Building_Permits 数据分析与预处理

徐晶　　　2120171083

一、数据读取和属性分类

利用 Python 中的 pandas 库进行 csv 数据文件的读取：

```
def read_csv(path, na_values=None):
    '读取csv数据集'
    return pd.read_csv(path, na_values=na_values, low_memory=False)

csv_path='./Building_Permits.csv'
dataFrame=read_csv(csv_path,None)
```

对属性进行分类，分为标称属性和数值属性：

```
#标称属性
name_category = ['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']
#数值属性
name_value = ['Number of Existing Stories', 'Number of Proposed Stories', 'Estimated Cost', 'Revised Cost', 'Existing Units', 'Proposed Units']
```

二、数据可视化和摘要

2.1 数据摘要

标称数据：

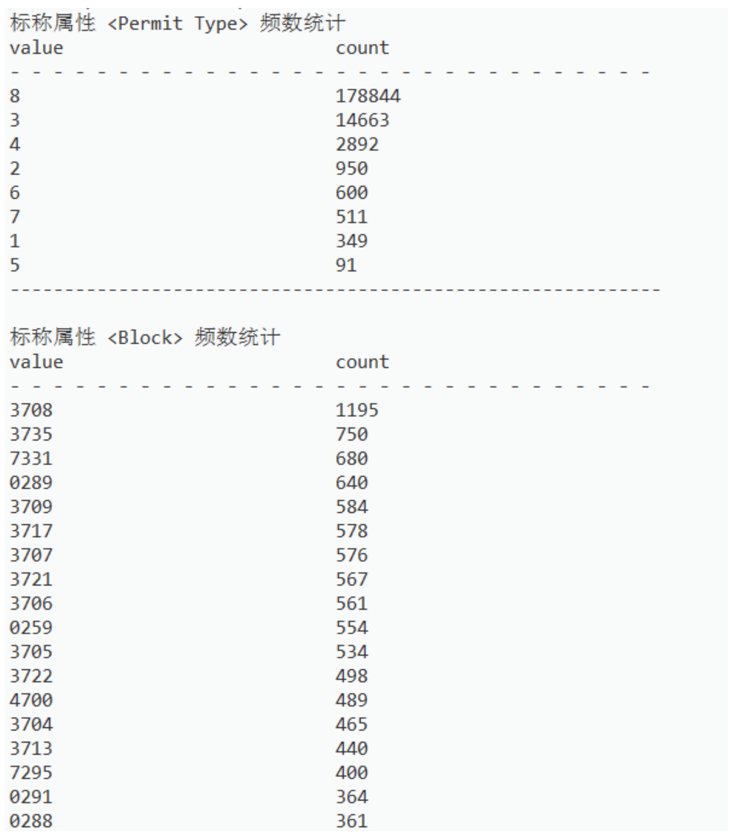
对于标称属性，使用 pandas 中的 value_counts 函数统计每个标称属性的所有可能取值的频数。

```
def count(dataFrame, columns, dropna=False, format_width=30):
    '标称属性，给出每个可能取值的频数'
    format_text = '{{:<{0}}}{{:<{0}}}'.format(format_width)
    for col in columns:
        print('标称属性 <{}> 频数统计'.format(col))
        print(format_text.format('value', 'count'))
        print('- ' * format_width)

        counts = pd.value_counts(dataFrame[col].values, dropna=False)
        for i, index in enumerate(counts.index):
            if pd.isnull(index): # NaN?
                print(format_text.format('-NaN-', counts.values[i]))
            else:
                print(format_text.format(index, counts[index]))
        print('--' * format_width)
        print()

count(dataFrame,name_category)
```

部分结果如下：



```
标称属性 <Permit Type> 频数统计
value                          count
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
8                              178844
3                              14663
4                              2892
2                              950
6                              600
7                              511
1                              349
5                              91
-------------------------------------------------------------

标称属性 <Block> 频数统计
value                          count
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
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
```

