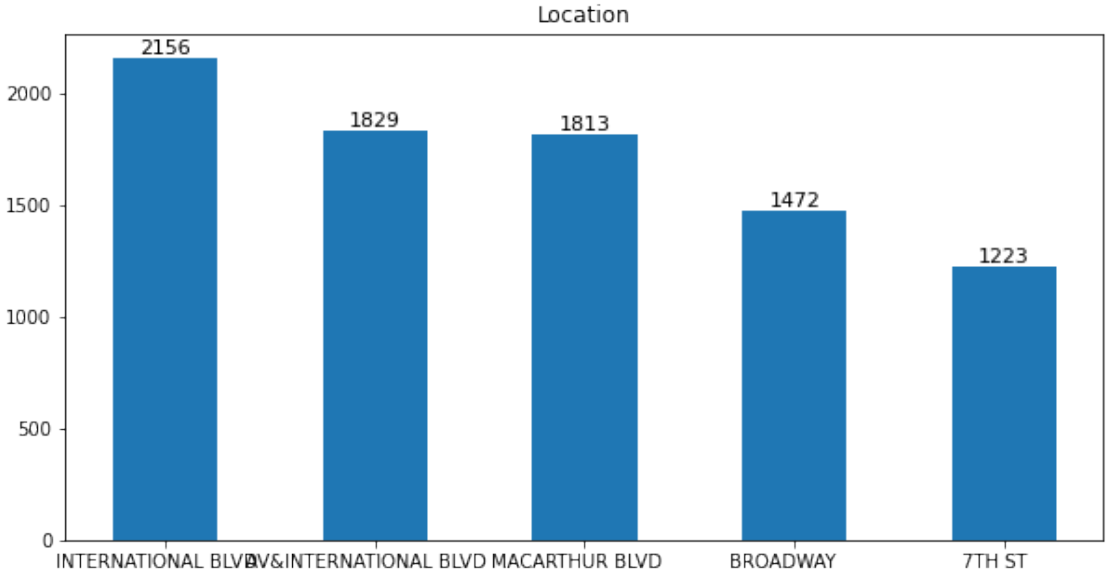


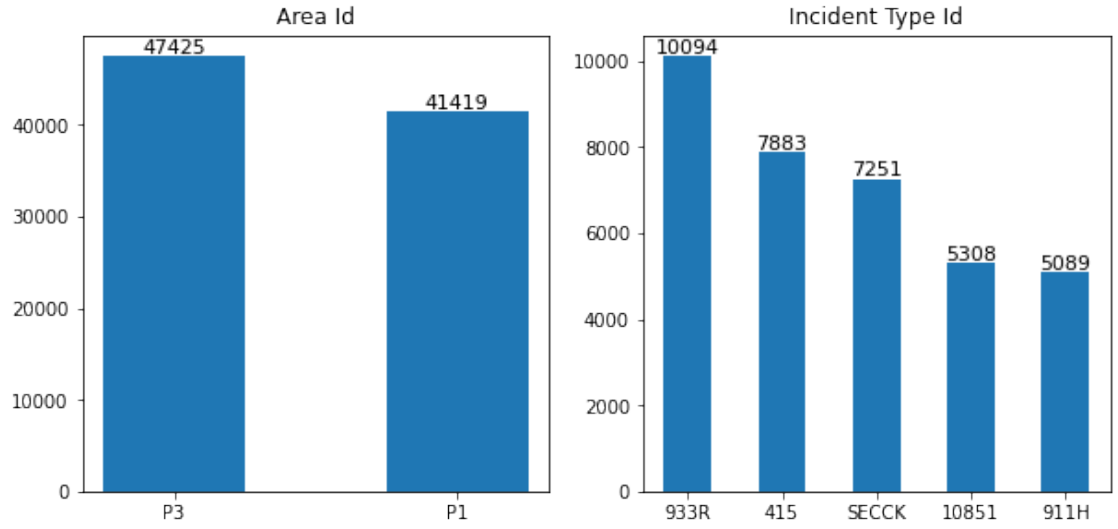


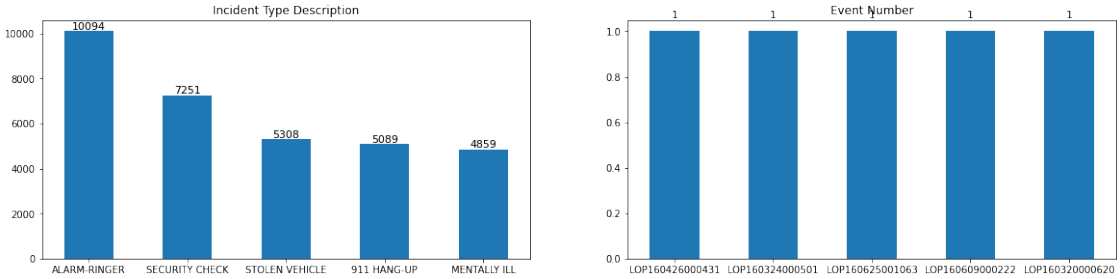




records-for-2016.csv 的频数聚合分析:







[54]:
```
# 五数概括 Minimum（最小值）、Q1、Median（中位数、）、Q3、Maximum（最大值）

import pandas as pd
import matplotlib.pyplot as plt
from scipy.spatial.distance import pdist
from math import ceil
import numpy as np
%matplotlib inline

data1_wine_review = pd.read_csv("winemag-data_first150k.csv", encoding="utf-8")[
    ["price", "points"]]
data2_wine_review = pd.read_csv("winemag-data-130k-v2.csv", encoding="utf-8")[
```

```
    ["price", "points"]]
data_all_wine = [data1_wine_review, data2_wine_review]
wine_name_list = ["winemag-data_first150k.csv", "winemag-data-130k-v2.csv"]


def fiveNumber(nums):

    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: "+str(Minimum) +',  ' + 'Q1: ' +str(Q1) +',  ' + 'Median:␣
↪' +str(Median) + ',  ' + 'Q3: ' +str(Q3) + ',  ' + 'Maximum: ' +str(Maximum)

print("Wine Reviews 数据集的数值属性有: points 和 price")
i = 0
attribute_list = ["price", "points"]
for datax in data_all_wine:
    print("\n" + wine_name_list[i] + "的五数概括: ")
    d = pd.DataFrame(data=datax["price"]) #price 属性有空值, 转成 DataFrame 格式
处理该空值
    d=d.dropna(axis=0, how='any')
    d=d.values
    d=d.flatten()
    m=fiveNumber(d)
    points_five1=fiveNumber(datax["points"])
    print("points 缺省值数量: "+str(datax[["points"]].isnull().sum()[0])+ ";   五
数概括: "+str(points_five1)+\
          "\nprice 缺省值数量: "+str(datax[["price"]].isnull().sum()[0])+";   五
数概括: "+str(m))
```

```
    i+=1
```

