visit-patterns-by-census-block-group analysis

2020 年 5 月 3 日

```
In [1]: import os
        import pandas as pd
        import matplotlib.pyplot as plt
        os.chdir("C:/Users/acer_pc/Downloads/visit-patterns-by-census-block-group")
        data = pd.read_csv("cbg_patterns.csv")
```

读取数据各属性的基本信息。可看出非空元素个数

```
In [2]: data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 220735 entries, 0 to 220734
Data columns (total 13 columns):
census_block_group          220734 non-null float64
date_range_start            220735 non-null int64
date_range_end              220735 non-null int64
raw_visit_count             220629 non-null float64
raw_visitor_count           220629 non-null float64
visitor_home_cbgs           220735 non-null object
visitor_work_cbgs           220735 non-null object
distance_from_home          220518 non-null float64
related_same_day_brand      220735 non-null object
related_same_month_brand    220735 non-null object
top_brands                  220735 non-null object
popularity_by_hour          220735 non-null object
popularity_by_day           220735 non-null object
dtypes: float64(4), int64(2), object(7)
memory usage: 21.9+ MB
```

查看各项数据的独立元素个数

```
In [3]: data.nunique()
```

```
Out[3]: census_block_group          220734
        date_range_start                 1
        date_range_end                   1
        raw_visit_count              93774
        raw_visitor_count            41483
        visitor_home_cbgs           191832
        visitor_work_cbgs           166013
        distance_from_home           70557
        related_same_day_brand       73198
        related_same_month_brand    185558
        top_brands                   98086
        popularity_by_hour          220630
        popularity_by_day           220630
        dtype: int64
```

打印数据前 5 行

```
In [4]: data.head()
```

```
Out[4]:    census_block_group  date_range_start  date_range_end  raw_visit_count  \
        0        1.005951e+10        1538352000      1541030400          75122.0
        1        1.009051e+10        1538352000      1541030400          95649.0
        2        1.047957e+10        1538352000      1541030400          14009.0
        3        1.069040e+10        1538352000      1541030400         128169.0
        4        1.073011e+10        1538352000      1541030400          51453.0

           raw_visitor_count                                  visitor_home_cbgs  \
        0            18314.0  {"010059501003":127,"010059509001":111,"010059...
        1            38942.0  {"010730113021":210,"010090506022":205,"010090...
        2             3039.0              {"010479567011":67,"010479567021":60}
        3            25418.0  {"010690402013":370,"010690402011":322,"010690...
        4             9499.0  {"010090507001":183,"010730113021":167,"010730...

                                          visitor_work_cbgs  distance_from_home  \
        0  {"010059501003":109,"010810407002":62,"0108104...            194724.0
        1  {"010890111001":271,"010730045001":269,"010439...            120587.0
        2                               {"010479567021":52}             67774.0
        3  {"010690402024":313,"010690415004":203,"010450...             42684.0
        4  {"010730045001":140,"010730027001":123,"010730...             18878.0

                                    related_same_day_brand  \
```

```
0  ["Chick-fil-A","mcdonalds","Marathon Petroleum...
1  ["Shell Oil","mcdonalds","Chick-fil-A","Chevron"]
2                                ["Dollar General"]
3  ["Chick-fil-A","Sam's Club","Dollar General","...
4           ["Chevron","Daylight Donuts","walmart"]

                           related_same_month_brand  \
0  ["walmart","mcdonalds","Dollar General","Chick...
1  ["walmart","mcdonalds","Shell Oil","Chick-fil-...
2  ["walmart","Dollar General","mcdonalds","Chevr...
3  ["walmart","Dollar General","mcdonalds","Marat...
4  ["walmart","Chevron","Dollar General","Shell O...

                                        top_brands  \
0      ["CrossFit","Health Mart","Coldwell Banker"]
1                                                []
2                                ["Dollar General"]
3  ["Chick-fil-A","Sam's Club","Olive Garden","mc...
4                           ["Chevron","CrossFit"]

                                  popularity_by_hour  \
0  [2617,2457,2403,2519,2646,3007,3886,7566,5508,...
1  [6556,6325,6222,6355,6586,7350,8568,8099,7378,...
2  [807,790,796,786,851,951,1134,1797,1355,1241,1...
3  [2121,1828,1784,1704,1861,2373,3730,7497,7093,...
4  [3804,3716,3686,3672,3735,4115,4855,5946,4526,...

                                   popularity_by_day
0  {"Monday":12000,"Tuesday":12224,"Wednesday":10...
1  {"Monday":12125,"Tuesday":12984,"Wednesday":12...
2  {"Monday":2314,"Tuesday":2340,"Wednesday":2195...
3  {"Monday":21141,"Tuesday":21143,"Wednesday":17...
4  {"Monday":8402,"Tuesday":8414,"Wednesday":8550...
```

