数据摘要

1.标称属性

building数据集中的标称属性有：
['Permit Type', 'Block', 'Lot', 'Street Number', 'Street Number Suffix', 'Street Name', 'Street Suffix','Current Status', 'Structural Notification', 'Voluntary Soft-Story Retrofit', 'Fire Only Permit','Existing Use', 'Proposed Use', 'Plansets', 'TIDF Compliance', 'Existing Construction Type','Proposed Construction Type', 'Site Permit', 'Supervisor District', 'Neighborhoods - Analysis Boundaries']

对于标称属性，求出每个可能值的频数并保存在txt文件中，代码如下：

```
def frequency(NominalAttribute, data):
    for n in NominalAttribute:
        doc = open('frequencyOfNominalAttribute.txt', 'a')
        print('*************** %s ***************' % n, file=doc)
        print(data[n].value_counts(), file=doc)
        doc.close()
```

求出的部分结果如下：

```
*************** Permit Type ***************
8    178844
3     14663
4      2892
2       950
6       600
7       511
1       349
5        91
Name: Permit Type, dtype: int64
*************** Block ***************
3708     1195
3735      750
7331      680
0289      640
3709      584
3717      578
3707      576
3721      567
3706      561
0259      554
3705      534
3722      498
4700      489
3704      465
3713      440
7295      400
0291      364
0288      361
3710      357
```

2.数值属性

此数据集中的数值属性包括：
NumericalAttribute = ['Number of Existing Stories', 'Number of Proposed Stories','Estimated Cost', 'Revised Cost', 'Existing Units', 'Proposed Units']

对于数值属性，求出了每个数值属性的最大值、最小值、均值、中位数、四分位数及缺失值的个数，代码如下：

```
def describeNumericalAttribute(NumericalAttribute, data):
    for n in NumericalAttribute:
        doc = open("describeNumericalAttribute.txt", 'a')
        print('*************** %s ***************' % n, file=doc)
        print("max:%s" % (data[n].max()), file=doc)
        print("min:%s" % (data[n].min()), file=doc)
        print("mean:%s" % (data[n].mean()), file=doc)
        print("median:%s" % (data[n].median()), file=doc)
        print("quantile1:%s" % (data[n].quantile(0.25)), file=doc)
        print("quantile2:%s" % (data[n].quantile(0.5)), file=doc)
        print("quantile3:%s" % (data[n].quantile(0.75)), file=doc)
        print("theNumberOfNull:%s" % (data[n].count()), file=doc)
        doc.close
```

求出的部分结果如下所示：

```
*************** Number of Existing Stories ***************
max:78.0
min:0.0
mean:5.705773271157344
median:3.0
quantile1:2.0
quantile2:3.0
quantile3:4.0
theNumberOfNull:156116
*************** Number of Proposed Stories ***************
max:78.0
min:0.0
mean:5.745042683552092
median:3.0
quantile1:2.0
quantile2:3.0
quantile3:4.0
theNumberOfNull:156032
*************** Estimated Cost ***************
max:537958646.0
min:1.0
mean:168955.44329681533
median:11000.0
quantile1:3300.0
quantile2:11000.0
quantile3:35000.0
theNumberOfNull:160834
```