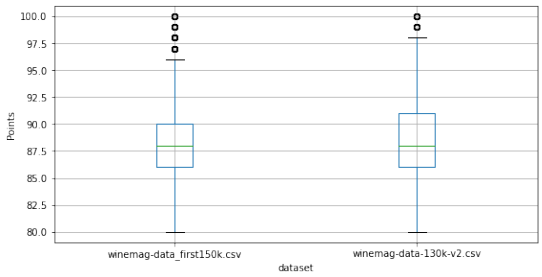
```
Wine Reviews 数据集的数值属性有: points 和 price

winemag-data_first150k.csv 的五数概括:
points 缺省值数量: 0;   五数概括: Minimum: 80,  Q1: 86.0,  Median: 88.0,  Q3: 90.
↪0,  Maximum:
100
price 缺省值数量: 13695;   五数概括: Minimum: 4.0,  Q1: 16.0,  Median: 24.0,  Q3:␣
↪40.0,
Maximum: 2300.0

winemag-data-130k-v2.csv 的五数概括:
points 缺省值数量: 0;   五数概括: Minimum: 80,  Q1: 86.0,  Median: 88.0,  Q3: 91.
↪0,  Maximum:
100
price 缺省值数量: 8996;   五数概括: Minimum: 4.0,  Q1: 17.0,  Median: 25.0,  Q3:␣
↪42.0,
Maximum: 3300.0
```

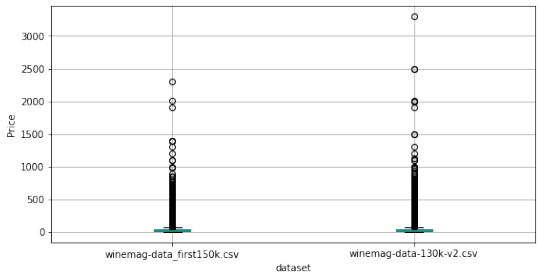
[66]:
```
# 使用直方图、盒图等检查数据分布及离群点

import pandas as pd
import matplotlib.pyplot as plt
from scipy.spatial.distance import pdist
from math import ceil
import numpy as np
%matplotlib inline

data1_wine_review = pd.read_csv("winemag-data_first150k.csv", encoding="utf-8")[
    ["price", "points"]]
data2_wine_review = pd.read_csv("winemag-data-130k-v2.csv", encoding="utf-8")[
    ["price", "points"]]
data_all_wine = [data1_wine_review, data2_wine_review]
wine_name_list = ["winemag-data_first150k.csv", "winemag-data-130k-v2.csv"]

print("points 和 price 的盒图:")
```

```
plt.figure(figsize=(20,10))
ax1 = plt.subplot(2,2,1)
ax2 = plt.subplot(2,2,2)

plt.sca(ax1)
point_box = pd.DataFrame({"winemag-data_first150k.csv":
 ↪data1_wine_review[["points"][0]],
                     "winemag-data-130k-v2.csv":
 ↪data2_wine_review[["points"][0]]})
point_box.boxplot()
plt.ylabel("Points")
plt.xlabel("dataset")

plt.sca(ax2)
price_box = pd.DataFrame({"winemag-data_first150k.csv":
 ↪data1_wine_review[["price"][0]],
                     "winemag-data-130k-v2.csv":
 ↪data2_wine_review[["price"][0]]})
price_box.boxplot()
plt.ylabel("Price")
plt.xlabel("dataset")

plt.show()
point1_out=[]
point2_out=[]



print("points 和 price 的直方图:")
plt.figure(figsize=(20,10))
ax3 = plt.subplot(2,2,1)
ax4 = plt.subplot(2,2,2)
ax5 = plt.subplot(2,2,3)
ax6 = plt.subplot(2,2,4)

plt.sca(ax3)
```

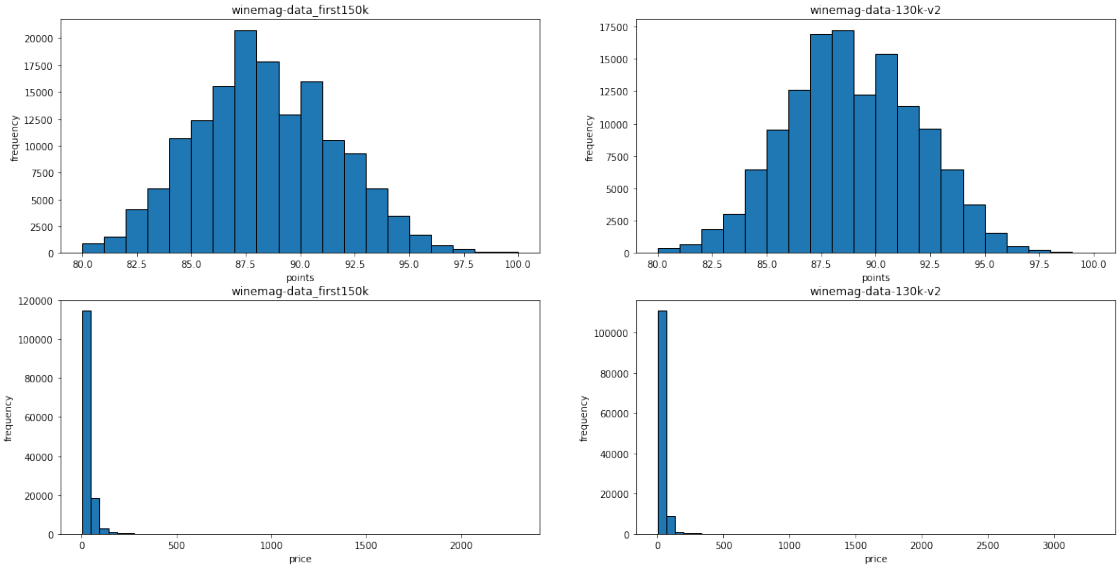
```
plt.hist(data1_wine_review[["points"][0]], bins=20,  edgecolor = 'black',\
         histtype='bar', align='mid', orientation='vertical')
plt.xlabel('points')
plt.ylabel('frequency')
plt.title('winemag-data_first150k')

plt.sca(ax4)
plt.hist(data2_wine_review[["points"][0]], bins=20,  edgecolor = 'black',\
         histtype='bar', align='mid', orientation='vertical')
plt.xlabel('points')
plt.ylabel('frequency')
plt.title('winemag-data-130k-v2')

plt.sca(ax5)
plt.hist(data1_wine_review[["price"][0]], bins=50, edgecolor = 'black',\
         histtype='bar', align='mid', orientation='vertical')
plt.xlabel('price')
plt.ylabel('frequency')
plt.title('winemag-data_first150k')

plt.sca(ax6)
plt.hist(data2_wine_review[["price"][0]], bins=50,  edgecolor = 'black',\
         histtype='bar', align='mid', orientation='vertical')
plt.xlabel('price')
plt.ylabel('frequency')
plt.title('winemag-data-130k-v2')
plt.show()
```