查看数值类型属性的五数概括

```
In [5]: data.describe()
```

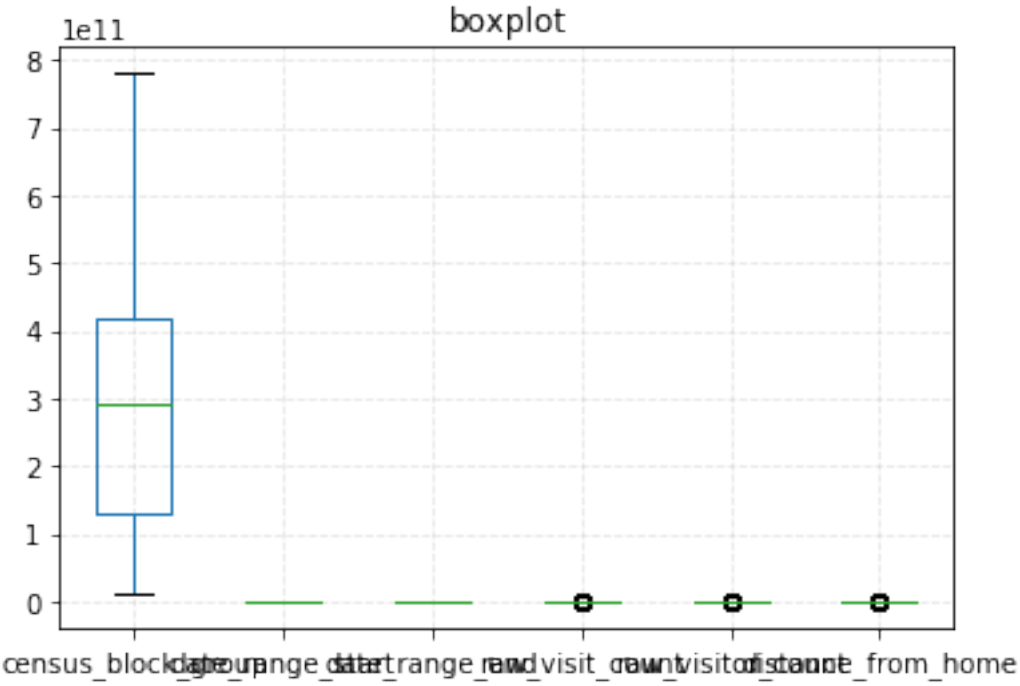
```
Out[5]:        census_block_group  date_range_start  date_range_end  raw_visit_count  \
       count        2.207340e+05      2.207350e+05    2.207350e+05     2.206290e+05
       mean         2.870864e+11      1.538352e+09    1.541030e+09     4.793066e+04
```

std	1.640723e+11	0.000000e+00	0.000000e+00	6.252655e+04
min	1.001020e+10	1.538352e+09	1.541030e+09	6.000000e+01
25%	1.312101e+11	1.538352e+09	1.541030e+09	1.704200e+04
50%	2.901900e+11	1.538352e+09	1.541030e+09	3.064000e+04
75%	4.200349e+11	1.538352e+09	1.541030e+09	5.667800e+04
max	7.803099e+11	1.538352e+09	1.541030e+09	7.179900e+06

	raw_visitor_count	distance_from_home
count	2.206290e+05	2.205180e+05
mean	1.182032e+04	3.511280e+04
std	3.045832e+04	9.973193e+04
min	5.000000e+01	7.060000e+02
25%	3.430000e+03	8.584000e+03
50%	6.541000e+03	1.461400e+04
75%	1.309900e+04	3.139775e+04
max	6.113949e+06	6.297845e+06