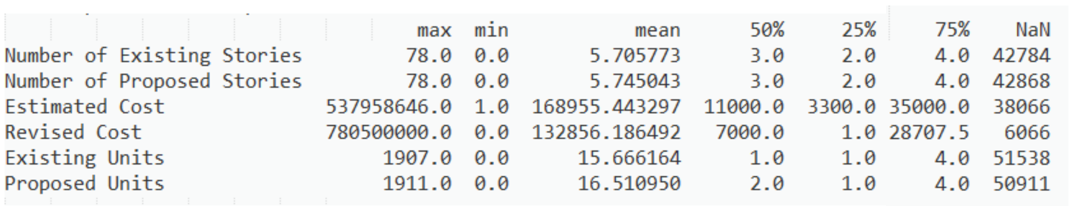
数值属性：

对于数值属性，使用 pandas 中 describe()函数给出其最小、最大、均值、中位数、四分位数及缺失值个数：

```python
def describe(dataFrame, columns):
    '数值属性，给出最大、最小、均值、中位数、四分位数及缺失值的个数'
    desc = dataFrame[columns].describe()
    statistic = DataFrame()
    statistic['max'] = desc.loc['max']
    statistic['min'] = desc.loc['min']
    statistic['mean'] = desc.loc['mean']
    statistic['50%'] = desc.loc['50%']
    statistic['25%'] = desc.loc['25%']
    statistic['75%'] = desc.loc['75%']
    statistic['NaN'] = dataFrame[columns].isnull().sum()
    print(statistic)

describe(dataFrame,name_value)
```

部分结果如下：



```
                                 max  min           mean     50%     25%      75%    NaN
Number of Existing Stories      78.0  0.0       5.705773     3.0     2.0      4.0  42784
Number of Proposed Stories      78.0  0.0       5.745043     3.0     2.0      4.0  42868
Estimated Cost           537958646.0  1.0  168955.443297 11000.0  3300.0  35000.0  38066
Revised Cost             780500000.0  0.0  132856.186492  7000.0     1.0  28707.5   6066
Existing Units                1907.0  0.0      15.666164     1.0     1.0      4.0  51538
Proposed Units                1911.0  0.0      16.510950     2.0     1.0      4.0  50911
```

2.2 数据可视化

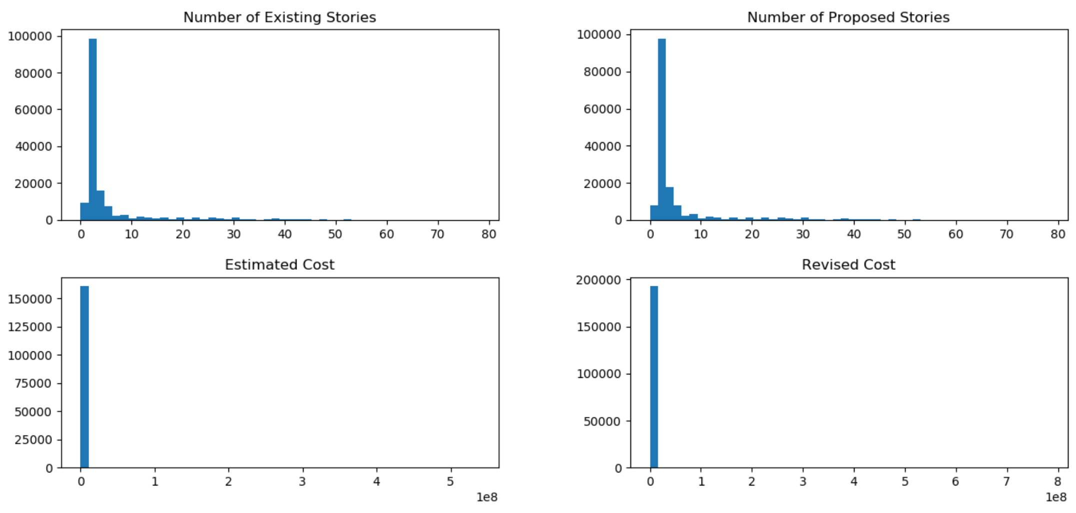
直方图

使用 matplotlib 绘制直方图

```python
def histogram(dataFrame, columns):
    '直方图'
    for i, col in enumerate(columns):
        if i % cell_size == 0:
            fig = plt.figure()
        ax = fig.add_subplot(col_size, row_size, (i % cell_size) + 1)
        dataFrame[col].hist(ax=ax, grid=False, figsize=(15, 15), bins=50)
        plt.title(col)
        if (i + 1) % cell_size == 0 or i + 1 == len(columns):
            plt.subplots_adjust(wspace=0.3, hspace=0.3)
            plt.show()

histogram(dataFrame,name_value)
```