points 和 price 的盒图:





points 和 price 的直方图:



[131]:
```
# 数据缺失的处理

import pandas as pd
import matplotlib.pyplot as plt
from scipy.spatial.distance import pdist
from math import ceil
import numpy as np
%matplotlib inline

data1_wine_review = pd.read_csv("winemag-data_first150k.csv", encoding="utf-8")
data2_wine_review = pd.read_csv("winemag-data-130k-v2.csv", encoding="utf-8")

data_all_wine = [data1_wine_review, data2_wine_review]
wine_name_list = ["winemag-data_first150k.csv", "winemag-data-130k-v2.csv"]

def fiveNumber(nums):
    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: "+str(Minimum) +',  ' + 'Q1: ' +str(Q1) +',  ' + 'Median:␣
 ↪' +str(Median) + ',  ' + 'Q3: ' +str(Q3) + ',  ' + 'Maximum: ' +str(Maximum)

# 剔除缺失值

data1_1=data1_wine_review.dropna(axis=0, how='any')
data1_2=data2_wine_review.dropna(axis=0, how='any')

print("\n剔除缺失值:")
plt.figure(figsize=(20,10))
ax1 = plt.subplot(2,2,1)
ax2 = plt.subplot(2,2,2)
ax3 = plt.subplot(2,2,3)
ax4 = plt.subplot(2,2,4)

plt.sca(ax1)
box = pd.DataFrame({"winemag-data_first150k.csv":
 ↪data1_wine_review[["points"][0]], "winemag-data-130k-v2.csv":
 ↪data2_wine_review[["points"][0]]})
box.boxplot()
plt.ylabel("Original points")
plt.xlabel("dataset")

plt.sca(ax2)
box = pd.DataFrame({"winemag-data_first150k.csv":data1_1["points"],␣
 ↪"winemag-data-130k-v2.csv":data1_2["points"]})
box.boxplot()
```

```
plt.ylabel("Delete NaN points")
plt.xlabel("dataset")

plt.sca(ax3)
box = pd.DataFrame({"winemag-data_first150k.csv":data1_wine_review["price"],␣
 ↪"winemag-data-130k-v2.csv":data2_wine_review["price"]})
box.boxplot()
plt.ylabel("Original price")
plt.xlabel("dataset")

plt.sca(ax4)
box = pd.DataFrame({"winemag-data_first150k.csv":data1_1["price"],␣
 ↪"winemag-data-130k-v2.csv":data1_2["price"]})
box.boxplot()
plt.ylabel("Delete NaN price")
plt.xlabel("dataset")

plt.show()



# 高频值填充
print("\n用最高频率值来填补缺失值:")
frequency1=data1_wine_review["price"].mode()
frequency2=data2_wine_review["price"].mode()
data2_1=data1_wine_review["price"].fillna(frequency1[0])
data2_2=data2_wine_review["price"].fillna(frequency2[0])

plt.figure(figsize=(20,10))
ax1 = plt.subplot(2,2,1)
ax2 = plt.subplot(2,2,2)

plt.sca(ax1)
# hist(data1_wine_review["price"], 50, "price", "frequency", "original")
plt.hist(data1_wine_review["price"], bins=50,  edgecolor = 'black',␣
 ↪histtype='bar', align='mid', orientation='vertical')
```

```
plt.xlabel("price")
plt.ylabel("frequency")
plt.title("winemag-data_first150k.csv original")

plt.sca(ax2)
# hist(data2_1, 50, "price", "frequency", "add frequency")
plt.hist(data2_1, bins=50,  edgecolor = 'black', histtype='bar', align='mid',␣
 ↪orientation='vertical')
plt.xlabel("price")
plt.ylabel("frequency")
plt.title("winemag-data_first150k.csv add frequency")
plt.show()

print("由于样本间数据差异大，直方图中 0~60 范围的数据上升到了 120000 以上，可以看其五数概括变化:")
print(fiveNumber(data2_1))
print("其均值和 Q3 从原来 24, 40 变成了 22, 38")

plt.figure(figsize=(20,10))
ax3 = plt.subplot(2,2,1)
ax4 = plt.subplot(2,2,2)

plt.sca(ax3)
plt.hist(data2_wine_review["price"], bins=50,  edgecolor = 'black',␣
 ↪histtype='bar', align='mid', orientation='vertical')
plt.xlabel("price")
plt.ylabel("frequency")
plt.title("winemag-data-130k-v2.csv original")

plt.sca(ax4)
plt.hist(data2_2, bins=50,  edgecolor = 'black', histtype='bar', align='mid',␣
 ↪orientation='vertical')
plt.xlabel("price")
plt.ylabel("frequency")
plt.title("winemag-data-130k-v2.csv add frequency")
plt.show()
```

```
print("同样在 0~60 的数据上升的比较多，从不到 12 万到接近 12 万，看其五数概括变化:
 ↪")
print(fiveNumber(data2_2))
print("其 Q1 和 Q3 从原来 17, 42 变成了 18, 40")



# 属性间的相关性填充，由于数值只有 point 和 price，以这两者考虑相似性
print("\n通过属性的相关关系来填补缺失值:")
print("\n采用余弦相似性和皮尔逊相关系数进行相关性评价，无需进行数据标准化:")
data4_1=data1_wine_review
data4_2=data2_wine_review
data4_1=data4_1.dropna(axis=0, how='any')
data4_2=data4_2.dropna(axis=0, how='any')
points1=data4_1["points"]
price1=data4_1["price"]
points2=data4_2["points"]
price2=data4_2["price"]
cos1 = np.vstack([points1,price1])
p1 = 1 - pdist(cos1,'cosine')
cos2 = np.vstack([points2,price2])
p2 = 1 - pdist(cos2,'cosine')
print("winemag-data_first150k.csv: PLCC="+str(points1.
 ↪corr(price1,method="pearson"))+"  Cosine similarity="+str(p1))
print("winemag-data-130k-v2.csv: PLCC="+str(points2.
 ↪corr(price2,method="pearson"))+"  Cosine similarity="+str(p2))
print("可见相关性不是很强，但是度数越高价格越贵，因此但可以进行一些简单的填充")
# 以每个度数的中值来填充结果
xx = data1_wine_review[["points", "price"]].groupby(by="points").median()
xx=xx.values
xx=xx.flatten()
yy = data2_wine_review[["points", "price"]].groupby(by="points").median()
yy=yy.values
yy=yy.flatten()

data_add1=data1_wine_review
```