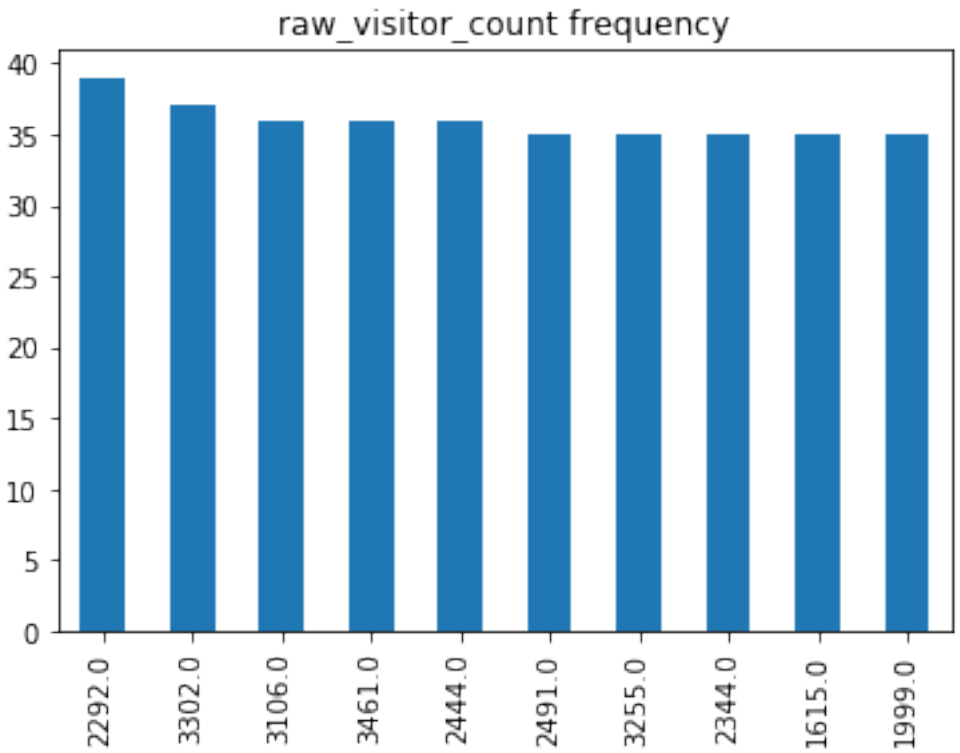
盒图

```
In [15]: df = pd.DataFrame(data)
         df.plot.box(title="boxplot")
         plt.grid(linestyle="--", alpha=0.3)
         plt.show()
```