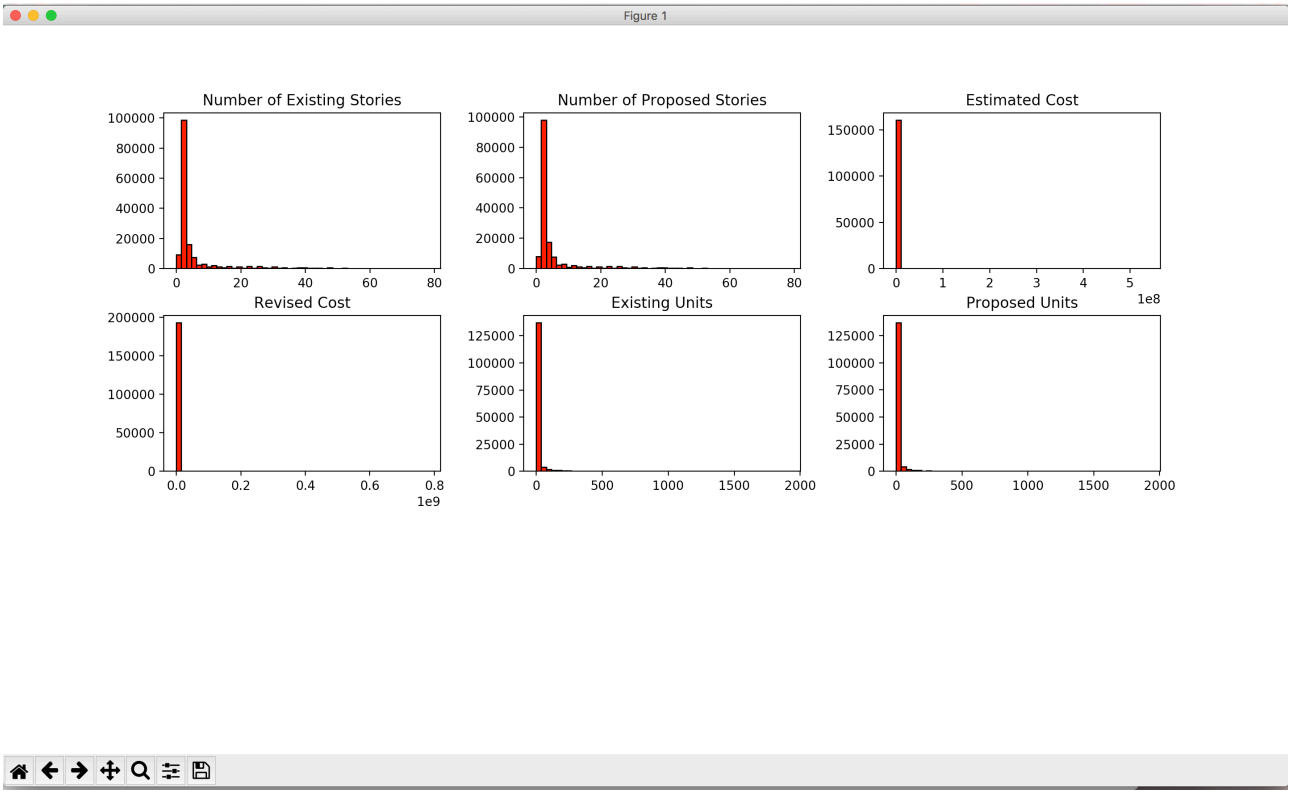
数据的可视化

1.直方图

对每种数值属性绘制直方图，代码如下

```
def histogram(NumericalAttribute, data):
    for i, col in enumerate(NumericalAttribute):
        if i % cellSize == 0:  # 一页有cellSize张子图，如果第i列是第cellSize+1张图，就另起一页
            fig = plt.figure(FIGSIZE=(15, 15))
        ax = fig.add_subplot(rowSize, colSize, (i % cellSize) + 1)
        data[col].hist(ax=ax, grid=False,
                       bins=50, facecolor='red', edgecolor='black')
        plt.title(col)
        if (i + 1) % cellSize == 0 or i + 1 == len(NumericalAttribute):
            plt.subplots_adjust(wspace=0.3, hspace=0.3)
            plt.show()
```