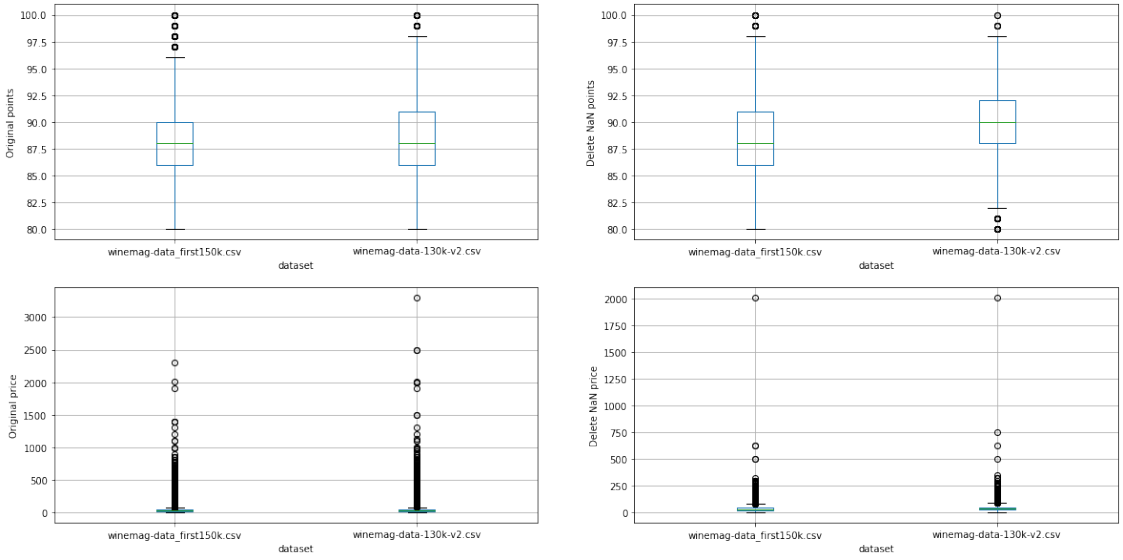
```
data_add2=data2_wine_review
dataadd_g1=pd.DataFrame()
dataadd_g2=pd.DataFrame()
for i in range(80,101):
    data_ad1=data_add1.loc[data_add1['points'].isin([i])].fillna(xx[i-80])
    data_ad2=data_add2.loc[data_add1['points'].isin([i])].fillna(yy[i-80])
    if(i==80):
        data_add_g1=data_ad1
        data_add_g2=data_ad2
    else:
        data_add_g1=pd.concat([data_add_g1,data_ad1],axis=0)
        data_add_g2=pd.concat([data_add_g2,data_ad2],axis=0)
prices1=data_add_g1[["price"][0]]
prices2=data_add_g2[["price"][0]]

plt.figure(figsize=(20,10))
ax31 = plt.subplot(1,3,1)
ax32 = plt.subplot(1,3,2)
ax33 = plt.subplot(1,3,3)

plt.sca(ax31)
box = pd.DataFrame({"winemag-data_first150k.csv":prices1, "winemag-data-130k-v2.
 ↪csv":prices2})
box.boxplot()
plt.ylabel("prices")
plt.xlabel("dataset")

plt.sca(ax32)
plt.hist(prices1, bins=50,  edgecolor = 'black', histtype='bar', align='mid',␣
 ↪orientation='vertical')
plt.xlabel("price")
plt.ylabel("frequency")
plt.title("winemag-data_first150k.csv relative filling")

plt.sca(ax33)
```

```
plt.hist(prices2, bins=50,  edgecolor = 'black', histtype='bar', align='mid',␣
 ↪orientation='vertical')
plt.xlabel("price")
plt.ylabel("frequency")
plt.title("winemag-data-130k-v2.csv relative filling")

plt.show()

# hist(prices1,50,"prices","frequency","winemag-data_first150k.csv relative␣
 ↪filling")
print("winemag-data_first150k.csv 通过属性的相关关系填充后，五数概
括"+str(fiveNumber((prices1.values).flatten())))
# hist(prices2,50,"prices","frequency","winemag-data-130k-v2.csv relative␣
 ↪filling")
print("winemag-data-130k-v2.csv 通过属性的相关关系填充后，  五数概
括"+str(fiveNumber((prices2.values).flatten())))
print("相关性填充根据酒的度数和价格之间的关系进行，某种酒的 NaN 值取自该酒所在度数
的所有酒价格的中值;\
        \n从直方图中可以看到价格在 0~60 之间的数量较填充前,winemag-data_first150k
突破了 12 万，winemag-data-130k-v2.csv 接近 12 万，同时 winemag-data_first150k 还
填充了一些价格为 1000 的数据\
        五数概括并没有明显的变化，只有 winemag-data-130k-v2.csv 的 Q3 从 42 变为了
41.")



# 对象之间的相似性填充
print("\n通过数据对象之间的相似性来填补缺失值:")
print("分析可知，缺失的 price 多为 France 和 Italy 两个国家的，因此可用相应国家的众
数进行相应填充。")
data3_1=data1_wine_review
data3_2=data2_wine_review
data3_3=data3_1[data3_1["country"]=="France"]
data3_4=data3_1[data3_1["country"]=="Italy"]
```