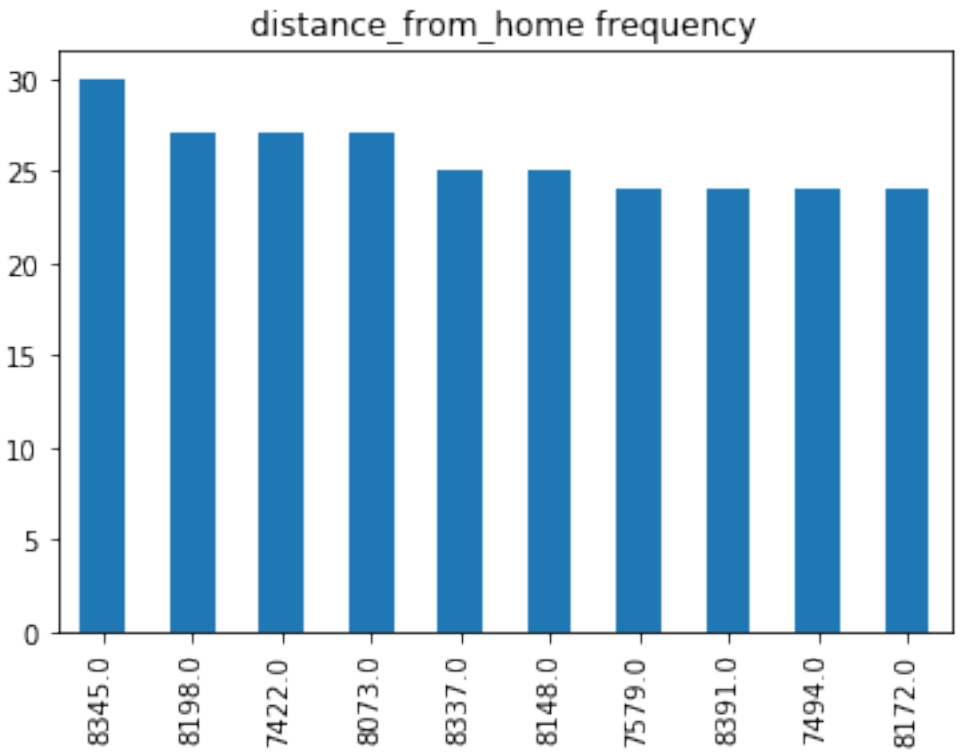


数据可视化（柱状图）

In [22]:
```
df['raw_visitor_count'].value_counts().head(10).plot.bar()
plt.title("raw_visitor_count frequency")
plt.show()
```



In [23]:
```
df['distance_from_home'].value_counts().head(10).plot.bar()
plt.title("distance_from_home frequency")
plt.show()
```



分别对缺失数据进行丢弃、众数填补、基于属性内信息填补

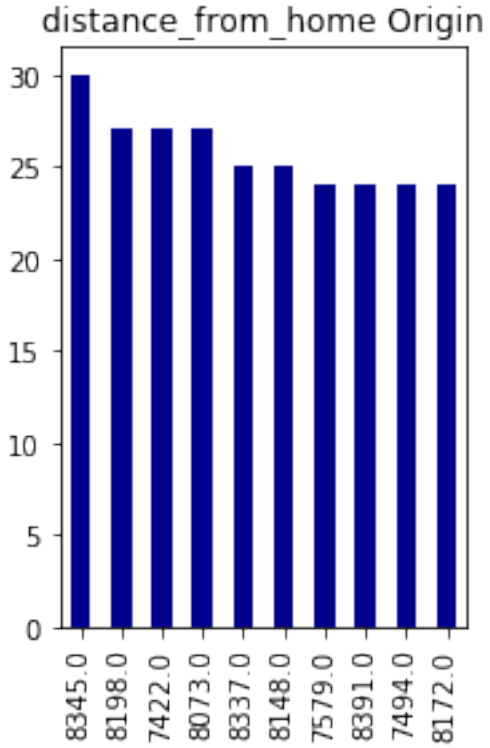
```
In [27]: # NaN data processing
         # for example: distance_from_home frequency
         p1 = df.dropna() #drop

         dis_mode = df['distance_from_home'].mode()
         modes = {'distance_from_home':dis_mode}
         p2 = df.fillna(value=modes)  # filled with mode

         p3 = df['distance_from_home'].fillna(method="ffill") # filled with neighbor's value
```

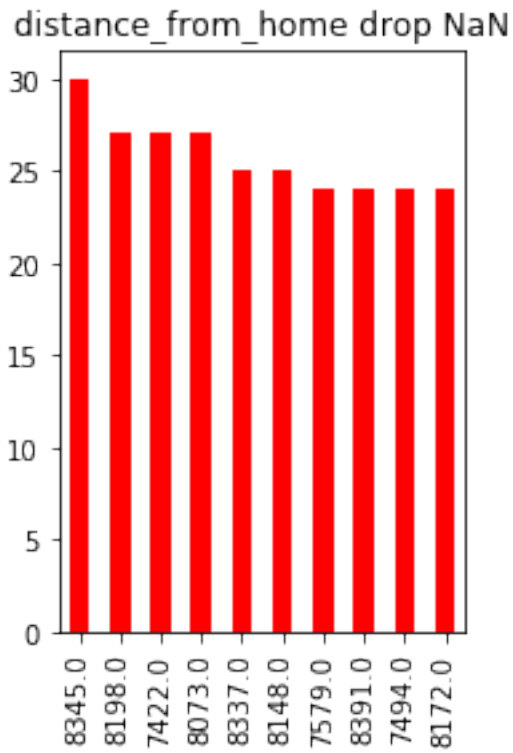
缺失数据处理结果与原始数据可视化对比

```
In [29]: # compare1
         plt.subplot(121)
         ax=df['distance_from_home'].value_counts().head(10).plot.bar(color='DarkBlue')
         plt.title("distance_from_home Origin")
         plt.show()

         plt.subplot(122)
```