部分结果如下：



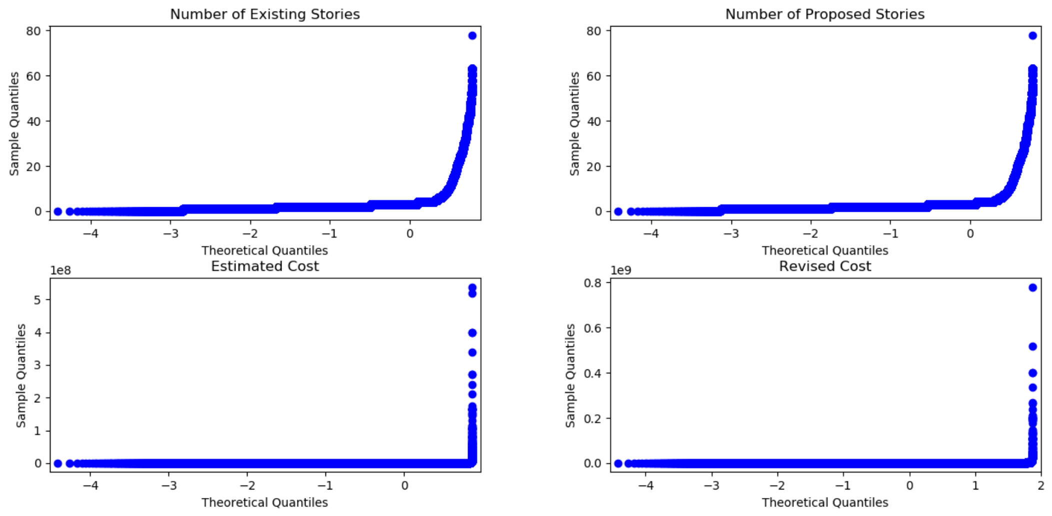
qq 图：

使用 matplotlib 绘制 qq 图

```python
def qqplot(dataFrame, columns):
    'qq图'
    for i, col in enumerate(columns):
        if i % cell_size == 0:
            fig = plt.figure(figsize=(15, 15))
        ax = fig.add_subplot(col_size, row_size, (i % cell_size) + 1)
        sm.qqplot(dataFrame[col], ax=ax)
        ax.set_title(col)
        if (i + 1) % cell_size == 0 or i + 1 == len(columns):
            plt.subplots_adjust(wspace=0.3, hspace=0.3)
            plt.show()

qqplot(dataFrame,name_value)
```