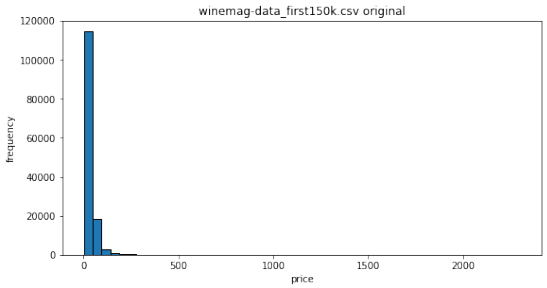
```
data3_5=data3_2[data3_2["country"]=="France"]
data3_6=data3_2[data3_2["country"]=="Italy"]

frequency_France1=data3_3["price"].mode()
frequency_Italy1=data3_4["price"].mode()

frequency_France2=data3_5["price"].mode()
frequency_Italy2=data3_6["price"].mode()
print("winemag-data_first150k.csv: France mode:"+str(frequency_France1[0])+" ␣
 ↪Italy mode:"+str(frequency_Italy1[0]))
print("winemag-data-130k-v2.csv: France mode:"+str(frequency_France2[0])+" ␣
 ↪Italy mode:"+str(frequency_Italy2[0]))

print("由此可见，这两个数据集上两个国家的 price 众数都为 20.0，因此可以用 20.0 来进
行填充，填充后的对比结果为：")

data3_1_fill=data1_wine_review["price"].fillna(frequency_France1[0])
data3_2_fill=data2_wine_review["price"].fillna(frequency_France2[0])

plt.figure(figsize=(20,10))
ax21 = plt.subplot(2,2,1)
ax22 = plt.subplot(2,2,2)

plt.sca(ax21)
plt.hist(data1_wine_review["price"], bins=50,  edgecolor = 'black',␣
 ↪histtype='bar', align='mid', orientation='vertical')
plt.xlabel("price")
plt.ylabel("frequency")
plt.title("winemag-data_first150k.csv original")

plt.sca(ax22)
plt.hist(data3_1_fill, bins=50,  edgecolor = 'black', histtype='bar',␣
 ↪align='mid', orientation='vertical')
plt.xlabel("price")
plt.ylabel("frequency")
plt.title("winemag-data_first150k.csv after filling")
```

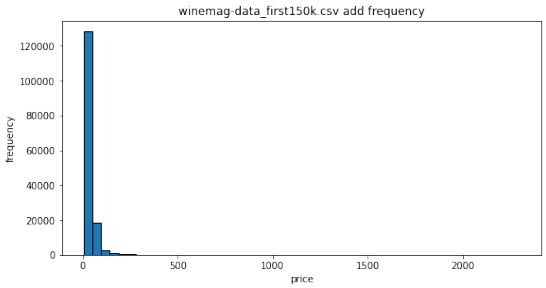
```
plt.show()



plt.figure(figsize=(20,10))
ax221 = plt.subplot(2,2,1)
ax222 = plt.subplot(2,2,2)

plt.sca(ax221)
plt.hist(data2_wine_review["price"], bins=50,  edgecolor = 'black',␣
 ↪histtype='bar', align='mid', orientation='vertical')
plt.xlabel("price")
plt.ylabel("frequency")
plt.title("winemag-data-130k-v2.csv original")

plt.sca(ax222)
plt.hist(data3_2_fill, bins=50,  edgecolor = 'black', histtype='bar',␣
 ↪align='mid', orientation='vertical')
plt.xlabel("price")
plt.ylabel("frequency")
plt.title("winemag-data-130k-v2.csv after filling")
plt.show()
```

剔除缺失值:



用最高频率值来填补缺失值：

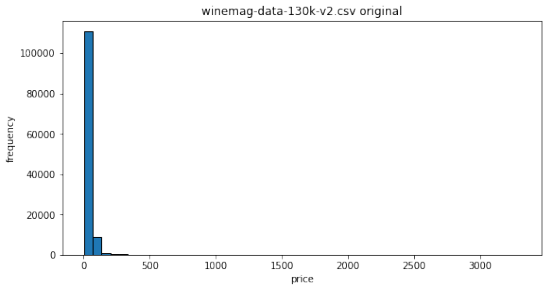


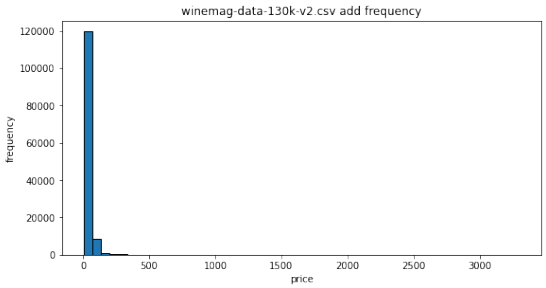


由于样本间数据差异大，直方图中 0~60 范围的数据上升到了 120000 以上，可以看其五数概括变化：

Minimum: 4.0, Q1: 16.0, Median: 22.0, Q3: 38.0, Maximum: 2300.0

其均值和 Q3 从原来 24，40 变成了 22，38





同样在 0~60 的数据上升的比较多，从不到 12 万到接近 12 万，看其五数概括变化：

Minimum: 4.0, Q1: 18.0, Median: 25.0, Q3: 40.0, Maximum: 3300.0

其 Q1 和 Q3 从原来 17，42 变成了 18，40

通过属性的相关关系来填补缺失值：

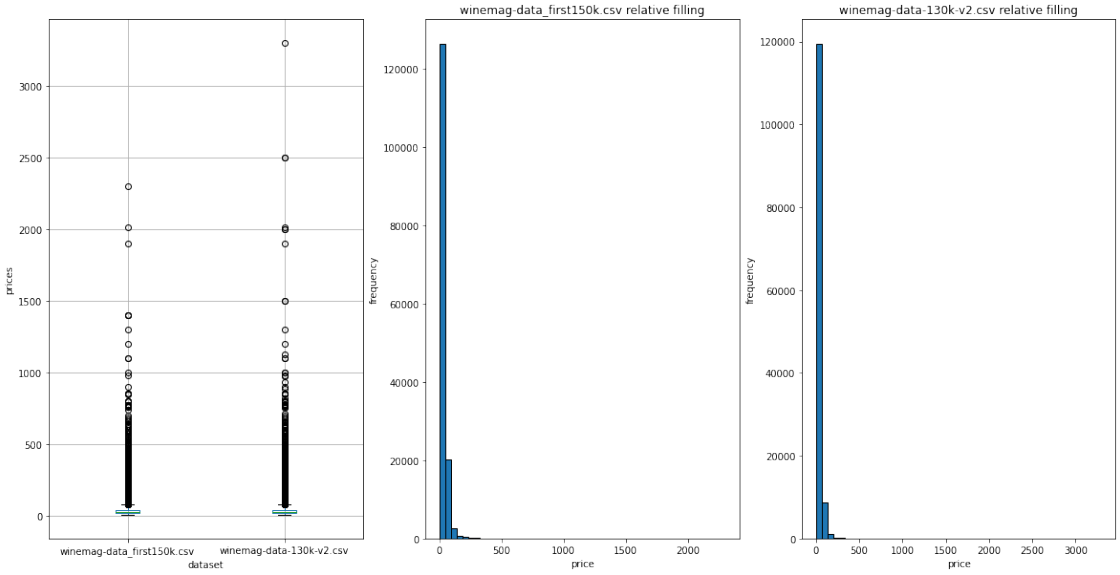
采用余弦相似性和皮尔逊相关系数进行相关性评价，无需进行数据标准化：

winemag-data_first150k.csv: PLCC=0.43562770582555 Cosine

similarity=[0.82398976]

winemag-data-130k-v2.csv: PLCC=0.3946263507693823 Cosine similarity=[0.8227089]

可见相关性不是很强，但是度数越高价格越贵，因此但可以进行一些简单的填充



winemag-data_first150k.csv 通过属性的相关关系填充后，五数概括 Minimum: 4.0， Q1:␣
↪16.0， Median:

24.0， Q3: 40.0， Maximum: 2300.0

winemag-data-130k-v2.csv 通过属性的相关关系填充后， 五数概括 Minimum: 4.0， Q1: 17.
↪0， Median:

25.0， Q3: 41.0， Maximum: 3300.0

相关性填充根据酒的度数和价格之间的关系进行，某种酒的 NaN 值取自该酒所在度数的所有酒价格的中值；

从直方图中可以看到价格在 0~60 之间的数量较填充前,winemag-data_first150k 突破了 12 万，winemag-

data-130k-v2.csv 接近 12 万，同时 winemag-data_first150k 还填充了一些价格为 1000 的数据

五数概括并没有明显的变化，只有 winemag-data-130k-v2.csv 的 Q3 从 42 变为了 41.

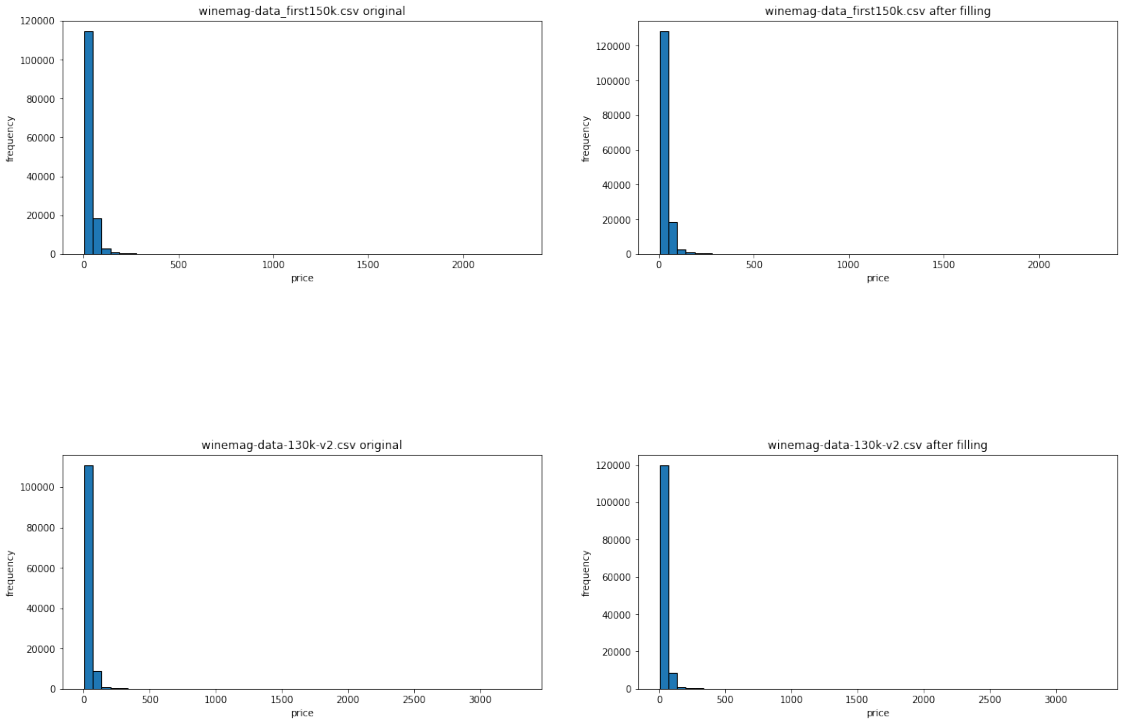
通过数据对象之间的相似性来填补缺失值:

分析可知，缺失的 price 多为 France 和 Italy 两个国家的，因此可用相应国家的众数进行相应填充。

winemag-data_first150k.csv: France mode:20.0 Italy mode:20.0

winemag-data-130k-v2.csv: France mode:20.0 Italy mode:20.0

由此可见，这两个数据集上两个国家的 price 众数都为 20.0，因此可以用 20.0 来进行填充，填充后的对比结果为：



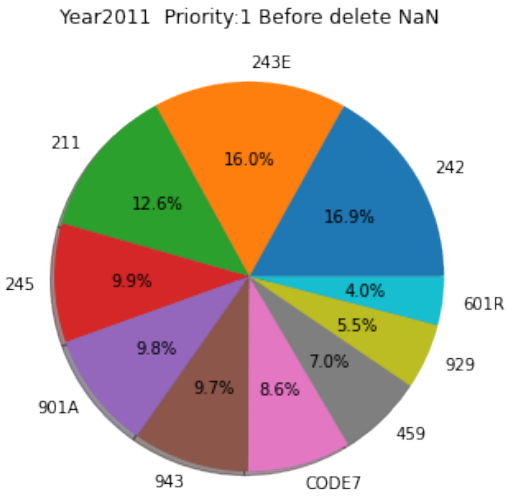
[130]:
```
# 数据缺失的处理
import pandas as pd
import matplotlib.pyplot as plt
from scipy.spatial.distance import pdist
from math import ceil
import numpy as np
%matplotlib inline

data1 = pd.read_csv("records-for-2011.csv",encoding="utf-8")
data2 = pd.read_csv("records-for-2012.csv",encoding="utf-8")
data3 = pd.read_csv("records-for-2013.csv",encoding="utf-8")
data4 = pd.read_csv("records-for-2014.csv",encoding="utf-8")
data5 = pd.read_csv("records-for-2015.csv",encoding="utf-8")
```