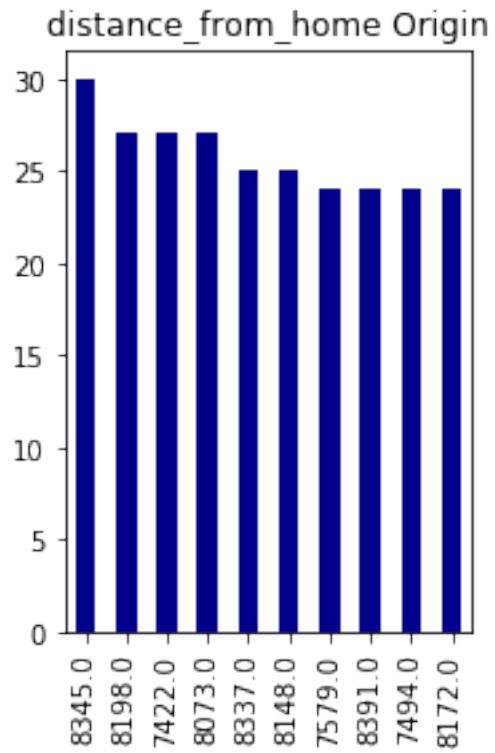
```
bx = p1['distance_from_home'].value_counts().head(10).plot.bar(color='Red')
plt.title("distance_from_home drop NaN")
plt.show()
```

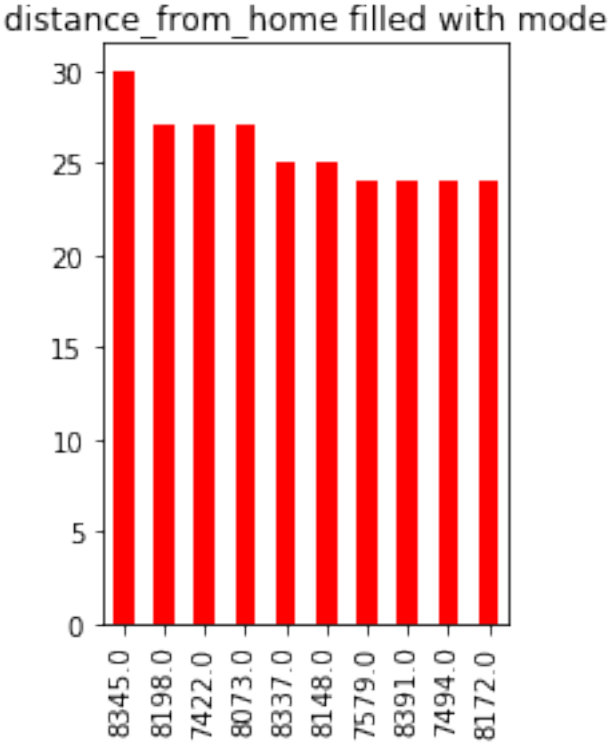
```
D:\anaconda3\lib\site-packages\matplotlib\figure.py:98: MatplotlibDeprecationWarning:
Adding an axes using the same arguments as a previous axes currently reuses the earlier instance.
  "Adding an axes using the same arguments as a previous axes "
```