绘制出的直方图如下所示：

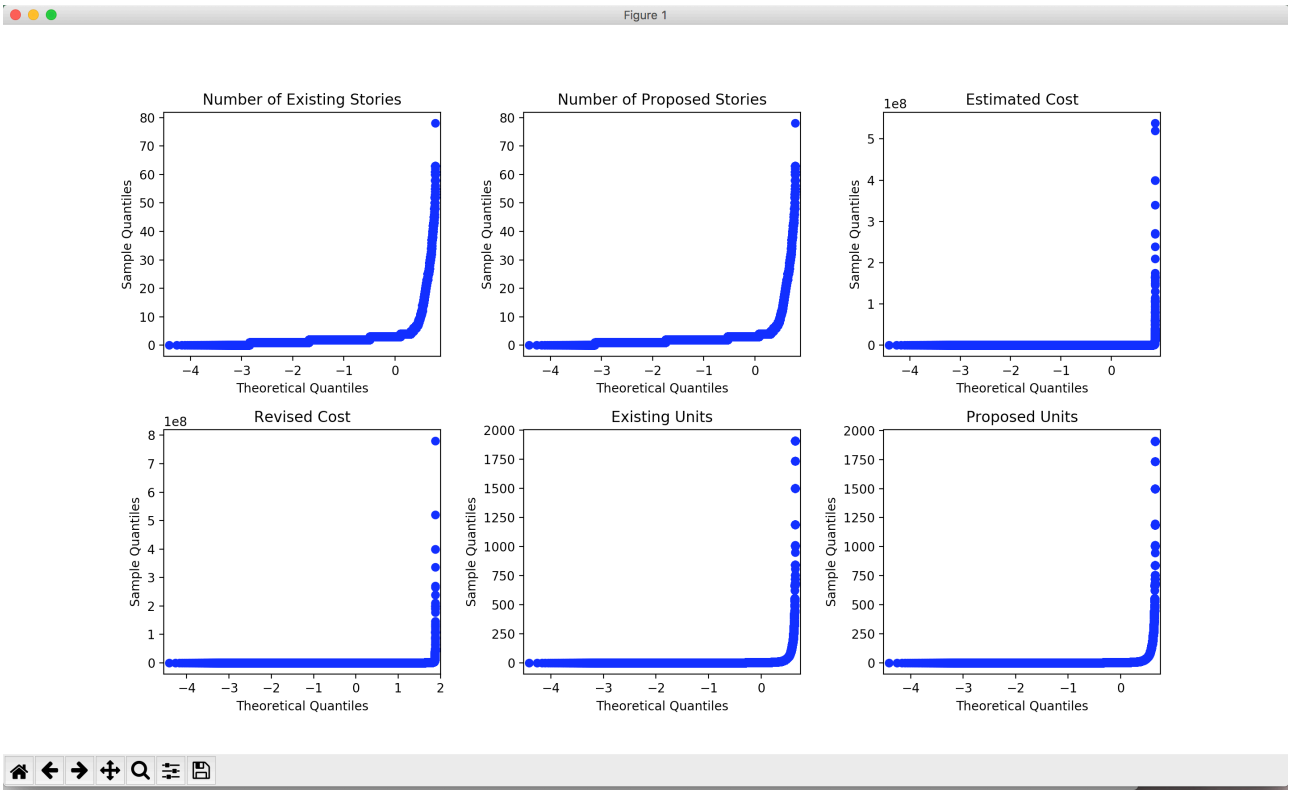


2.qq图

绘制qq图的代码如下所示：

```
def qq(NumericalAttribute, data):
    for i, col in enumerate(NumericalAttribute):
        if i % 6 == 0:
            fig = plt.figure()
        ax = fig.add_subplot(2, 3, (i % 6) + 1)
        sm.qqplot(data[col], ax=ax)
        ax.set_title(col)
        if (i + 1) % 6 == 0 or i + 1 == len(NumericalAttribute):
            plt.subplots_adjust(wspace=0.3, hspace=0.3)
            plt.show()
```