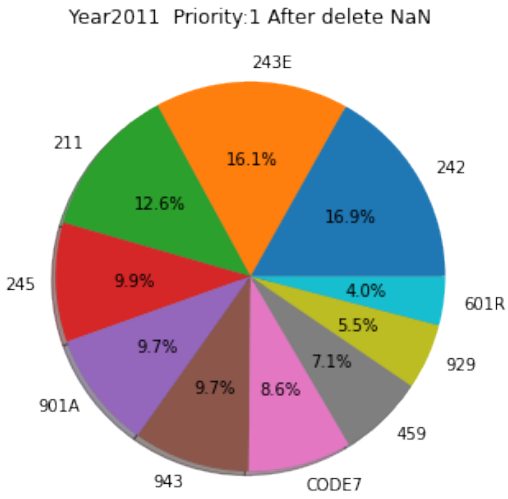
```
data6 = pd.read_csv("records-for-2016.csv",encoding="utf-8")
data_all_crime = [data1, data2, data3, data4, data5, data6]

# 剔除空属性值
print("\n剔除空属性值: ")
year=2011
for data in data_all_crime:
    print(str(year)+":的犯罪记录情况如下")
    data_id=data[["Incident Type Id","Priority","Area Id","Beat","Incident Type␣
 ↪Description","Closed Time"]]
    data_id=data_id.dropna(axis=0,how='any')   # 剔除空属性值
    data_id=data_id[["Incident Type Id","Priority"]]
    data_p1=data_id[data_id["Priority"].isin([1])]
    data_p1=data_p1["Incident Type Id"].value_counts(sort=True)

    data_id2=data[["Incident Type Id","Priority","Area Id","Beat","Incident␣
 ↪Type Description","Closed Time"]]
    data_id2=data_id2.dropna(axis=0,how='all')   # 剔除全空
    data_id2=data_id2[["Incident Type Id","Priority"]]
    data_p2=data_id2[data_id2["Priority"].isin([1])]
    data_p2=data_p2["Incident Type Id"].value_counts(sort=True)

    data_p1=data_p1.head(10)
    crime_type1=data_p1.index.tolist()
    crime_num1=data_p1.values

    data_p2=data_p2.head(10)
    crime_type2=data_p2.index.tolist()
    crime_num2=data_p2.values


    plt.figure(figsize=(12,12))
    ax1 = plt.subplot(2,2,1)
    ax2 = plt.subplot(2,2,2)

    plt.sca(ax1)
```

```
    plt.pie(crime_num2,labels=crime_type2,autopct='%1.1f%%',shadow="true")# 绘制饼图
    plt.title("Year"+str(year)+"  Priority:1 Before delete NaN")

    plt.sca(ax2)
    plt.pie(crime_num1,labels=crime_type1,autopct='%1.1f%%',shadow="true")# 绘制饼图
    plt.title("Year"+str(year)+"  Priority:1 After delete NaN")

    plt.show()
    year+=1
print("就犯罪种类前 10 占比中的优先级 1 而言，其删除影响不大，每年导致的误差都大致在 2%以内，其中绝大多数都在百分之零点几的范围内波动")
```