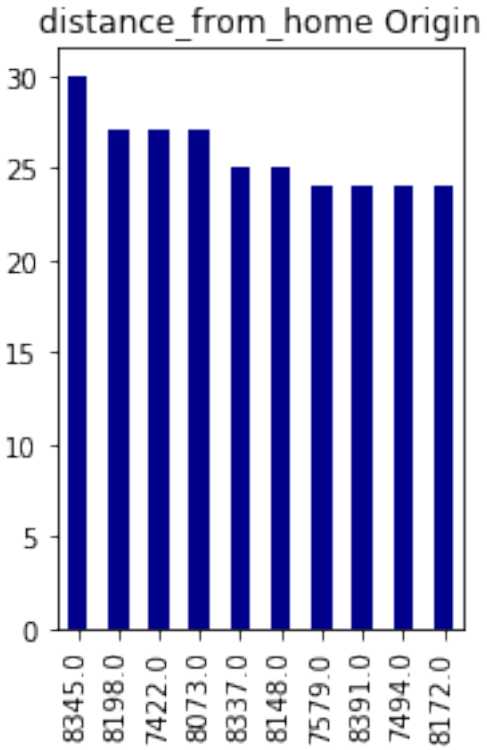


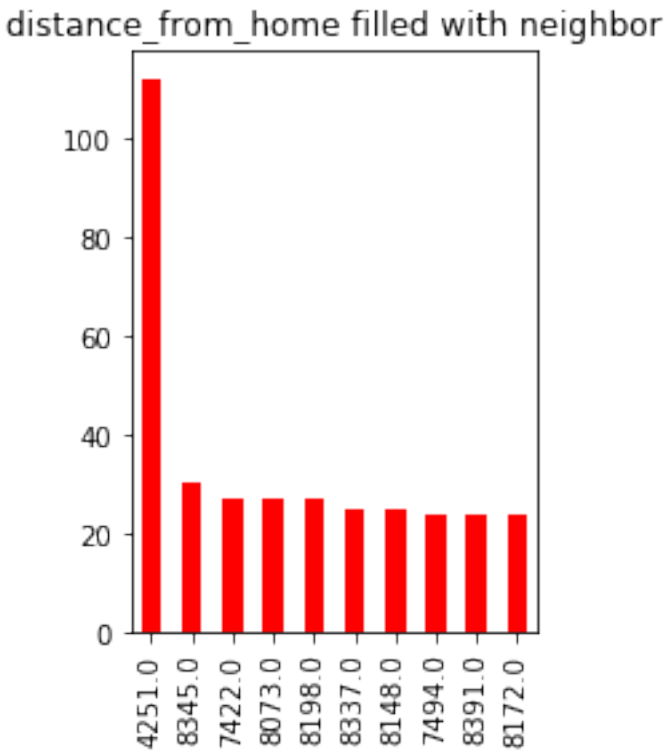
In [31]:
```
# compare2
plt.subplot(121)
ax=df['distance_from_home'].value_counts().head(10).plot.bar(color='DarkBlue')
plt.title("distance_from_home Origin")
plt.show()

plt.subplot(122)
bx = p2['distance_from_home'].value_counts().head(10).plot.bar(color='Red')
plt.title("distance_from_home filled with mode")
plt.show()
```


distance_from_home Origin
30
25
20
15
10
5
0
8345.0
8198.0
7422.0
8073.0
8337.0
8148.0
7579.0
8391.0
7494.0
8172.0


distance_from_home filled with mode
30
25
20
15
10
5
0
8345.0
8198.0
7422.0
8073.0
8337.0
8148.0
7579.0
8391.0
7494.0
8172.0

In [34]:
```
# compare3
plt.subplot(121)
ax=df['distance_from_home'].value_counts().head(10).plot.bar(color='DarkBlue')
plt.title("distance_from_home Origin")
plt.show()

plt.subplot(122)
bx = p3.value_counts().head(10).plot.bar(color='Red')
plt.title("distance_from_home filled with neighbor")
plt.show()
```




distance_from_home filled with neighbor
100
80
60
40
20
0
4251.0
8345.0
7422.0
8073.0
8198.0
8337.0
8148.0
7494.0
8391.0
8172.0