绘制的qq图如下，可以直观地利用qq图检验分布是否为正态分布

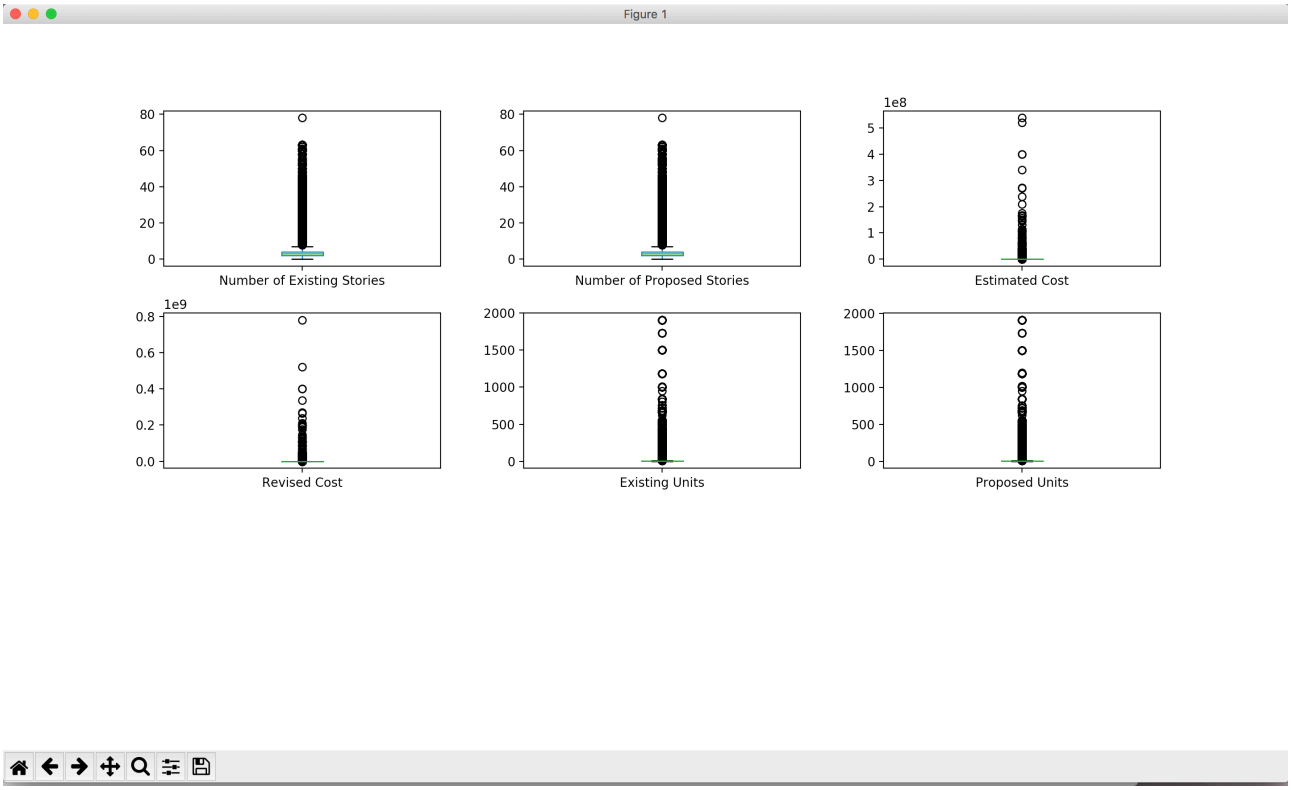


3.盒图

绘制盒图的代码如下：

```
def boxplot(NumericalAttribute, data):
    for i, col in enumerate(NumericalAttribute):
        if i % cellSize == 0:
            fig = plt.figure()
        ax = fig.add_subplot(colSize, rowSize, (i % cellSize) + 1)
        data[col].plot.box(ax=ax)
        if (i + 1) % cellSize == 0 or i + 1 == len(NumericalAttribute):
            plt.subplots_adjust(wspace=0.3, hspace=0.3)
            plt.show()
```

绘制出的部分盒图如下，可以直观的观察到离群值的分布



缺失数据的处理

1.将缺失部分剔除

可以进行填充的字段有如下，都是标称属性

```
NullValue = ['Structural Notification', 'Voluntary Soft-Story Retrofit',
             'Fire Only Permit', 'TIDF Compliance']
```

可填充的属性字段中除了最后一个均为bool型，如果空的话表示否（0），可以用N填充。而最后一个属性，只有两条记录不是空，空表示否，也可以用N填充。
剔除缺失数据的代码如下：

```
df_dropna = data.dropna()
print(df_dropna.shape)
```

输出(0,43)，代表剔除之后数据为空。

2.通过属性的相关关系填补缺失值

```
df_fillna = data[NullValue].fillna('NA')

doc = open('frequencyOfNominalAttribute.txt', 'a')
print('original:', file=doc)
doc.close()
frequency(NullValue, data)

doc = open('frequencyOfNominalAttribute.txt', 'a')
print('\nnew:', file=doc)
doc.close()
frequency(NullValue, df_fillna)
```

因为是标称属性，所以只能比较频数，结果如下。其中original为原始数据集部分，new为填充数据之后的频数

```
original:
*************** Structural Notification ***************
Y    6922
Name: Structural Notification, dtype: int64
*************** Voluntary Soft-Story Retrofit ***************
Y    35
Name: Voluntary Soft-Story Retrofit, dtype: int64
*************** Fire Only Permit ***************
Y    18827
Name: Fire Only Permit, dtype: int64
*************** TIDF Compliance ***************
Y    1
P    1
Name: TIDF Compliance, dtype: int64

new:
*************** Structural Notification ***************
NA    191978
Y       6922
Name: Structural Notification, dtype: int64
*************** Voluntary Soft-Story Retrofit ***************
NA    198865
Y         35
Name: Voluntary Soft-Story Retrofit, dtype: int64
*************** Fire Only Permit ***************
NA    180073
Y      18827
Name: Fire Only Permit, dtype: int64
*************** TIDF Compliance ***************
NA    198898
Y          1
P          1
Name: TIDF Compliance, dtype: int64
```