部分结果如下：



根据 qq 图可知图像若是近似直线的，其对应属性为正态分布态。

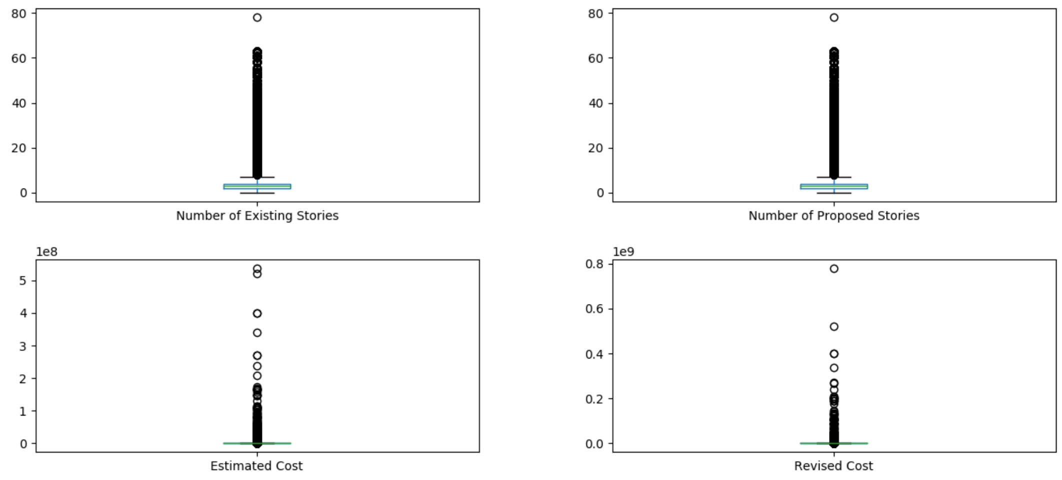
盒图：

使用 matplotlib 绘制盒图，对离群值进行识别：

```python
def boxplot(dataFrame, columns):
    '盒图'
    for i, col in enumerate(columns):
        if i % cell_size == 0:
            fig = plt.figure()
        ax = fig.add_subplot(col_size, row_size, (i % cell_size) + 1)
        dataFrame[col].plot.box(ax=ax, figsize=(15, 15))
        if (i + 1) % cell_size == 0 or i + 1 == len(columns):
            plt.subplots_adjust(wspace=0.3, hspace=0.3)
            plt.show()

boxplot(dataFrame,name_value)
```

部分结果如下：



三、数据缺失处理

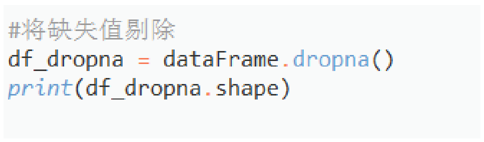
3.1 将缺失部分剔除

无缺失的字段：Permit Number，Permit Type，Permit Type Definition，Permit Creation Date，Block，Lot，Street Number，Street Name，Current Status，Current Status Date，Filed Date，Record ID 。

无填充意义的字段：Unit，Unit suffix，Description，Issued Date，Completed Date，First Construction Document Date，Permit Expiration Date，Existing Construction Type Description，Proposed Construction Type Description，Zipcode，Location，Street Number Suffix，Street Name Suffix，Existing Use，Existing Units，Proposed Use，Proposed Units，Plansets，Existing Construction Type，Proposed Construction Type，Supervisor District，Neighborhoods - Analysis Boundaries，Number of Existing Stories，Number of Proposed Stories，Estimated Cost，Revised Cost。

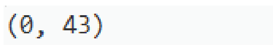
可填充的属性字段：Structural Notification，Voluntary Soft-Story Retrofit，Fire Only Permit，TIDF Compliance，Site Permit

可填充的属性字段中除了 TIDF Compliance 均为布尔型，空表示否，可用 N 填充；TIDF Compliance 字段只有两条记录不为空，空表示否，可用 N 填充，对其进行剔除。



```
#将缺失值剔除
df_dropna = dataFrame.dropna()
print(df_dropna.shape)
```

输出：



```
(0, 43)
```

说明剔除缺失部分数据后，数据集为空。

3.2 用最高频率值来填补缺失值

无

3.3 通过属性的相关关系来填补缺失值

```
#通过属性的相关关系来填补缺失值
cols = ['Structural Notification', 'Voluntary Soft-Story Retrofit', 'Fire Only Permit', 'TIDF Compliance']
df_fillna = dataFrame[cols].fillna('N')

print('                    旧数据\n')
count(dataFrame, cols)

print('\n\n                    新数据\n')
count(df_fillna, cols)
```

部分结果如下：

```
标称属性 <Structural Notification> 频数统计
value                              count
- - - - - - - - - - - - - - - - - - - - - - - - - - - - -
-NaN-                              191978
Y                                  6922
-----------------------------------------------------------

标称属性 <Voluntary Soft-Story Retrofit> 频数统计
value                              count
- - - - - - - - - - - - - - - - - - - - - - - - - - - - -
-NaN-                              198865
Y                                  35
-----------------------------------------------------------

标称属性 <Fire Only Permit> 频数统计
value                              count
- - - - - - - - - - - - - - - - - - - - - - - - - - - - -
-NaN-                              180073
Y                                  18827
-----------------------------------------------------------

标称属性 <TIDF Compliance> 频数统计
value                              count
- - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

3.4 通过数据对象之间的相似性来填补缺失值

无