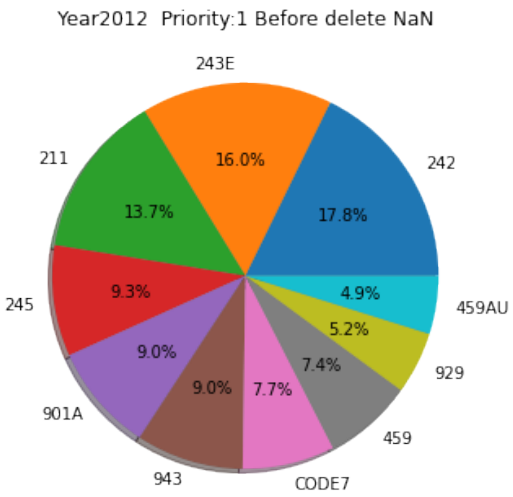
剔除空属性值：

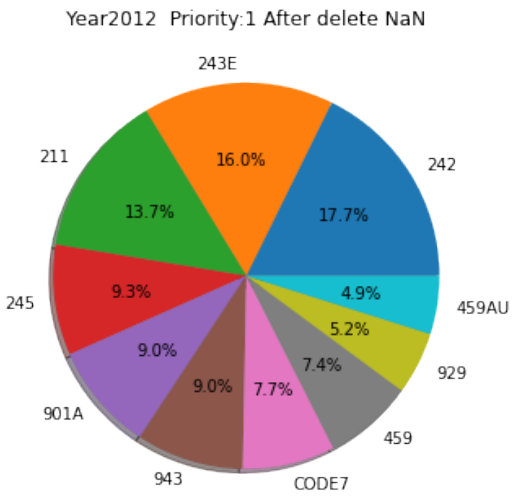
2011:的犯罪记录情况如下



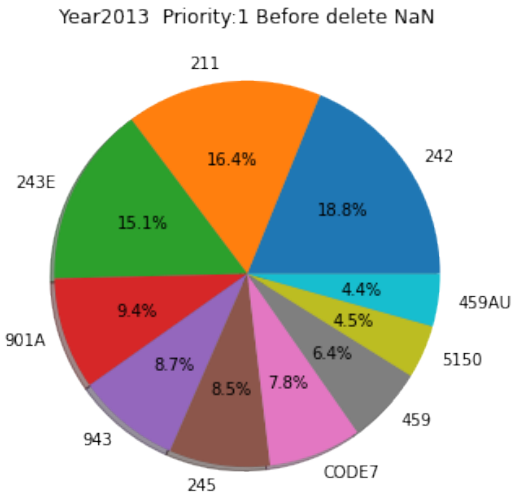


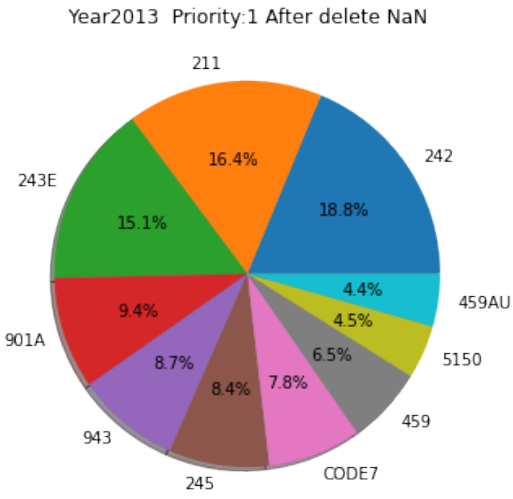
2012:的犯罪记录情况如下



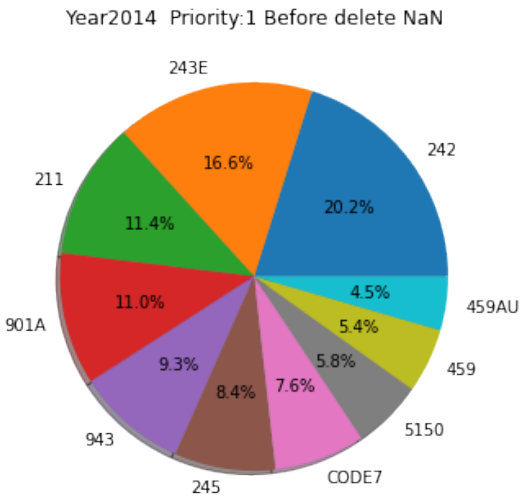


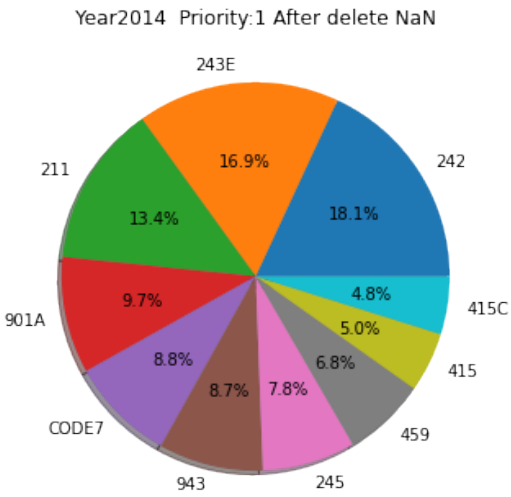
2013:的犯罪记录情况如下



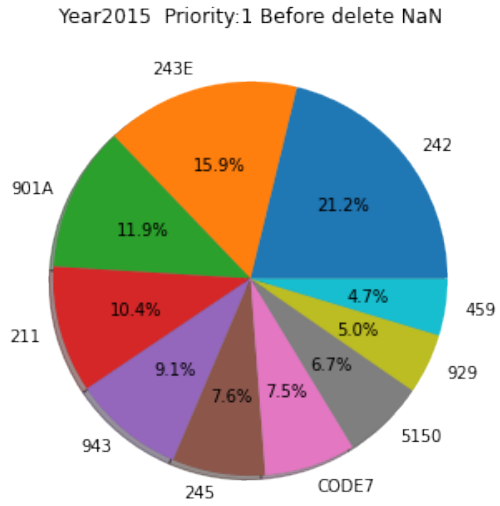


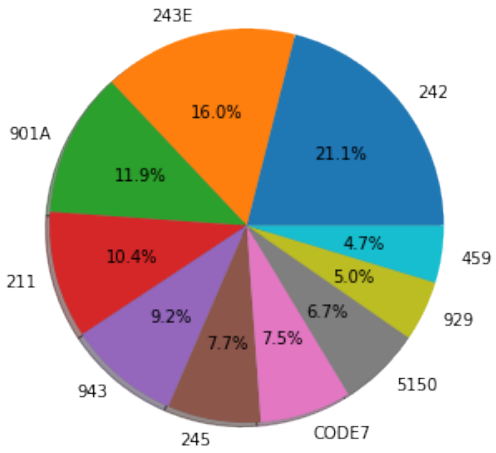
2014:的犯罪记录情况如下



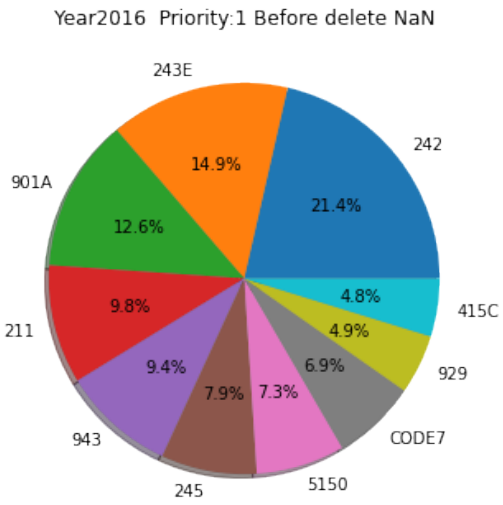


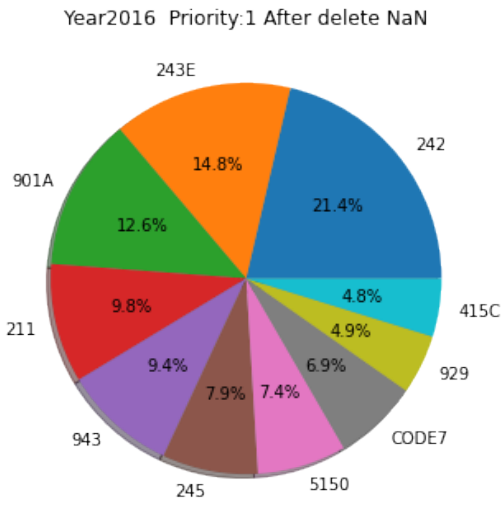
2015:的犯罪记录情况如下





2016:的犯罪记录情况如下





就犯罪种类前 10 占比中的优先级 1 而言，其删除影响不大，每年导致的误差都大致在 2% 以内，其中绝大多数都在百分之零点几的范围内波动

[ ]:

[ ]: