互评作业一

wine reviews第一个数据集描述及预处理

wine reviews数据集中包含了两个文件，即winemag-data_first150k.csv与winemag-data-130k.csv，分别由10列130k行与10列150行。本章仅仅对winemag-data_first150k.csv数据集进行处理。这上述数据集中统计了280k条葡萄酒的信息，即生产国家、评价、分数、价格与生产省份等。其中标称属性有id, country, designation, province, region_1, region_2, title, variety和winery。数值属性有points和price。本章节将分别对wine reviews 的两个数据集进行描述与预处理。

数据分析

本节将统计标称属性取值的频数与对数值属性进行五数概括与统计缺失值的个数。

数据摘要

- 标称属性
  统计标称属性所有可能取值的频数并保存到了本地的txt文件中。由于取值的可能性太多，在这里将不再进行展示了。代码如下：

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import mode
import sklearn.ensemble
from sklearn.neighbors import KNeighborsClassifier, KNeighborsRegressor

data_name = r'C:\Users\Xiaomeng Fan\Desktop\研究生的各种作业\数据挖掘\作业四
\wine_1\winemag-data_first150k.csv'
data_name2 = r'C:\Users\Xiaomeng Fan\Desktop\研究生的各种作业\数据挖掘\作业四
\wine_1\winemag-data-130k.csv'
txt=open(r'C:\Users\Xiaomeng Fan\Desktop\研究生的各种作业\数据挖掘\作业四
\wine_1\Wine_reviews_result1.txt','w',encoding = 'utf-8')
problem_1 = 0
file = pd.read_csv(data_name)
data = pd.DataFrame(file)

print(file.columns)
atts = file.columns
num=0
if problem_1:
  for i in atts[1:]:
    print(i)
    if i!= 'points' and i!= 'price' and i!='description':
        dict_temp = {}
        for j in file.loc[:,i]:
            if j in dict_temp.keys():
                dict_temp[j]+=1
            else:
                dict_temp[j]=1
                num+=1

        # print(dict_temp)
        txt.write(i+'\n')
        txt.write(str(dict_temp))
        txt.write('\n')
        print(num)
  txt.close()
```

```
Index(['Unnamed: 0', 'country', 'description', 'designation', 'points',
       'price', 'province', 'region_1', 'region_2', 'variety', 'winery'],
      dtype='object')
```

- 数值属性
  对price与points数值属性进行五数概括及统计缺失值的个数。我们将price与points数据中的缺失值剔除出去，再对清理之后的数据进行五数概括。对price的数据处理代码如下：

```
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.nanpercentile(nums,25)
    Median=np.nanmedian(nums)
    Q3=np.nanpercentile(nums,75)




    return Minimum,Q1,Median,Q3,Maximum

price_num_list = []
att_pool = ['price']
for att in att_pool:

  for point in file.loc[:,att]:
   if isinstance(point,int) or isinstance(point,float):
    price_num_list.append(point)
   else:
    print(point)
min_temp, Q1, median, Q3, max_temp = fiveNumber(price_num_list)
print(att+' five number:')
print(min_temp,Q1,median,Q3,max_temp)
price_np = np.array(price_num_list)
```

```
price five number:
4.0 16.0 24.0 40.0 2300.0
```

对points的数据处理代码如下：

```
points_num_list = []
att_pool = ['points']
for att in att_pool:

  for point in file.loc[:,att]:
   if isinstance(point,int) or isinstance(point,float):
```

```
            points_num_list.append(point)
        else:
            print(point)
points_np = np.array(points_num_list)
min_temp, Q1, median, Q3, max_temp = fiveNumber(points_num_list)
print(att+' five number:')
print(min_temp,Q1,median,Q3,max_temp)
```

```
points five number:
80 86.0 88.0 90.0 100
```

对空缺的数据进行统计的代码及结果如下：

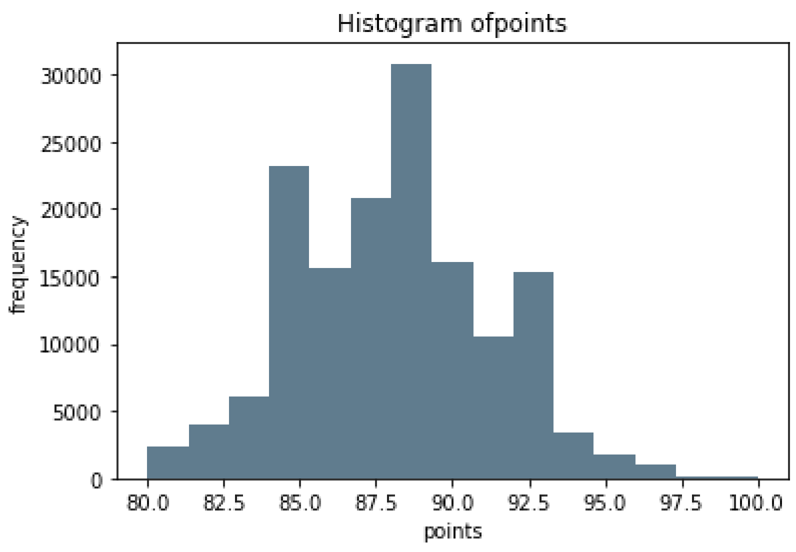
```
print(data.isnull().sum())
```

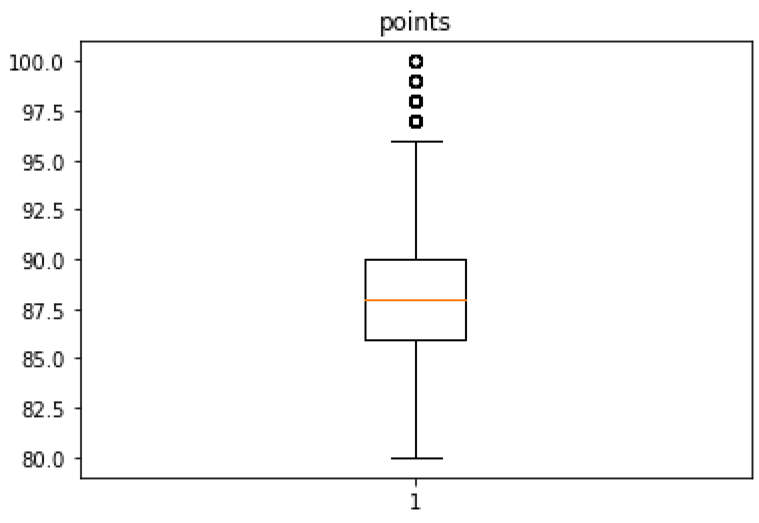
```
Unnamed: 0         0
country            5
description        0
designation    45735
points             0
price          13695
province           5
region_1       25060
region_2       89977
variety            0
winery             0
dtype: int64
```

数据可视化

- 直方图
  分别对points与price的数值属性绘制直方图。由于数值属性的数据存在缺失数据，因此我们需要首先将缺失数据剔除再进行相关可视化。对points绘制直方图与盒图的代码及结果如下：

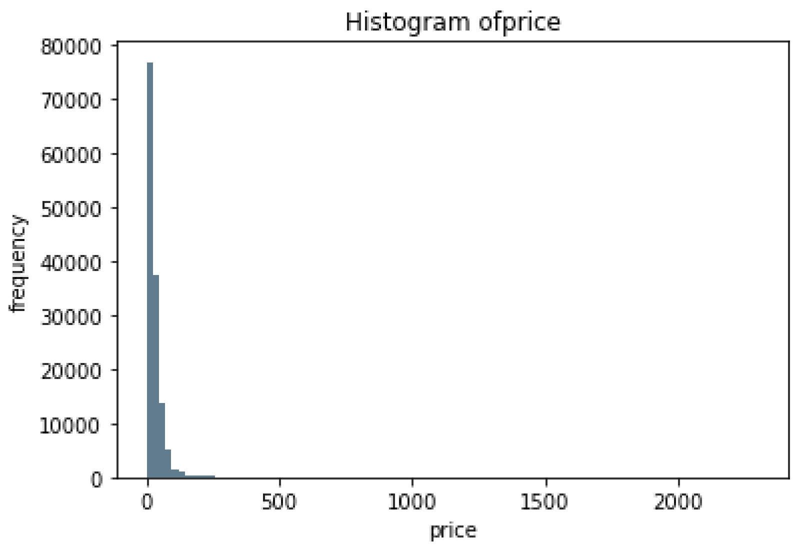
```
def show_boxplot(data_temp, att_temp):

    data_temp1 = data_temp.copy()

    for i in att_temp:
        data_i = data_temp1.dropna(subset=[i])
        att_i = data_i.loc[:,i].values
        plt.boxplot(att_i, notch=False, sym='o', vert=True)
        plt.title(i)
        plt.show()


def show_hist(data_temp, att_temp, bin_numbers):
    data_temp1 = data_temp.copy()

    for i in att_temp:
        data_i = data_temp1.dropna(subset=[i])
        att_i = data_i.loc[:,i].values
        plt.hist(att_i, bins = bin_numbers, color = '#607c8e', rwidth=5)
        plt.xlabel(i)
        plt.ylabel('frequency')
        plt.title('Histogram of'+ i )
        plt.show()



att = ['points']
show_hist(data, att, 15)
show_boxplot(data,att)
```

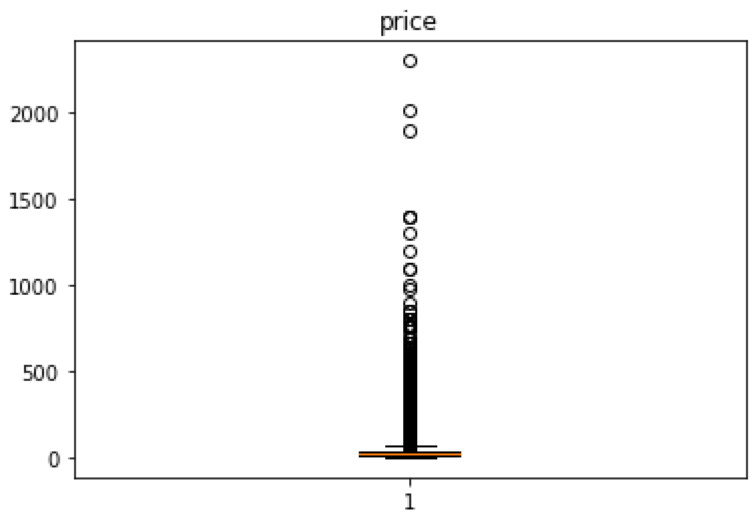




对price数据属性绘制直方图与盒图的代码与结果如下：

```
att = ['price']
show_hist(data, att,100)
show_boxplot(data,att)
```





由此可以看出，points的数据是接近于正太分布的，price的数据分布却不正常，主要表现为离群点很多，并且离群点的数值相较于平均值、众数与中尉数的差距很大。

数据缺失处理

观察数据集中缺失数据，分析其缺失的原因，分别使用下列四种策略对缺失值进行处理

将缺失部分剔除

我们将所有缺失数据全部剔除并保存。对处理之后的数据五数概括的代码及结果为：

```
def lost_edition1():
    data_lost1 = data.copy()
    data_lost1 = data_lost1.dropna()
    return data_lost1

save_or_not = False
data_lost1 = lost_edition1()
print(data_lost1.isnull().sum())
if save_or_not:
    data_lost1.to_csv('data_delete.csv')
```

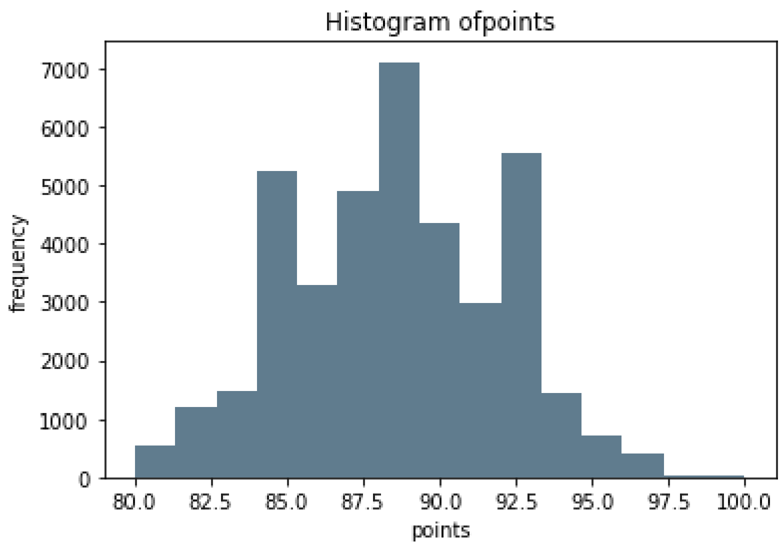
```
price_value = data_lost1.loc[:,'price'].values
points_value  = data_lost1.loc[:,'points'].values

min_temp, Q1, median, Q3, max_temp = fiveNumber(price_value)
print('price five number:')
print(min_temp,Q1,median,Q3,max_temp)

min_temp, Q1, median, Q3, max_temp = fiveNumber(points_value)
print('points five number:')
print(min_temp,Q1,median,Q3,max_temp)
```

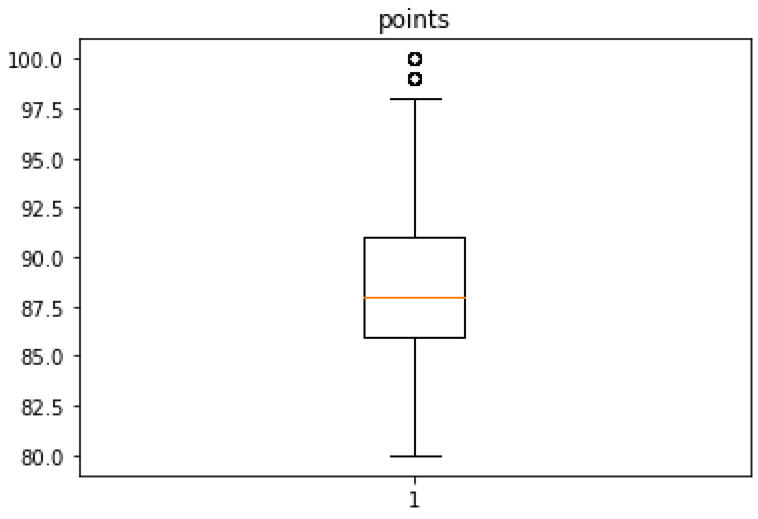
```
Unnamed: 0     0
country        0
description    0
designation    0
points         0
price          0
province       0
region_1       0
region_2       0
variety        0
winery         0
dtype: int64
price five number:
4.0 22.0 32.0 45.0 2013.0
points five number:
80 86.0 88.0 91.0 100
```

原数据的points属性的五数概括为：[80,86,88,90,100], 数据处理之后的五数概括为[80,86,88,91,100]。原数据的price属性的五数概括为：[4.0 16.0 24.0 40.0 2300.0], 数据处理之后的五数概括为[4 22 32 45 2013]。

对数据处理之后points的直方图与盒图的代码及结果为：

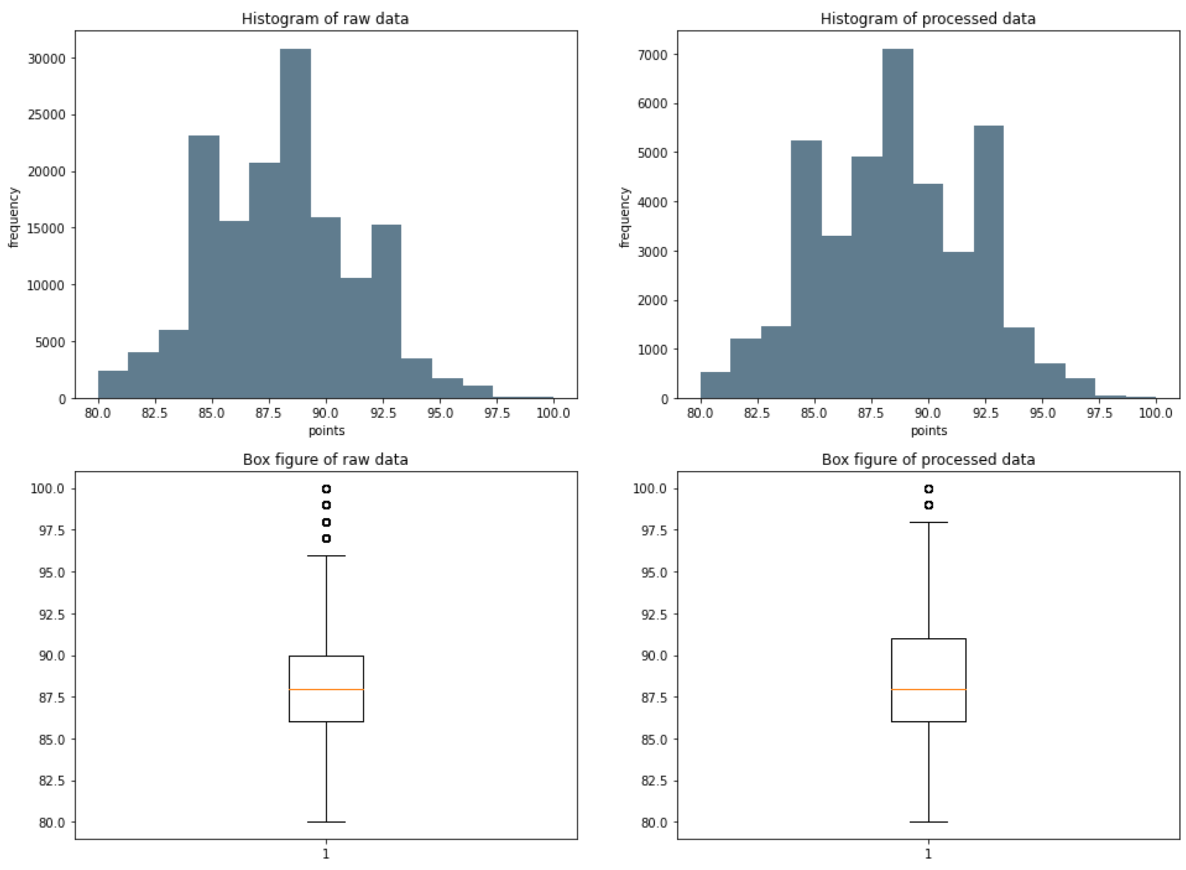
```
att = ['points']
show_hist(data_lost1, att, 15)
show_boxplot(data_lost1,att)
```





与原数据的对比结果如下：

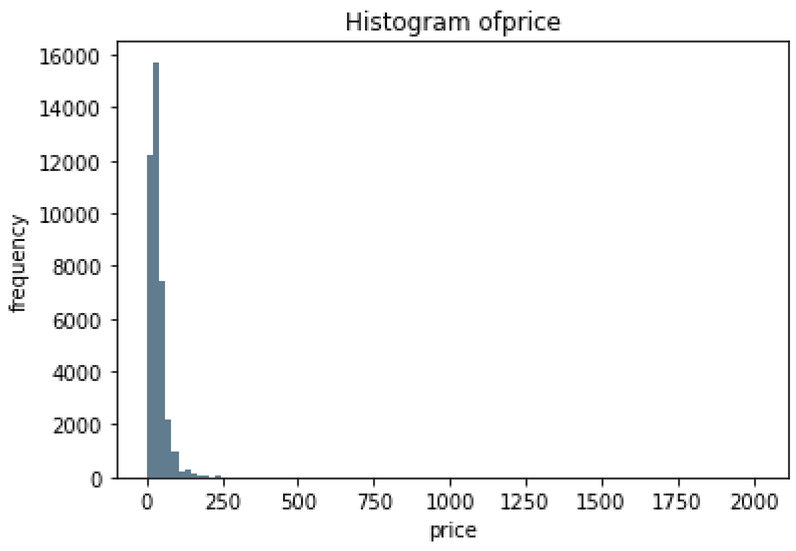
```
def subplot_show(data1, att1, data2, att2, bin_num, title1, title2):

    data1_temp = data1.copy()

    data_i = data1_temp.dropna(subset=[att1])
    data1_att = data_i.loc[:,att1].values
    data2_att = data2.loc[:,att2].values
    plt.figure(figsize=(16, 12))
    plt.subplot(221)
    plt.hist(data1_att,bins = bin_num, color = '#607c8e', rwidth=5)
    plt.xlabel(att1)
    plt.ylabel('frequency')
    plt.title('Histogram of '+ title1 )

    plt.subplot(222)
    plt.hist(data2_att,bins = bin_num, color = '#607c8e', rwidth=5)
    plt.xlabel(att2)
    plt.ylabel('frequency')
    plt.title('Histogram of '+ title2 )
    plt.subplot(223)
    plt.boxplot(data1_att, notch=False, sym='o', vert=True)

    plt.title('Box figure of '+title1)

    plt.subplot(224)
    plt.boxplot(data2_att, notch=False, sym='o', vert=True)
    plt.title('Box figure of '+title2)

    plt.show()
att = 'points'
subplot_show(data, att, data_lost1, att, 15, 'raw data', 'processed data')
```

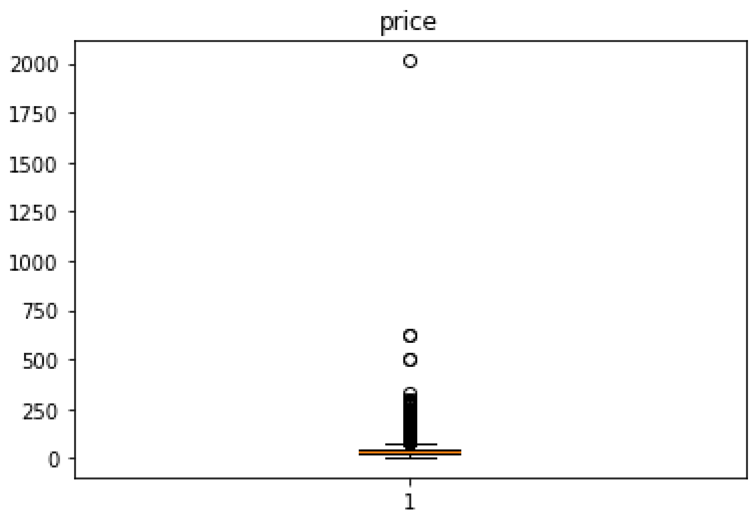


由此可以看出，在去掉全部的缺失数据之后，points数据里的离群点减少了，并且数据的更趋向于正态分布。

对数据处理之后price的直方图与盒图的代码及结果为：

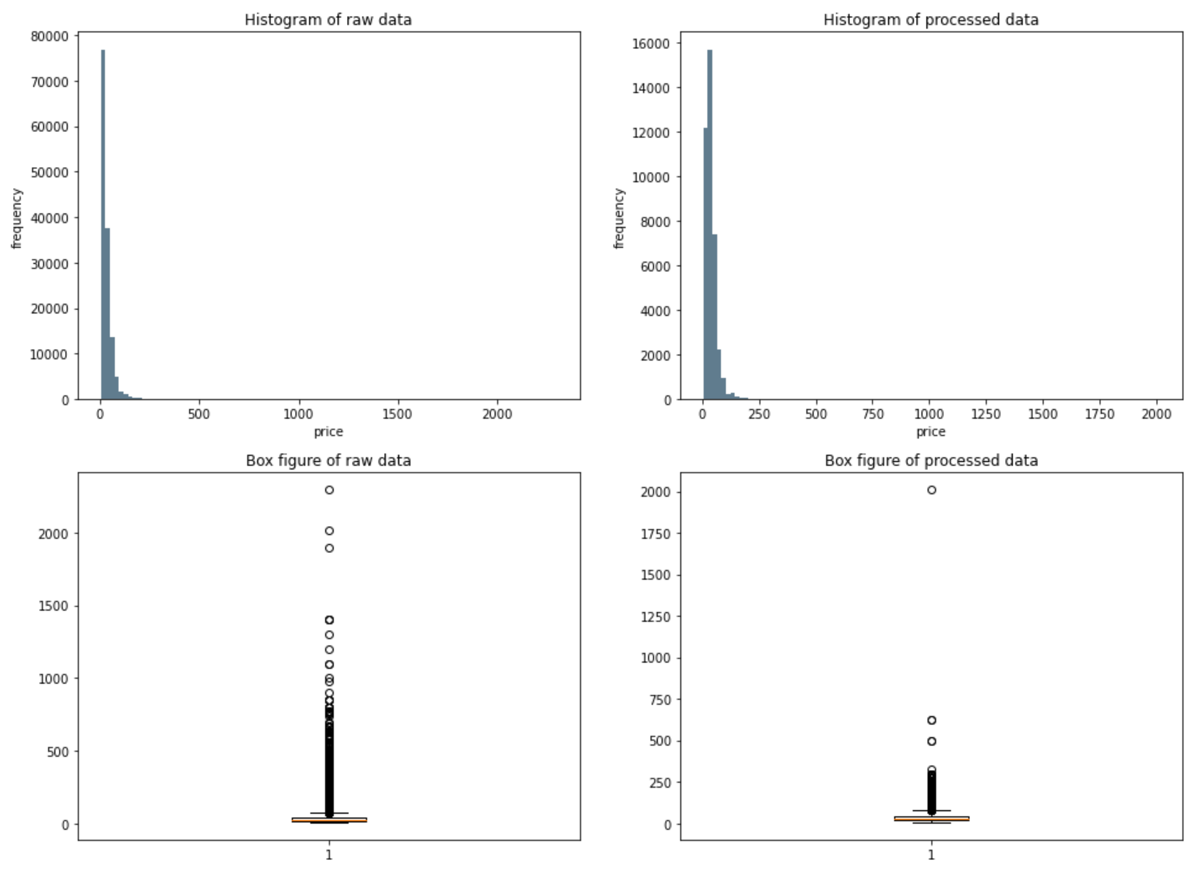
```
att = ['price']
show_hist(data_lost1, att, 100)
show_boxplot(data_lost1,att)
```





与原数据代码及结果为：

```
att = 'price'
subplot_show(data, att, data_lost1, att, 100, 'raw data', 'processed data')
```



由此可以看出，在去掉全部的缺失数据之后，price数据里的离群点减少了，并且数据的更趋向于正态分布。

与原数据的对比结果如下;

用最高频率值来填补缺失值

我们首先计算price属性最高频率的数值，将空缺值替换为该值并保存。对处理之后的数据五数概括的代码及结果为：

```python
def lost_edition2(price_np):

    data_lost2 = data.copy()
    price_mode = mode(price_np)

    mode_temp = price_mode[0].tolist()[0]
    data_lost2['price'] = data_lost2['price'].fillna(mode_temp)
    return data_lost2


save_or_not = False
data_lost2 = lost_edition2(price_np)
print(data_lost2.isnull().sum())
if save_or_not:
    data_lost2.to_csv('data_mode.csv')


price_value_2 = data_lost2.loc[:,'price'].values
points_value_2  = data_lost2.loc[:,'points'].values

min_temp, Q1, median, Q3, max_temp = fiveNumber(price_value_2)
```

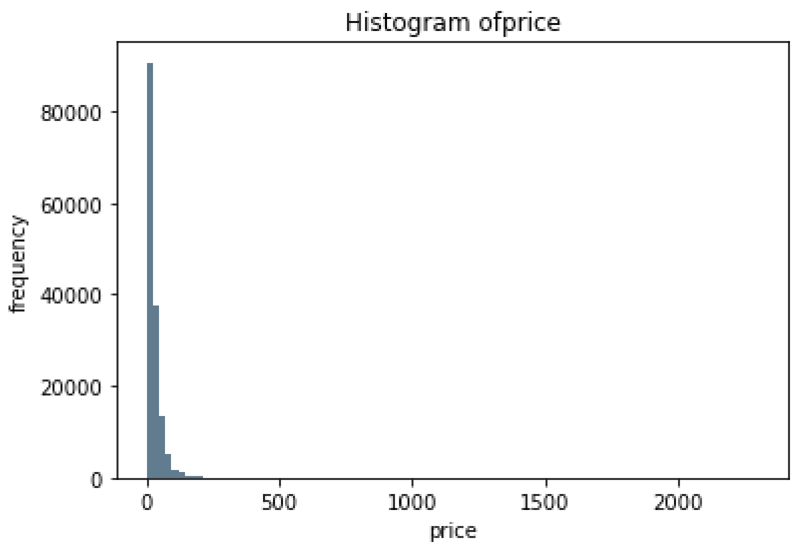
```
print('price five number:')
print(min_temp,Q1,median,Q3,max_temp)

min_temp, Q1, median, Q3, max_temp = fiveNumber(points_value_2)
print('points five number:')
print(min_temp,Q1,median,Q3,max_temp)
```

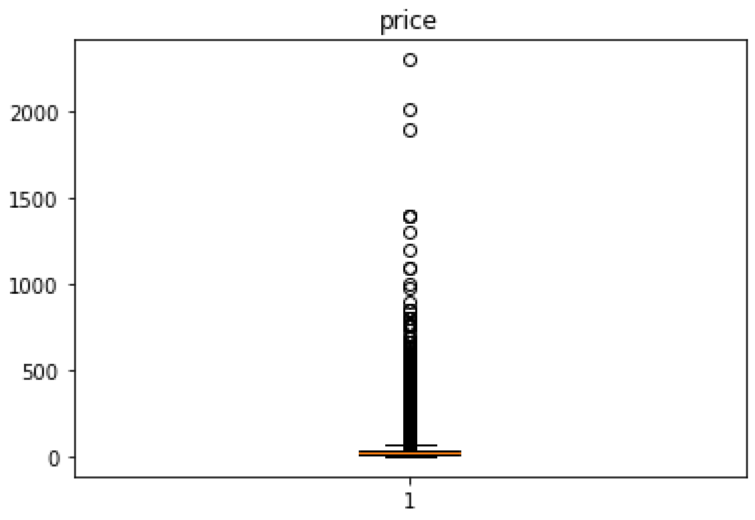
```
Unnamed: 0          0
country             5
description         0
designation     45735
points              0
price               0
province            5
region_1        25060
region_2        89977
variety             0
winery              0
dtype: int64
price five number:
4.0 16.0 22.0 38.0 2300.0
points five number:
80 86.0 88.0 90.0 100
```

```
原数据的points属性的五数概括为：[80,86,88,90,100]，数据处理之后的五数概括为
[80,86,88,90,100]。
原数据的price属性的五数概括为：[4.0 16.0 24.0 40.0 2300.0]，数据处理之后的五数概括为[4 16
22 38 2300]。
```

由于只对price数据进行了处理，因此只对price数据进行可视化，对数据处理之后price的直方图与盒图的代码及结果为：

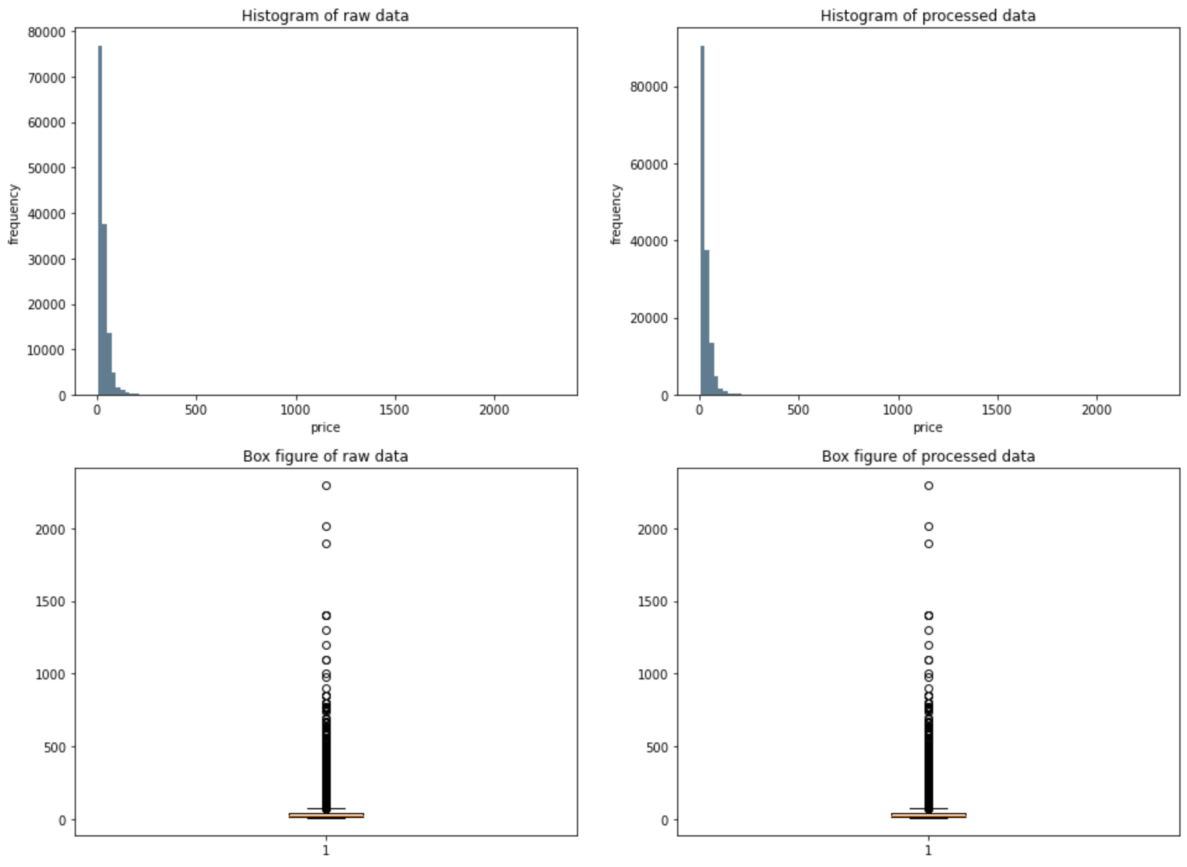
```
att = ['price']
show_hist(data_lost2, att, 100)
show_boxplot(data_lost2,att)
```





与原数据比较代码及结果为：

```
att = 'price'
subplot_show(data, att, data_lost2, att, 100, 'raw data', 'processed data')
```



通过属性的相关关系来填补缺失值

随机森林随机森林是利用多个决策树对样本进行训练、分类并预测的一种算法。其核心是随机，并包含两种含义，一是随机在原始训练数据中有放回的选取等量的数据作为训练样本，二是在建立决策树时，随机的选特征中选取一部分特征建立决策树。这两种随机使得各个决策树之间的相关性小，进一步提高模型的准确性。对于补充缺失值的问题，可以将该问题看作是一个回归问题。以本数据为例，将完整的

points属性看作是参考数据，将不完整的price看作是目标数据，将缺少的points的列去除，剩下的points数据与price数据构成训练数据。并将训练之后的模型用来根据缺少的price条目中的points数据进行拟合，最后对price数据进行填充。
我们使用随机森林的方法对price属性的缺失数据进行补全，其代码及结果如下：

```
def set_missing_prices():

    data_lost3 = data.copy()

    # 把已有的数值型特征取出来丢进Random Forest Regressor中
    price_df = data_lost3[['price', 'points']]

    # 乘客分成已知年龄和未知年龄两部分
    known_price = price_df[price_df.price.notnull()].values
    unknown_price = price_df[price_df.price.isnull()].values

    # y即目标年龄
    y = known_price[:, 0]

    # X即特征属性值
    X = known_price[:, 1:]

    # fit到RandomForestRegressor之中
    rfr = sklearn.ensemble.RandomForestRegressor(random_state=0,
n_estimators=2000, n_jobs=-1)
    rfr.fit(X, y)

    # 用得到的模型进行未知年龄结果预测
    predictedprice = rfr.predict(unknown_price[:, 1:])
#     print predictedAges
    # 用得到的预测结果填补原缺失数据
    data_lost3.loc[ (data_lost3.price.isnull()), 'price' ] = predictedprice

    return data_lost3

save_or_not = False
data_lost3 = set_missing_prices()
print(data_lost2.isnull().sum())
if save_or_not:
    data_lost3.to_csv('data_forest.csv')


price_value_3 = data_lost3.loc[:,'price'].values
points_value_3  = data_lost3.loc[:,'points'].values

min_temp, Q1, median, Q3, max_temp = fiveNumber(price_value_3)
print('price five number:')
print(min_temp,Q1,median,Q3,max_temp)

min_temp, Q1, median, Q3, max_temp = fiveNumber(points_value_3)
print('points five number:')
print(min_temp,Q1,median,Q3,max_temp)
```

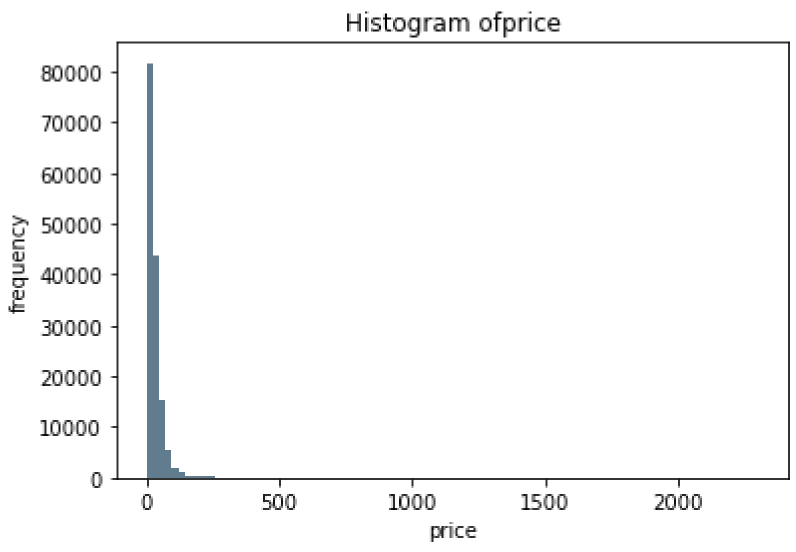
```
Unnamed: 0        0
country           5
description       0
```

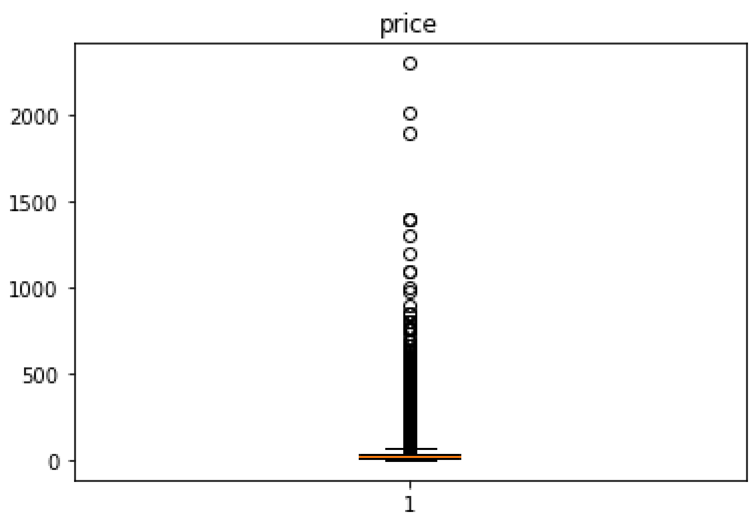
```
designation    45735
points             0
price              0
province           5
region_1       25060
region_2       89977
variety            0
winery             0
dtype: int64
price five number:
4.0 16.0 25.0 40.0 2300.0
points five number:
80 86.0 88.0 90.0 100
```

```
原数据的points属性的五数概括为：[80,86,88,90,100]，数据处理之后的五数概括为
[80,86,88,90,100]。
原数据的price属性的五数概括为：[4.0 16.0 24.0 40.0 2300.0]，数据处理之后的五数概括为[4 16
25 40 2300]。
```

由于只对price数据进行了处理，因此只对price数据进行可视化，对数据处理之后price的直方图与盒图的代码及结果为：

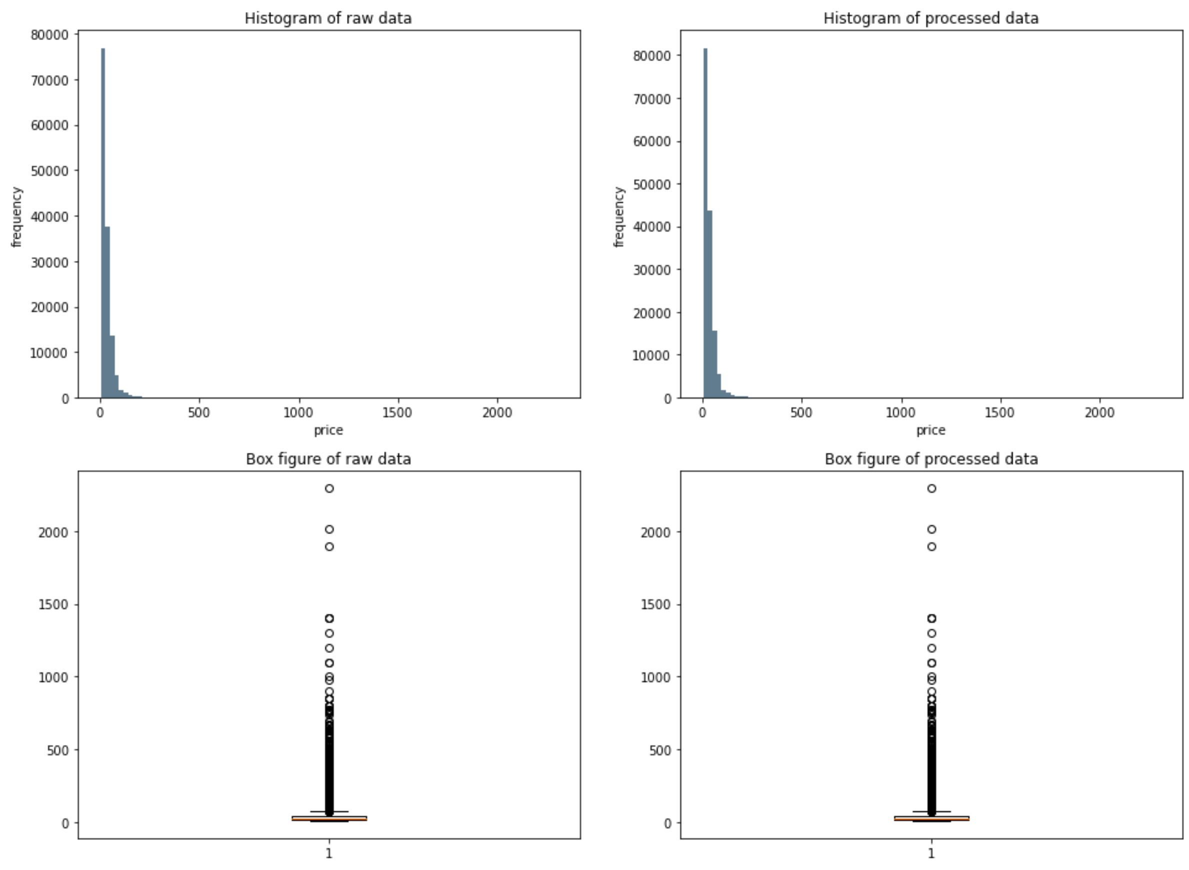
```
att = ['price']
show_hist(data_lost3, att, 100)
show_boxplot(data_lost3,att)
```





与原数据比较代码及结果为：

```
att = 'price'
subplot_show(data, att, data_lost3, att, 100, 'raw data', 'processed data')
```



通过数据对象之间的相似性来填补缺失值

利用KNN补充缺失数据是把目标列当做目标标量，利用非缺失的数据进行knn算法拟合，最后对目标列缺失进行预测。
我们使用KNN的方法对price属性的缺失数据进行补全，其代码及结果如下：

```
def knn_missing_filled(k = 3, dispersed = True):
    data_lost4 = data.copy()
    # x_train = data_lost4[data_lost4.price.notnull()]
['points'].values.reshape(-1,1)
    # y_train = data_lost4[data_lost4.price.notnull()]
['price'].values.reshape(-1,1)

    x_train =
data_lost4[data_lost4.price.notnull()].loc[:,'points'].values.reshape(-1,1)
    y_train =
data_lost4[data_lost4.price.notnull()].loc[:,'price'].values.reshape(-1,1)
    print(len(x_train))
    print(len(y_train))
    test = data_lost4[data_lost4.price.isnull()]['points'].values.reshape(-1,1)

    if dispersed:
        clf = KNeighborsClassifier(n_neighbors = k, weights = "distance")
    else:
        clf = KNeighborsRegressor(n_neighbors = k, weights = "distance")

    clf.fit(x_train, y_train)

    data_lost4.loc[ (data_lost4.price.isnull()), 'price' ] = clf.predict(test)

    return data_lost4



save_or_not = False
data_lost4 = knn_missing_filled()
print(data_lost2.isnull().sum())
if save_or_not:
    data_lost4.to_csv('data_knn.csv')


price_value_4 = data_lost4.loc[:,'price'].values
points_value_4  = data_lost4.loc[:,'points'].values

min_temp, Q1, median, Q3, max_temp = fiveNumber(price_value_4)
print('price five number:')
print(min_temp,Q1,median,Q3,max_temp)

min_temp, Q1, median, Q3, max_temp = fiveNumber(points_value_4)
print('points five number:')
print(min_temp,Q1,median,Q3,max_temp)
```

```
137235
137235
```

```
<ipython-input-68-1d21048f9d6b>:18: DataConversionWarning: A column-vector y was
passed when a 1d array was expected. Please change the shape of y to (n_samples,
), for example using ravel().
  clf.fit(x_train, y_train)
```

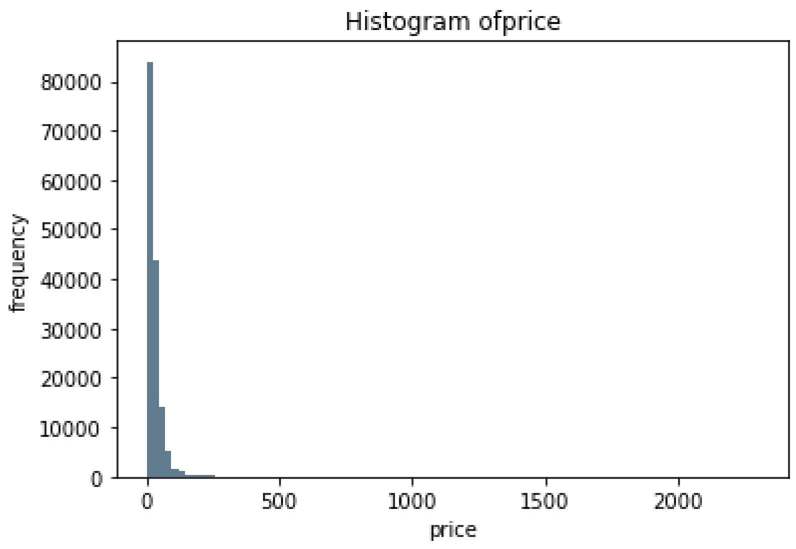
```
Unnamed: 0         0
country            5
```

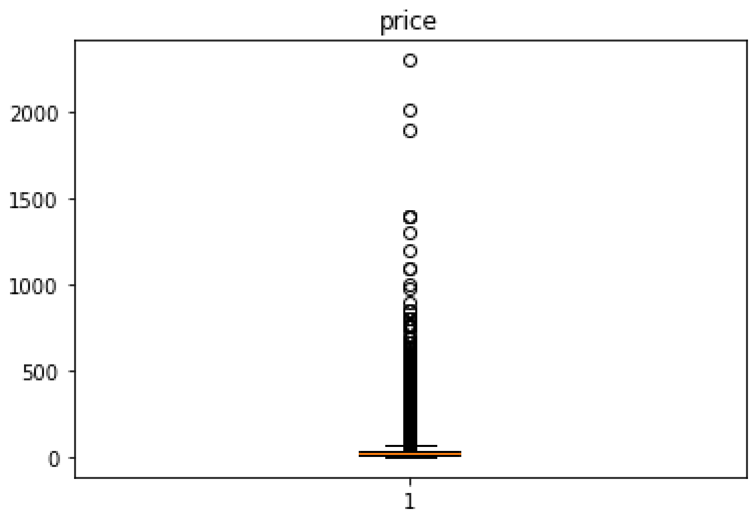
```
description        0
designation    45735
points             0
price              0
province           5
region_1       25060
region_2       89977
variety            0
winery             0
dtype: int64
price five number:
4.0 15.0 24.0 40.0 2300.0
points five number:
80 86.0 88.0 90.0 100
```

```
原数据的points属性的五数概括为：[80,86,88,90,100]，数据处理之后的五数概括为
[80,86,88,90,100]。
原数据的price属性的五数概括为：[4.0 16.0 24.0 40.0 2300.0]，数据处理之后的五数概括为[4 15
24 40 2300]。
```

由于只对price数据进行了处理，因此只对price数据进行可视化，对数据处理之后price的直方图与盒图的代码及结果为：

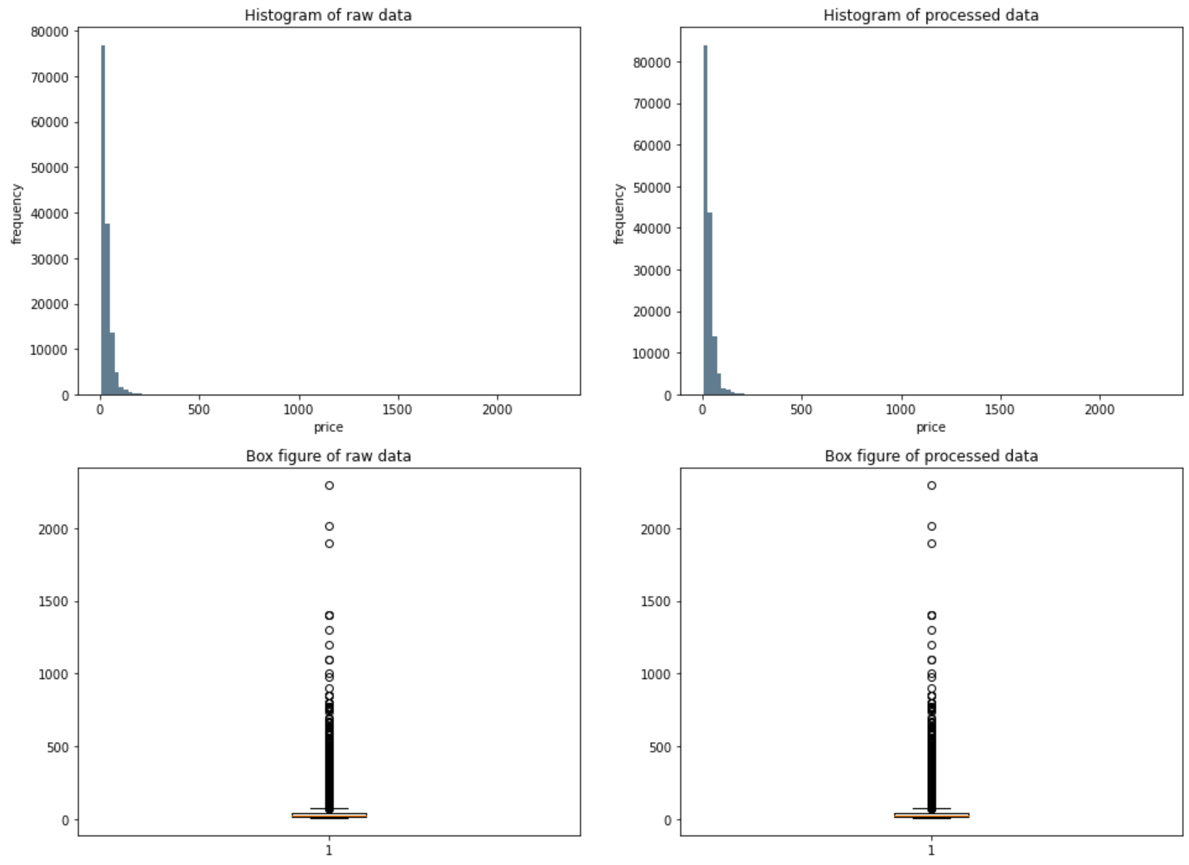
```
att = ['price']
show_hist(data_lost4, att, 100)
show_boxplot(data_lost4,att)
```





与原数据比较代码及结果为：

```
att = 'price'
subplot_show(data, att, data_lost4, att, 100, 'raw data', 'processed data')
```



wine reviews第二个数据集描述及预处理

wine reviews数据集中包含了两个文件，即winemag-data_first150k.csv与winemag-data-130k.csv。本章仅仅对winemag-data-130k.csv数据集进行处理。这上述数据集中统计了130k条葡萄酒的信息，即生产国家、评价、分数、价格与生产省份等。其中标称属性有id, country, designation, province, region_1, region_2, title, taster_name, taster_twitter_handle, variety和winery。数值属性有points和price。本章节将分别对wine reviews 的两个数据集进行描述与预处理。

数据分析

本节将统计标称属性取值的频数与对数值属性进行五数概括与统计缺失值的个数。

数据摘要

- 标称属性
  统计标称属性所有可能取值的频数并保存到了本地的txt文件中。由于取值的可能性太多，在这里将不再进行展示了。代码如下：

```
data_name = r'C:\Users\Xiaomeng Fan\Desktop\研究生的各种作业\数据挖掘\作业四
\wine_2\winemag-data-130k-v2.csv'
txt=open(r'C:\Users\Xiaomeng Fan\Desktop\研究生的各种作业\数据挖掘\作业四
\wine_2\wine_reviews_result1.txt','w',encoding = 'utf-8')
problem_1 = 0
file = pd.read_csv(data_name)
data = pd.DataFrame(file)

print(file.columns)
atts = file.columns
num=0
if problem_1:
  for i in atts[1:]:
    print(i)
    if i!= 'points' and i!= 'price' and i!='description':
        dict_temp = {}
        for j in file.loc[:,i]:
            if j in dict_temp.keys():
                dict_temp[j]+=1
            else:
                dict_temp[j]=1
                num+=1

        # print(dict_temp)
        txt.write(i+'\n')
        txt.write(str(dict_temp))
        txt.write('\n')
        print(num)
  txt.close()
```

```
Index(['Unnamed: 0', 'country', 'description', 'designation', 'points',
       'price', 'province', 'region_1', 'region_2', 'taster_name',
       'taster_twitter_handle', 'title', 'variety', 'winery'],
      dtype='object')
```

- 数值属性
  对price与points数值属性进行五数概括及统计缺失值的个数。我们将price与points数据中的缺失值剔除出去，再对清理之后的数据进行五数概括。对price的数据处理代码如下：

```
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.nanpercentile(nums,25)
    Median=np.nanmedian(nums)
    Q3=np.nanpercentile(nums,75)



    return Minimum,Q1,Median,Q3,Maximum

price_num_list = []
att_pool = ['price']
# for att in att_pool:

#   for point in file.loc[:,att]:
#    if isinstance(point,int) or isinstance(point,float):
#     price_num_list.append(point)
#    else:
#     print(point)

data_drop = data.copy()
data_drop = data.dropna(subset = att_pool).loc[:,att_pool[0]].values
min_temp, Q1, median, Q3, max_temp = fiveNumber(data_drop)
print(att_pool[0]+' five number:')
print(min_temp,Q1,median,Q3,max_temp)
price_np = np.array(data_drop)
```

```
price five number:
4.0 17.0 25.0 42.0 3300.0
```

对points的数据处理代码如下：

```
points_num_list = []
att_pool = ['points']
for att in att_pool:

  for point in file.loc[:,att]:
   if isinstance(point,int) or isinstance(point,float):
    points_num_list.append(point)
   else:
    print(point)
points_np = np.array(points_num_list)
min_temp, Q1, median, Q3, max_temp = fiveNumber(points_num_list)
print(att_pool[0]+' five number:')
print(min_temp,Q1,median,Q3,max_temp)
```

```
points five number:
80 86.0 88.0 91.0 100
```

对空缺的数据进行统计的代码及结果如下：

```
print(data.isnull().sum())
```

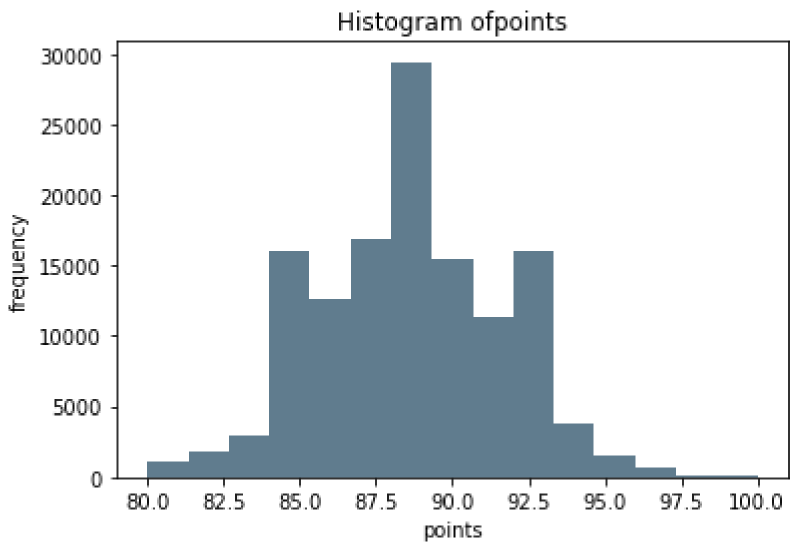
```
Unnamed: 0                  0
country                    63
description                 0
```

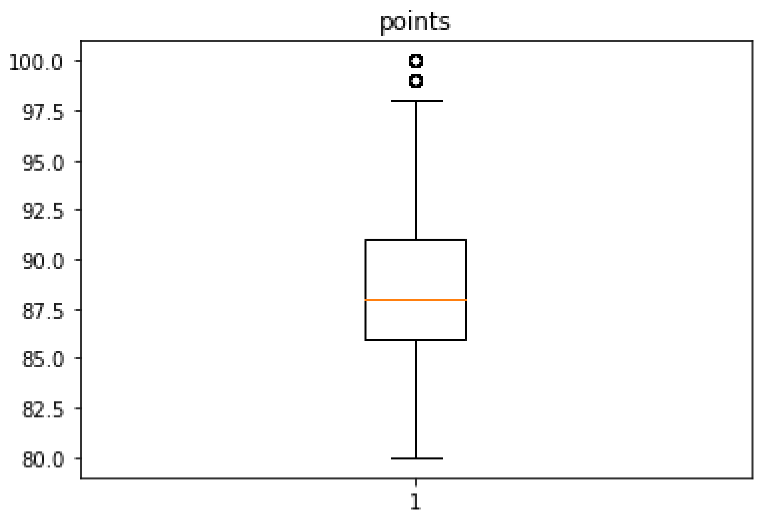
```
designation              37465
points                       0
price                     8996
province                    63
region_1                 21247
region_2                 79460
taster_name              26244
taster_twitter_handle    31213
title                        0
variety                      1
winery                       0
dtype: int64
```

数据可视化

- 直方图
  分别对points与price的数值属性绘制直方图。由于数值属性的数据存在缺失数据，因此我们需要首先将缺失数据剔除再进行相关可视化。对points绘制直方图与盒图的代码及结果如下：

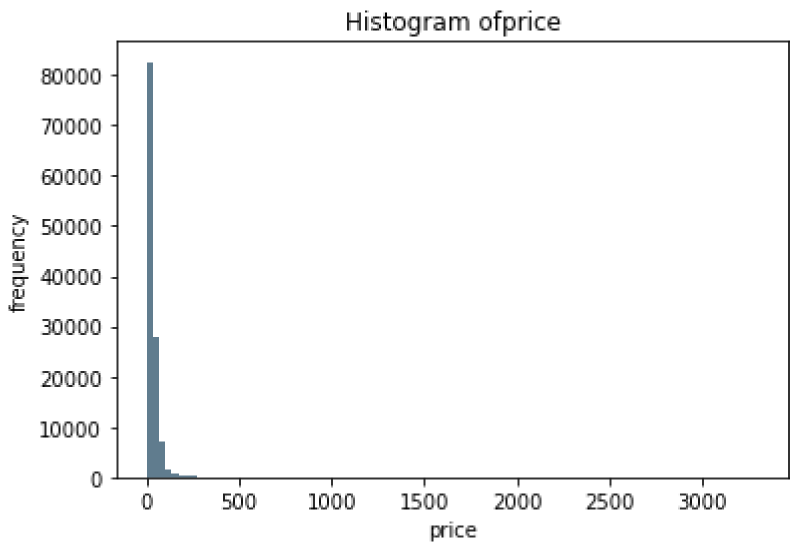
```python
def show_boxplot(data_temp, att_temp):

    data_temp1 = data_temp.copy()

    for i in att_temp:
        data_i = data_temp1.dropna(subset=[i])
        att_i = data_i.loc[:,i].values
        plt.boxplot(att_i, notch=False, sym='o', vert=True)
        plt.title(i)
        plt.show()


def show_hist(data_temp, att_temp, bin_numbers):
    data_temp1 = data_temp.copy()

    for i in att_temp:
        data_i = data_temp1.dropna(subset=[i])
        att_i = data_i.loc[:,i].values
        plt.hist(att_i, bins = bin_numbers, color = '#607c8e', rwidth=5)
        plt.xlabel(i)
        plt.ylabel('frequency')
        plt.title('Histogram of'+ i )
        plt.show()




att = ['points']
show_hist(data, att, 15)
show_boxplot(data,att)
```

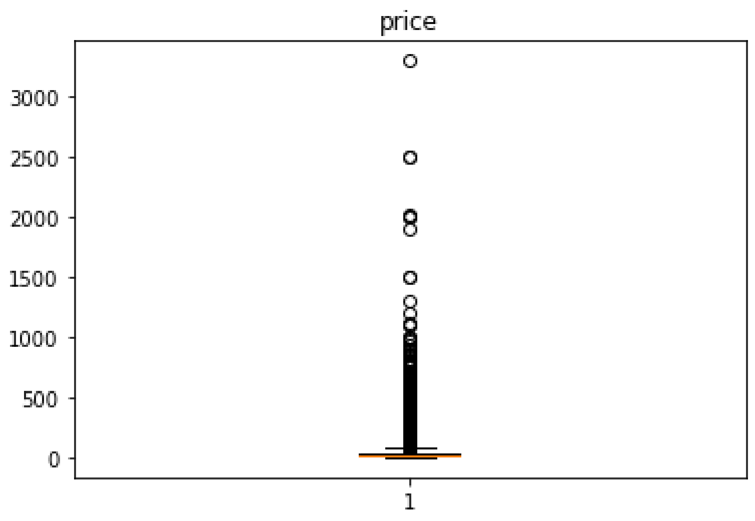




对price数据属性绘制直方图与盒图的代码与结果如下：

```
att = ['price']
show_hist(data, att,100)
show_boxplot(data,att)
```





由此可以看出，points的数据是接近于正太分布的，price的数据分布却不正常，主要表现为离群点很多，并且离群点的数值相较于平均值、众数与中尉数的差距很大。

数据缺失处理

观察数据集中缺失数据，分析其缺失的原因，分别使用下列四种策略对缺失值进行处理

将缺失部分剔除

我们将所有缺失数据全部剔除并保存。对处理之后的数据五数概括的代码及结果为：

```python
def lost_edition1():
    data_lost1 = data.copy()
    data_lost1 = data_lost1.dropna()
    return data_lost1

save_or_not = False
data_lost1 = lost_edition1()
print(data_lost1.isnull().sum())
if save_or_not:
    data_lost1.to_csv('data_delete.csv')
```

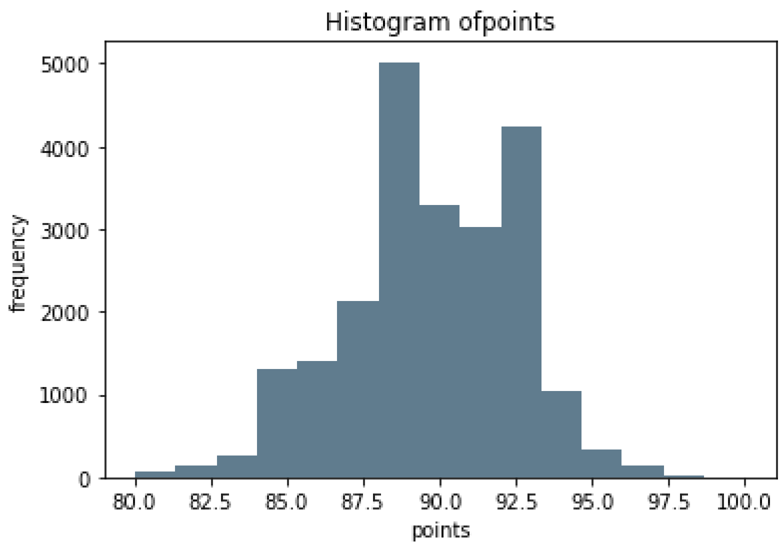
```
price_value = data_lost1.loc[:,'price'].values
points_value  = data_lost1.loc[:,'points'].values

min_temp, Q1, median, Q3, max_temp = fiveNumber(price_value)
print('price five number:')
print(min_temp,Q1,median,Q3,max_temp)

min_temp, Q1, median, Q3, max_temp = fiveNumber(points_value)
print('points five number:')
print(min_temp,Q1,median,Q3,max_temp)
```

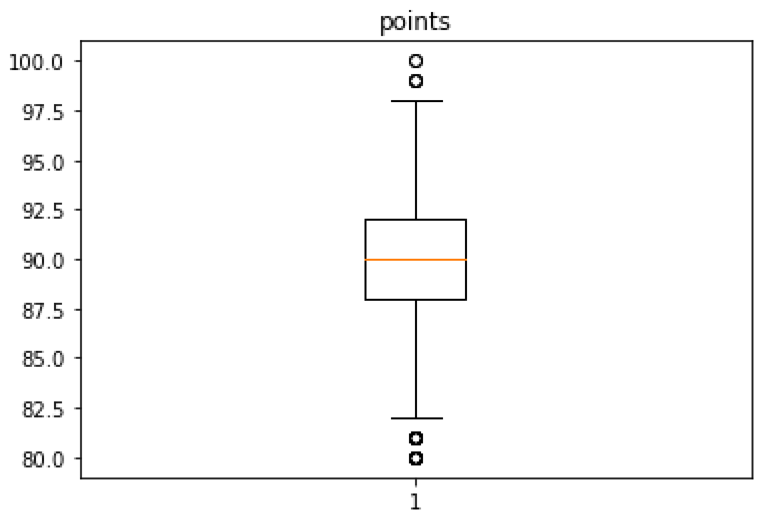
```
Unnamed: 0               0
country                  0
description              0
designation              0
points                   0
price                    0
province                 0
region_1                 0
region_2                 0
taster_name              0
taster_twitter_handle    0
title                    0
variety                  0
winery                   0
dtype: int64
price five number:
4.0 25.0 36.0 50.0 2013.0
points five number:
80 88.0 90.0 92.0 100
```

原数据的points属性的五数概括为：[80 86.0 88.0 91.0 100], 数据处理之后的五数概括为[80 88.0 90.0 92.0 100]。
原数据的price属性的五数概括为：[4.0 17.0 25.0 42.0 3300.0], 数据处理之后的五数概括为[4.0 25.0 36.0 50.0 2013.0]。

对数据处理之后points的直方图与盒图的代码及结果为：

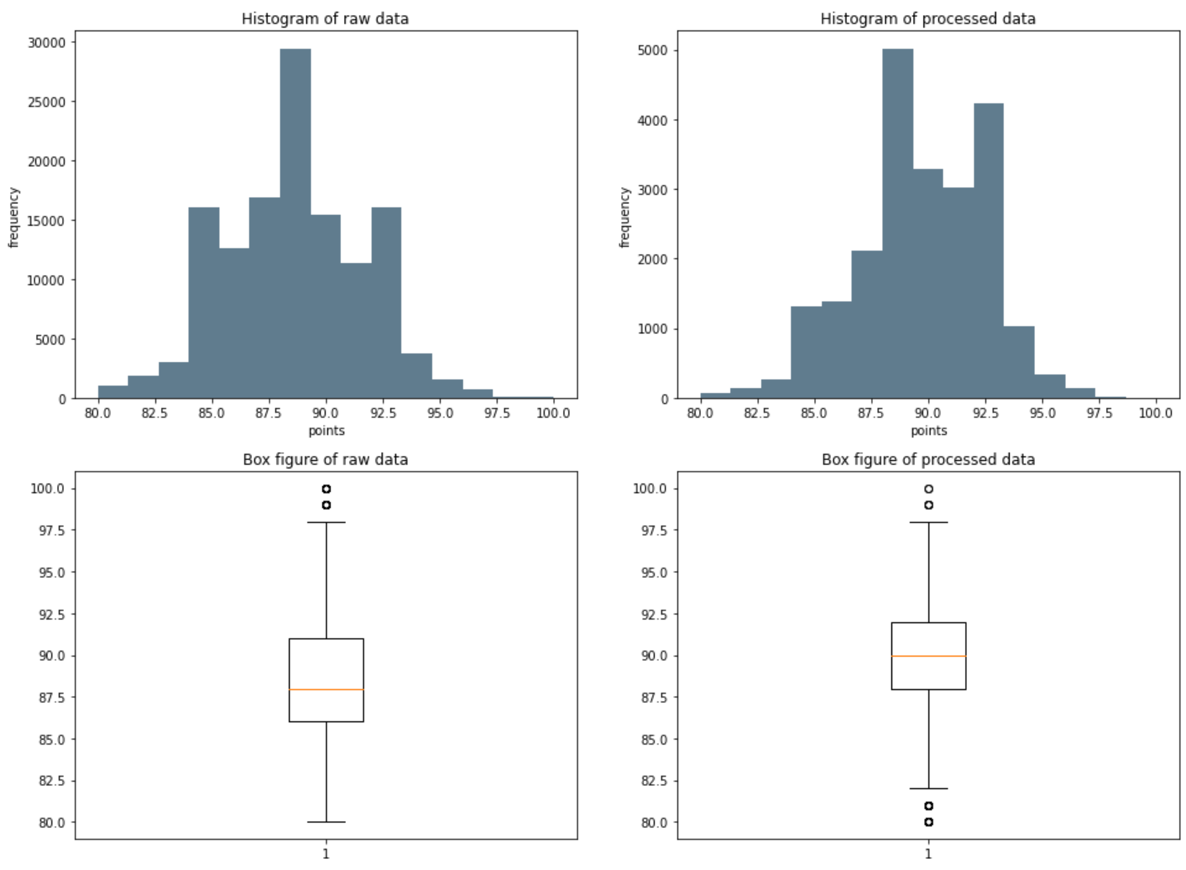
```
att = ['points']
show_hist(data_lost1, att, 15)
show_boxplot(data_lost1,att)
```





与原数据的对比结果如下：

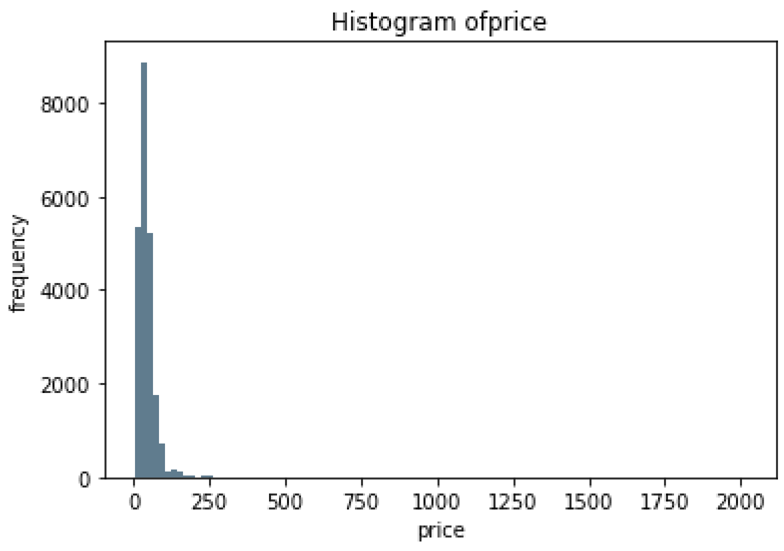
```
def subplot_show(data1, att1, data2, att2, bin_num, title1, title2):

    data1_temp = data1.copy()

    data_i = data1_temp.dropna(subset=[att1])
    data1_att = data_i.loc[:,att1].values
    data2_att = data2.loc[:,att2].values
    plt.figure(figsize=(16, 12))
    plt.subplot(221)
    plt.hist(data1_att,bins = bin_num, color = '#607c8e', rwidth=5)
    plt.xlabel(att1)
    plt.ylabel('frequency')
    plt.title('Histogram of '+ title1 )

    plt.subplot(222)
    plt.hist(data2_att,bins = bin_num, color = '#607c8e', rwidth=5)
    plt.xlabel(att2)
    plt.ylabel('frequency')
    plt.title('Histogram of '+ title2 )
    plt.subplot(223)
    plt.boxplot(data1_att, notch=False, sym='o', vert=True)

    plt.title('Box figure of '+title1)

    plt.subplot(224)
    plt.boxplot(data2_att, notch=False, sym='o', vert=True)
    plt.title('Box figure of '+title2)

    plt.show()
att = 'points'
subplot_show(data, att, data_lost1, att, 15, 'raw data', 'processed data')
```

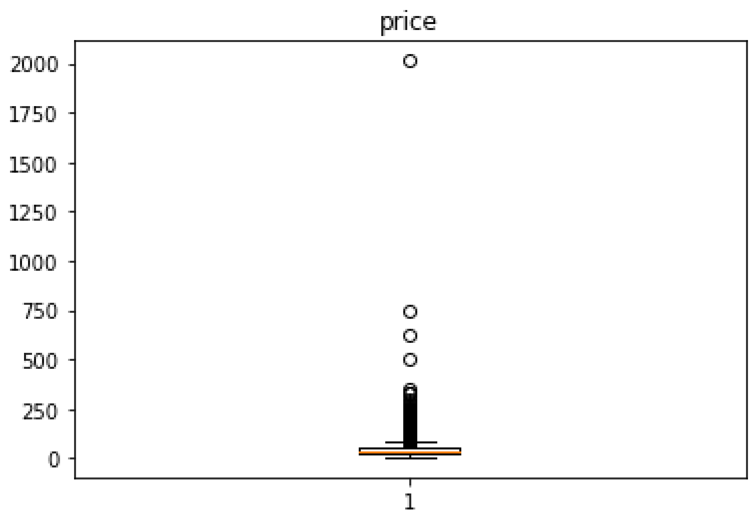


由此可以看出，在去掉全部的缺失数据之后，points数据里的离群点减少了，并且数据的更趋向于正态分布。

对数据处理之后price的直方图与盒图的代码及结果为：

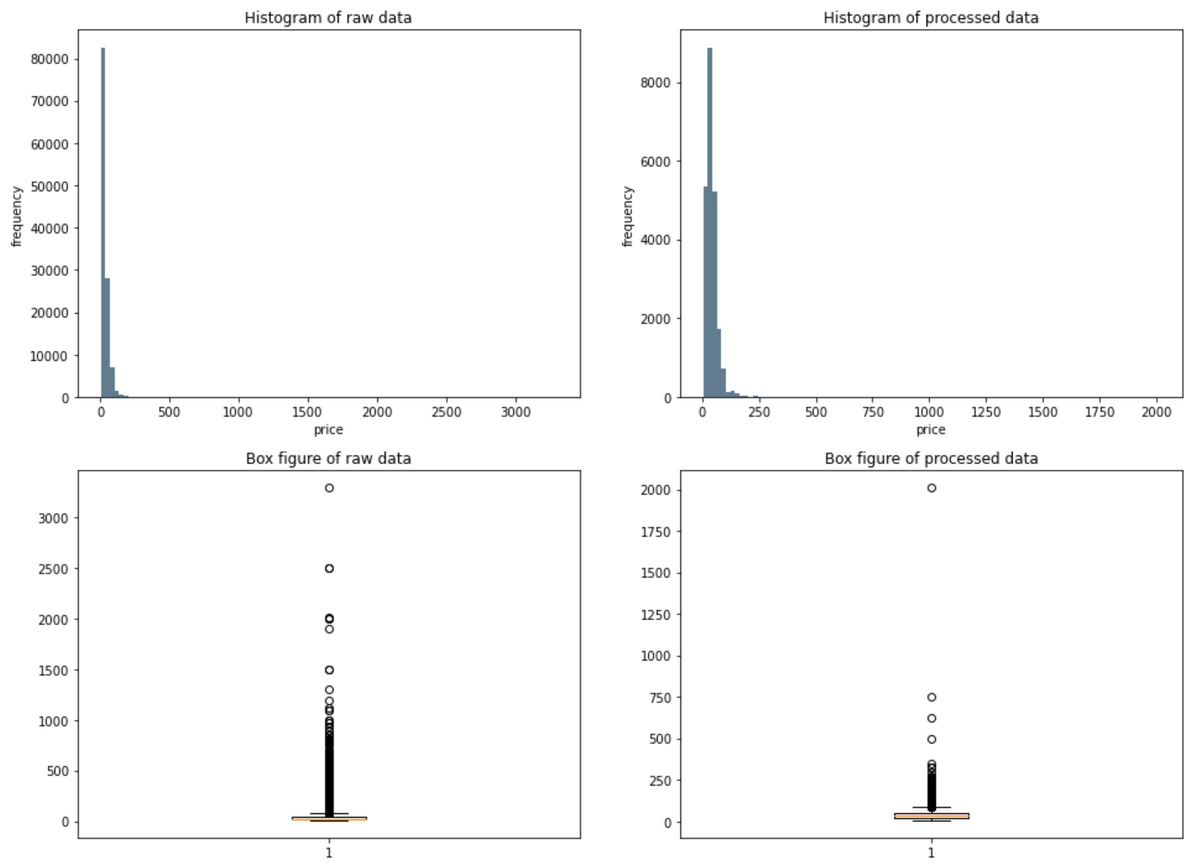
```
att = ['price']
show_hist(data_lost1, att, 100)
show_boxplot(data_lost1,att)
```





与原数据比较的代码及结果为：

```
att = 'price'
subplot_show(data, att, data_lost1, att, 100, 'raw data', 'processed data')
```



由此可以看出，在去掉全部的缺失数据之后，price数据里的离群点减少了，并且数据的更趋向于正态分布。

用最高频率值来填补缺失值

我们首先计算price属性最高频率的数值，将空缺值替换为该值并保存。对处理之后的数据五数概括的代码及结果为：

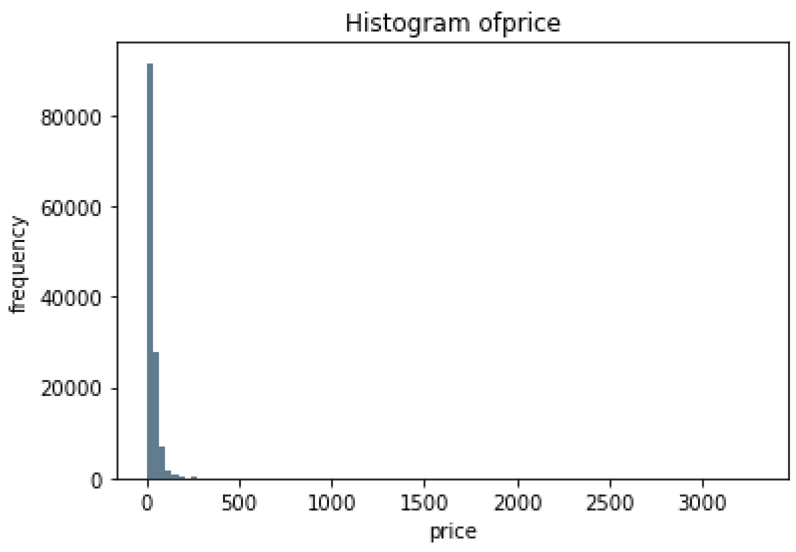
```
def lost_edition2(price_np):

    data_lost2 = data.copy()
    price_mode = mode(price_np)

    mode_temp = price_mode[0].tolist()[0]
    data_lost2['price'] = data_lost2['price'].fillna(mode_temp)
    return data_lost2


save_or_not = False
data_lost2 = lost_edition2(price_np)
print(data_lost2.isnull().sum())
if save_or_not:
    data_lost2.to_csv('data_mode.csv')


price_value_2 = data_lost2.loc[:,'price'].values
points_value_2  = data_lost2.loc[:,'points'].values

min_temp, Q1, median, Q3, max_temp = fiveNumber(price_value_2)
print('price five number:')
print(min_temp,Q1,median,Q3,max_temp)

min_temp, Q1, median, Q3, max_temp = fiveNumber(points_value_2)
print('points five number:')
print(min_temp,Q1,median,Q3,max_temp)
```

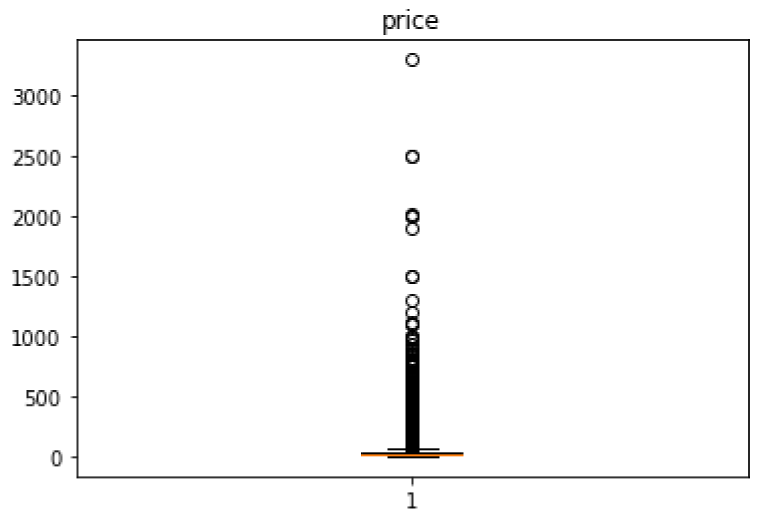
```
Unnamed: 0                 0
country                   63
description                0
designation            37465
points                     0
price                      0
province                  63
region_1               21247
region_2               79460
taster_name            26244
taster_twitter_handle  31213
title                      0
variety                    1
winery                     0
dtype: int64
price five number:
4.0 18.0 25.0 40.0 3300.0
points five number:
80 86.0 88.0 91.0 100
```

原数据的points属性的五数概括为：[80,86,88,90,100], 数据处理之后的五数概括为[80 86.0 88.0 91.0 100]。
原数据的price属性的五数概括为：[4.0 16.0 24.0 40.0 2300.0], 数据处理之后的五数概括为[4.0 18.0 25.0 40.0 3300.0]。

由于只对price数据进行了处理，因此只对price数据进行可视化，对数据处理之后price的直方图与盒图的代码及结果为：

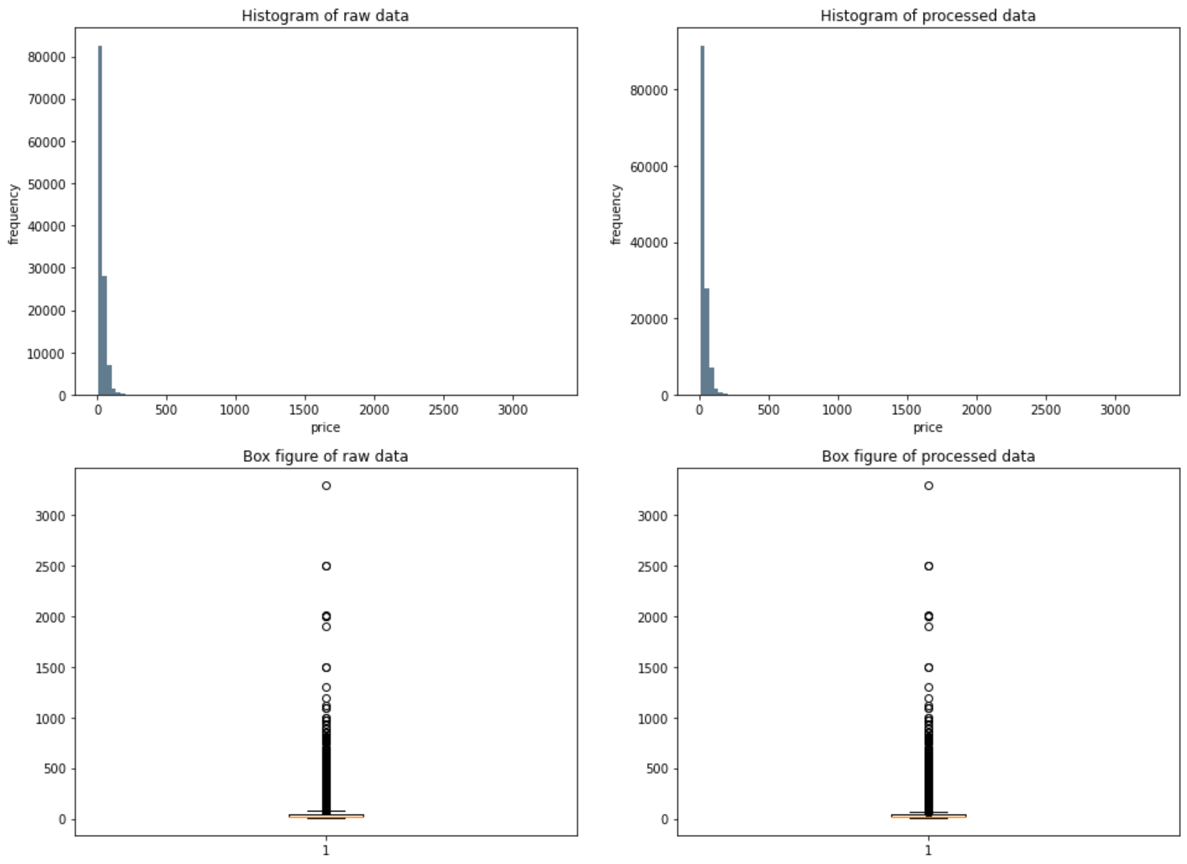
```
att = ['price']
show_hist(data_lost2, att, 100)
show_boxplot(data_lost2,att)
```





与原数据比较代码及结果为：

```
att = 'price'
subplot_show(data, att, data_lost2, att, 100, 'raw data', 'processed data')
```



通过属性的相关关系来填补缺失值

我们使用随机森林的方法对price属性的缺失数据进行补全，其代码及结果如下：

```
def set_missing_prices():

    data_lost3 = data.copy()
    # 把已有的数值型特征取出来丢进Random Forest Regressor中
    price_df = data_lost3[['price', 'points']]

    # 乘客分成已知年龄和未知年龄两部分
    known_price = price_df[price_df.price.notnull()].values
    unknown_price = price_df[price_df.price.isnull()].values

    # y即目标年龄
    y = known_price[:, 0]

    # X即特征属性值
    X = known_price[:, 1:]

    # fit到RandomForestRegressor之中
    rfr = sklearn.ensemble.RandomForestRegressor(random_state=0,
n_estimators=2000, n_jobs=-1)
    rfr.fit(X, y)

    # 用得到的模型进行未知年龄结果预测
    predictedprice = rfr.predict(unknown_price[:, 1:])
#     print predictedAges
    # 用得到的预测结果填补原缺失数据
    data_lost3.loc[ (data_lost3.price.isnull()), 'price' ] = predictedprice

    return data_lost3

save_or_not = False
data_lost3 = set_missing_prices()
print(data_lost2.isnull().sum())
if save_or_not:
    data_lost3.to_csv('data_forest.csv')


price_value_3 = data_lost3.loc[:,'price'].values
points_value_3  = data_lost3.loc[:,'points'].values

min_temp, Q1, median, Q3, max_temp = fiveNumber(price_value_3)
print('price five number:')
print(min_temp,Q1,median,Q3,max_temp)

min_temp, Q1, median, Q3, max_temp = fiveNumber(points_value_3)
print('points five number:')
print(min_temp,Q1,median,Q3,max_temp)
```

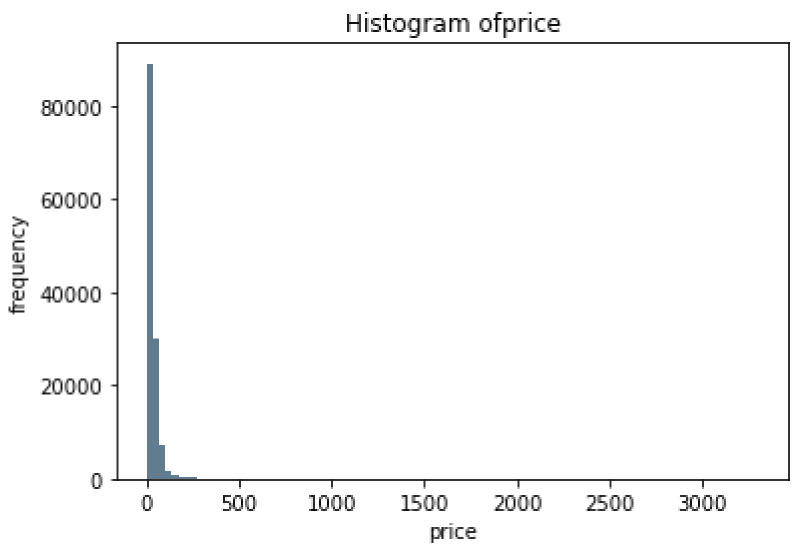
```
Unnamed: 0                   0
country                     63
description                  0
designation              37465
points                       0
price                        0
province                    63
region_1                 21247
region_2                 79460
taster_name              26244
taster_twitter_handle    31213
title                        0
```

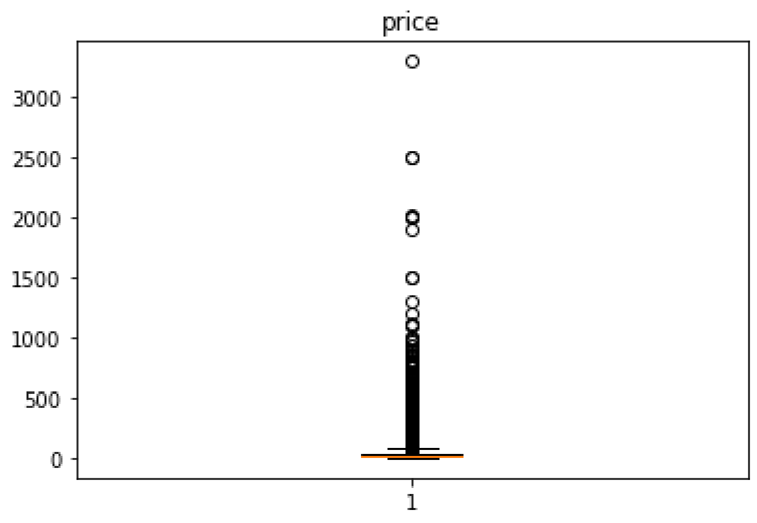
```
variety                    1
winery                     0
dtype: int64
price five number:
4.0 18.0 26.0 42.0 3300.0
points five number:
80 86.0 88.0 91.0 100
```

原数据的points属性的五数概括为：[80,86,88,90,100], 数据处理之后的五数概括为[80 86.0 88.0 91.0 100]。
原数据的price属性的五数概括为：[4.0 16.0 24.0 40.0 2300.0], 数据处理之后的五数概括为[4.0 18.0 26.0 42.0 3300.0]。

由于只对price数据进行了处理，因此只对price数据进行可视化，对数据处理之后price的直方图与盒图的代码及结果为：

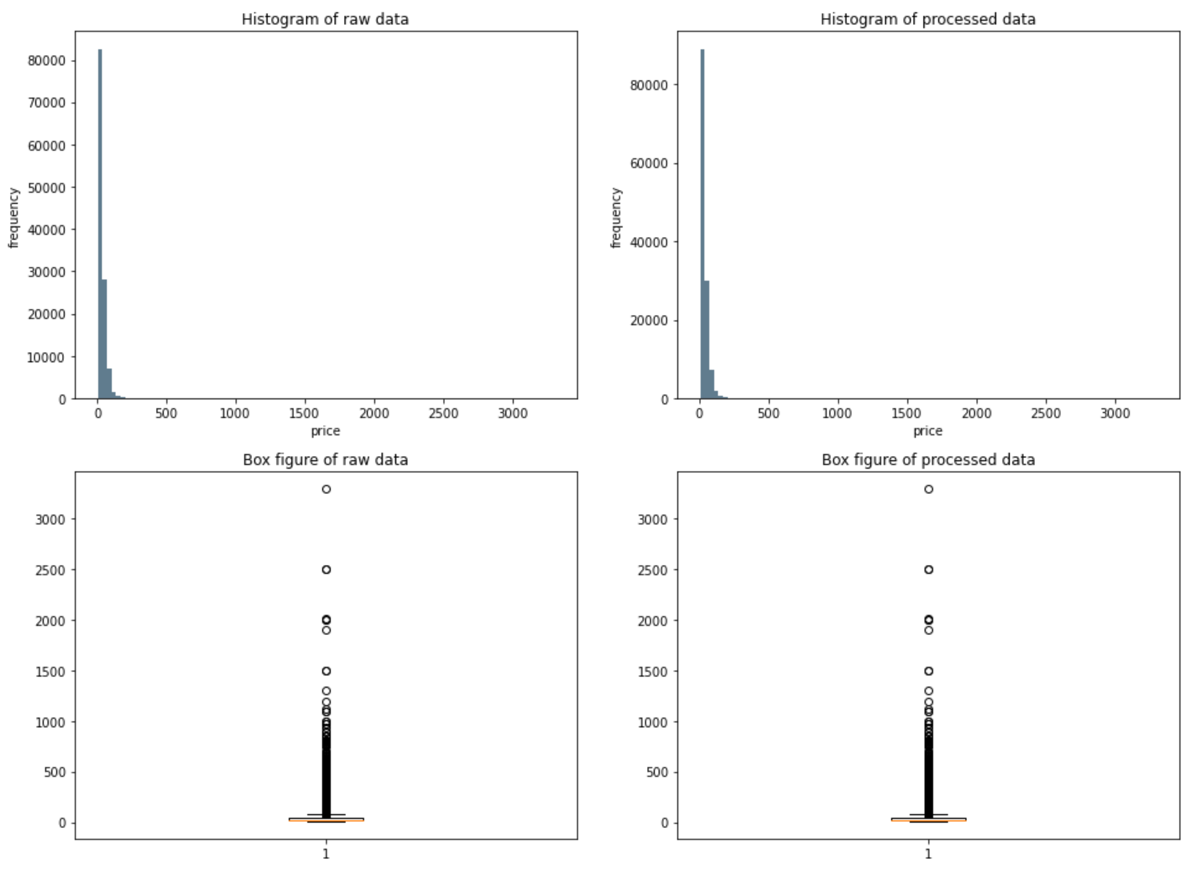
```
att = ['price']
show_hist(data_lost3, att, 100)
show_boxplot(data_lost3,att)
```





与原数据比较代码及结果为：

```
att = 'price'
subplot_show(data, att, data_lost3, att, 100, 'raw data', 'processed data')
```



通过数据对象之间的相似性来填补缺失值

我们使用KNN的方法对price属性的缺失数据进行补全，其代码及结果如下：

```
def knn_missing_filled(k = 3, dispersed = True):

    data_lost4 = data.copy()
    # x_train = data_lost4[data_lost4.price.notnull()]
['points'].values.reshape(-1,1)
    # y_train = data_lost4[data_lost4.price.notnull()]
['price'].values.reshape(-1,1)

    x_train =
data_lost4[data_lost4.price.notnull()].loc[:,'points'].values.reshape(-1,1)
    y_train =
data_lost4[data_lost4.price.notnull()].loc[:,'price'].values.reshape(-1,1)
    print(len(x_train))
    print(len(y_train))
    test = data_lost4[data_lost4.price.isnull()]['points'].values.reshape(-1,1)

    if dispersed:
        clf = KNeighborsClassifier(n_neighbors = k, weights = "distance")
    else:
        clf = KNeighborsRegressor(n_neighbors = k, weights = "distance")
    clf.fit(x_train, y_train)

    data_lost4.loc[ (data_lost4.price.isnull()), 'price' ] = clf.predict(test)

    return data_lost4



save_or_not = False
data_lost4 = knn_missing_filled()
print(data_lost2.isnull().sum())
if save_or_not:
    data_lost4.to_csv('data_knn.csv')


price_value_4 = data_lost4.loc[:,'price'].values
points_value_4  = data_lost4.loc[:,'points'].values

min_temp, Q1, median, Q3, max_temp = fiveNumber(price_value_4)
print('price five number:')
print(min_temp,Q1,median,Q3,max_temp)

min_temp, Q1, median, Q3, max_temp = fiveNumber(points_value_4)
print('points five number:')
print(min_temp,Q1,median,Q3,max_temp)
```

```
120975
120975
```

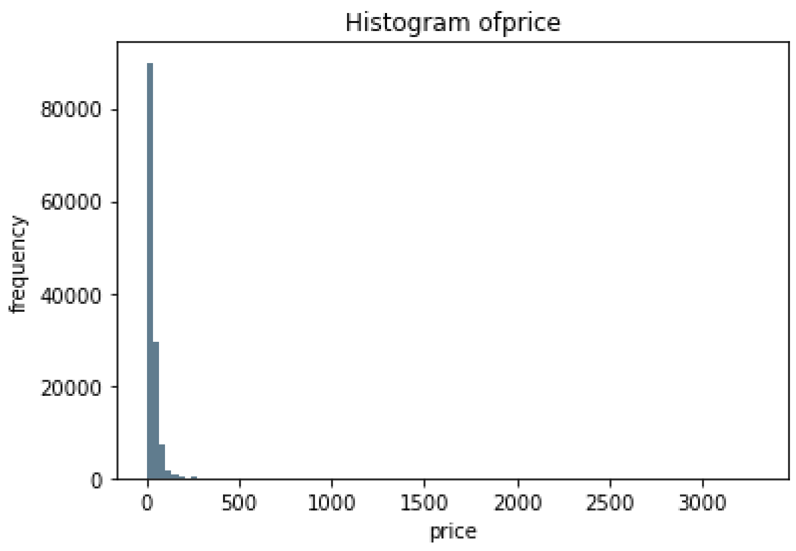
```
<ipython-input-82-1d21048f9d6b>:18: DataConversionWarning: A column-vector y was
passed when a 1d array was expected. Please change the shape of y to (n_samples,
), for example using ravel().
  clf.fit(x_train, y_train)
```

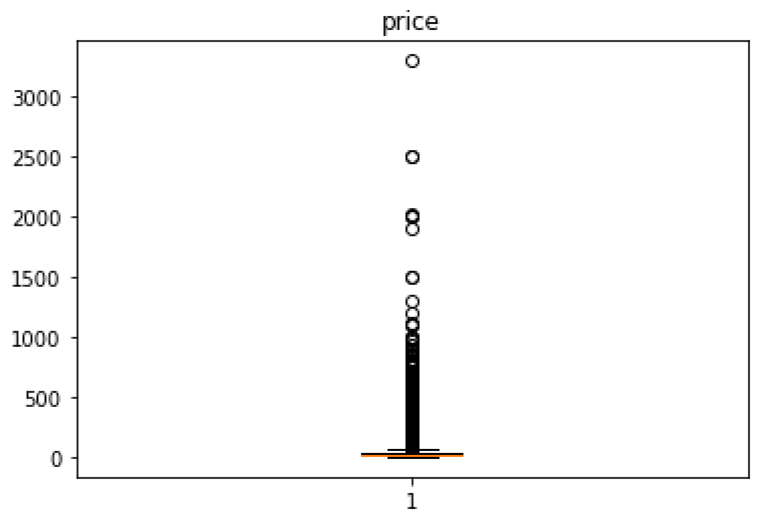
```
Unnamed: 0                   0
country                     63
description                  0
designation              37465
points                       0
price                        0
province                    63
region_1                 21247
region_2                 79460
taster_name              26244
taster_twitter_handle    31213
title                        0
variety                      1
winery                       0
dtype: int64
price five number:
4.0 17.0 25.0 40.0 3300.0
points five number:
80 86.0 88.0 91.0 100
```

原数据的points属性的五数概括为：[80,86,88,90,100], 数据处理之后的五数概括为[80 86.0 88.0 91.0 100]。
原数据的price属性的五数概括为：[4.0 16.0 24.0 40.0 2300.0], 数据处理之后的五数概括为[4.0 17.0 25.0 40.0 3300.0]。

由于只对price数据进行了处理，因此只对price数据进行可视化，对数据处理之后price的直方图与盒图的代码及结果为：

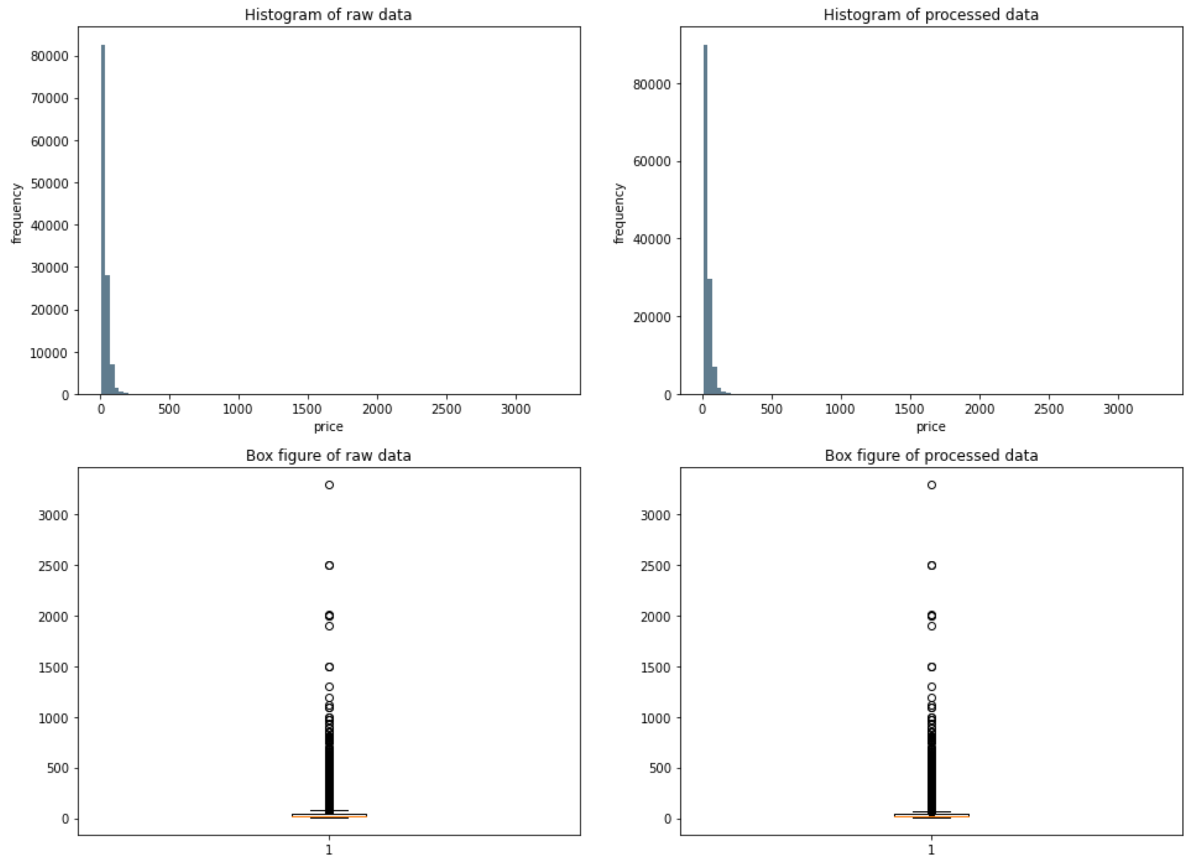
```
att = ['price']
show_hist(data_lost4, att, 100)
show_boxplot(data_lost4,att)
```





与原数据比较代码及结果为：

```
att = 'price'
subplot_show(data, att, data_lost4, att, 100, 'raw data', 'processed data')
```



Chicago Building Violations数据集描述及预处理

Chicago Building Violations数据包含了2006至今的违章行为。其共包含32列数据，即该数据集共包含32种属性。其中标称属性有id, country, designation, province, region_1, region_2, title, variety和winery。数值属性有'INSPECTION NUMBER', 'PROPERTY GROUP', 'Community Areas', 'Census Tracts','Wards','Historical Wards 2003-2015'。本章节将Chicago Building Violations数据集描述及预处理。

数据分析

本节将统计标称属性取值的频数与对数值属性进行五数概括与统计缺失值的个数。

数据摘要

- 标称属性
  统计标称属性所有可能取值的频数并保存到了本地的txt文件中。由于取值的可能性太多，在这里将不再进行展示了。代码如下：

```
data_name = r'C:\Users\Xiaomeng Fan\Desktop\研究生的各种作业\数据挖掘\作业四
\building\building-violations.csv'

txt=open(r'C:\Users\Xiaomeng Fan\Desktop\研究生的各种作业\数据挖掘\作业四
\building\building-violations_result1.txt','w',encoding = 'utf-8')
problem_1 = 0
file = pd.read_csv(data_name)
data = pd.DataFrame(file)
```

```
print(file.columns)
atts = file.columns
num=0
if problem_1:
  for i in atts[1:]:
    print(i)
    if i!= 'points' and i!= 'price' and i!='description':
        dict_temp = {}
        for j in file.loc[:,i]:
            if j in dict_temp.keys():
                dict_temp[j]+=1
            else:
                dict_temp[j]=1
                num+=1

        # print(dict_temp)
        txt.write(i+'\n')
        txt.write(str(dict_temp))
        txt.write('\n')
        print(num)
  txt.close()
```

```
Index(['ID', 'VIOLATION LAST MODIFIED DATE', 'VIOLATION DATE',
       'VIOLATION CODE', 'VIOLATION STATUS', 'VIOLATION STATUS DATE',
       'VIOLATION DESCRIPTION', 'VIOLATION LOCATION',
       'VIOLATION INSPECTOR COMMENTS', 'VIOLATION ORDINANCE', 'INSPECTOR ID',
       'INSPECTION NUMBER', 'INSPECTION STATUS', 'INSPECTION WAIVED',
       'INSPECTION CATEGORY', 'DEPARTMENT BUREAU', 'ADDRESS', 'STREET NUMBER',
       'STREET DIRECTION', 'STREET NAME', 'STREET TYPE', 'PROPERTY GROUP',
       'SSA', 'LATITUDE', 'LONGITUDE', 'LOCATION', 'Community Areas',
       'Zip Codes', 'Boundaries - ZIP Codes', 'Census Tracts', 'Wards',
       'Historical Wards 2003-2015'],
      dtype='object')
```

- 数值属性
  对'INSPECTION NUMBER', 'PROPERTY GROUP', 'Community Areas', 'Census Tracts','Wards','Historical Wards 2003-2015'数值属性进行五数概括及统计缺失值的个数。我们将INSPECTION NUMBER', 'PROPERTY GROUP', 'Community Areas', 'Census Tracts','Wards','Historical Wards 2003-2015'数据中的缺失值剔除出去，再对清理之后的数据进行五数概括。代码如下，由于数据集过大，无法对标称属性的数据进行统计，所以不再进行展示。

```
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.nanpercentile(nums,25)
    Median=np.nanmedian(nums)
    Q3=np.nanpercentile(nums,75)



    return Minimum,Q1,Median,Q3,Maximum
```

```
att_num = ['INSPECTION NUMBER', 'PROPERTY GROUP', 'Community Areas', 'Census
Tracts','Wards','Historical Wards 2003-2015']
complete_list = ['INSPECTION NUMBER', 'PROPERTY GROUP']
uncomplete_list = ['Community Areas', 'Census Tracts','Wards','Historical Wards
2003-2015']
att_nominal = [ 'VIOLATION LOCATION', 'VIOLATION ORDINANCE', 'INSPECTOR ID',
'INSPECTION STATUS', 'INSPECTION CATEGORY', 'DEPARTMENT BUREAU', 'STREET
DIRECTION', 'STREET NAME', 'STREET TYPE']

problem_1 = 0
if problem_1:
  for i in att_nominal :
    print(i)

    dict_temp = {}
    for j in data.loc[:,i]:
        if j in dict_temp.keys():
            dict_temp[j]+=1
        else:
            dict_temp[j]=1
            num+=1

        # print(dict_temp)
        txt.write(i+'\n')
        txt.write(str(dict_temp))
        txt.write('\n')
    break
        # print(num)
  txt.close()
```

```
INSPECTION NUMBER five number:
265575 2304416.0 10418746.0 11687280.0 13050915
PROPERTY GROUP five number:
1000 20560.0 154323.0 366984.0 677975
Community Areas five number:
1.0 24.0 36.0 58.0 77.0
Census Tracts five number:
1.0 179.0 374.0 572.0 801.0
Wards five number:
1.0 12.0 25.0 37.0 50.0
Historical Wards 2003-2015 five number:
1.0 14.0 28.0 41.0 53.0
```

对空缺的数据进行统计的代码及结果如下：

```
print(data.isnull().sum())
```

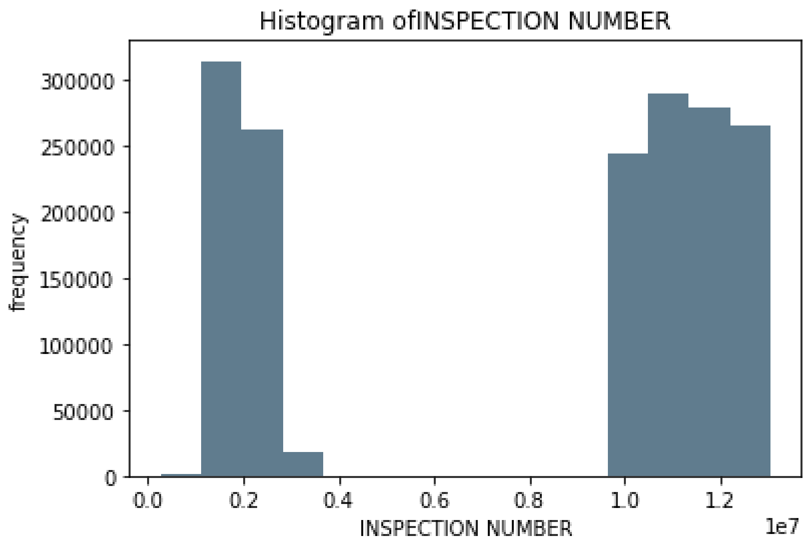
```
ID                                  0
VIOLATION LAST MODIFIED DATE        0
VIOLATION DATE                      0
VIOLATION CODE                      0
VIOLATION STATUS                    0
VIOLATION STATUS DATE         1036199
VIOLATION DESCRIPTION           10768
VIOLATION LOCATION             897282
VIOLATION INSPECTOR COMMENTS     175463
VIOLATION ORDINANCE               47581
INSPECTOR ID                          0
INSPECTION NUMBER                     0
INSPECTION STATUS                    16
INSPECTION WAIVED                     0
INSPECTION CATEGORY                   0
DEPARTMENT BUREAU                     0
ADDRESS                               0
STREET NUMBER                         0
STREET DIRECTION                      0
STREET NAME                           0
STREET TYPE                       13541
PROPERTY GROUP                        0
SSA                             1356267
LATITUDE                           1510
LONGITUDE                          1510
LOCATION                           1510
Community Areas                    2279
Zip Codes                          1510
Boundaries - ZIP Codes             2279
Census Tracts                      1545
Wards                              2279
Historical Wards 2003-2015         2279
dtype: int64
```

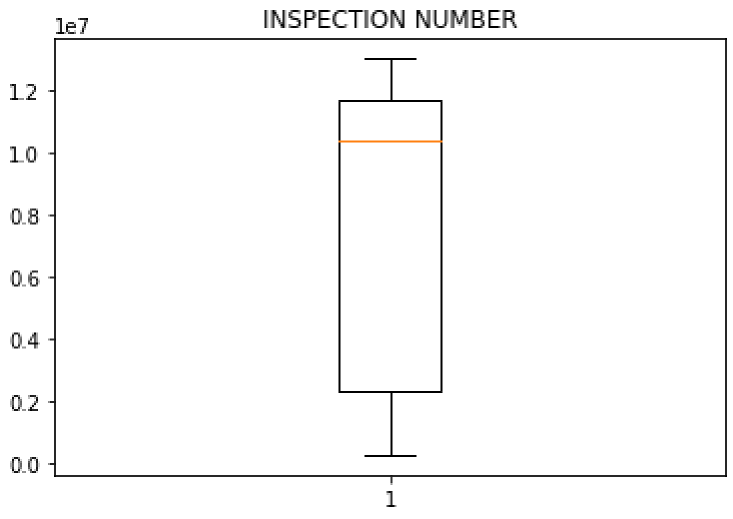
数据可视化

- 直方图
  分别对'INSPECTION NUMBER', 'PROPERTY GROUP', 'Community Areas', 'Census Tracts','Wards','Historical Wards 2003-2015'的数值属性绘制直方图。由于数值属性的数据存在缺失数据，因此我们需要首先将缺失数据剔除再进行相关可视化。对INSPECTION NUMBER绘制直方图与盒图的代码及结果如下：

```
def show_boxplot(data_temp, att_temp):

    data_temp1 = data_temp.copy()

    for i in att_temp:
        data_i = data_temp1.dropna(subset=[i])
        att_i = data_i.loc[:,i].values
        plt.boxplot(att_i, notch=False, sym='o', vert=True)
        plt.title(i)
        plt.show()


def show_hist(data_temp, att_temp, bin_numbers):
    data_temp1 = data_temp.copy()

    for i in att_temp:
        data_i = data_temp1.dropna(subset=[i])
        att_i = data_i.loc[:,i].values
        plt.hist(att_i, bins = bin_numbers, color = '#607c8e', rwidth=5)
        plt.xlabel(i)
        plt.ylabel('frequency')
        plt.title('Histogram of'+ i )
        plt.show()
```

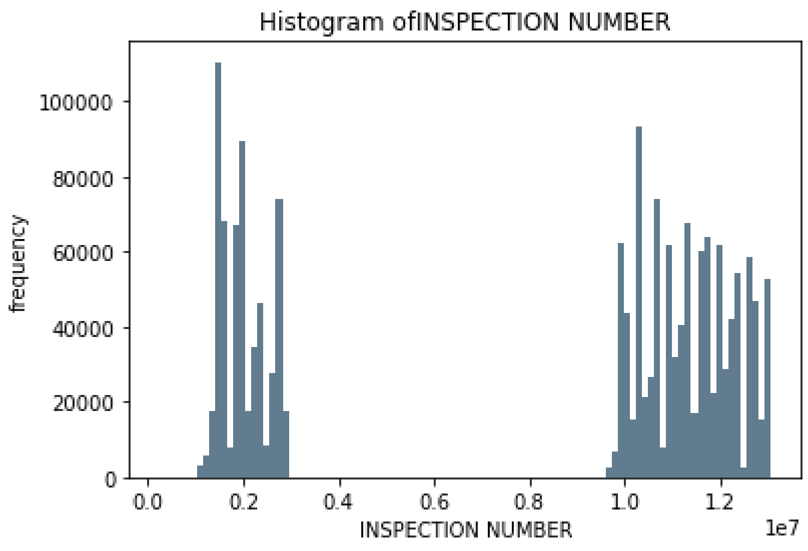
```
att = ['INSPECTION NUMBER']
show_hist(data, att, 15)
show_boxplot(data,att)
```

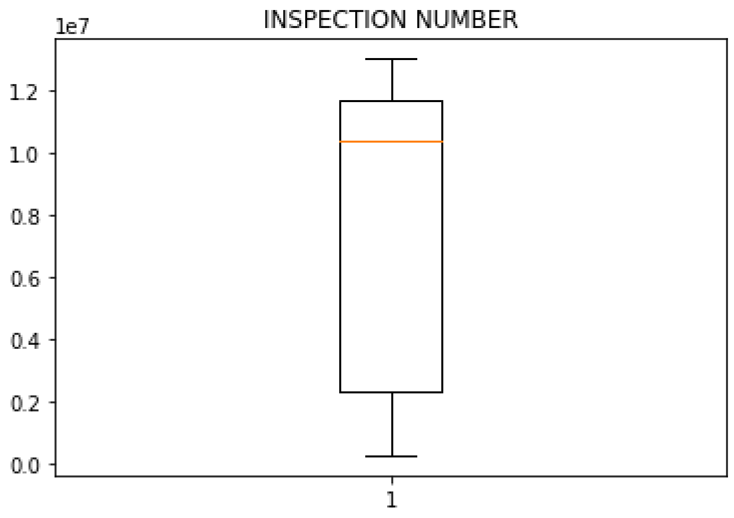




对INSPECTION NUMBER数据属性绘制直方图与盒图的代码与结果如下：

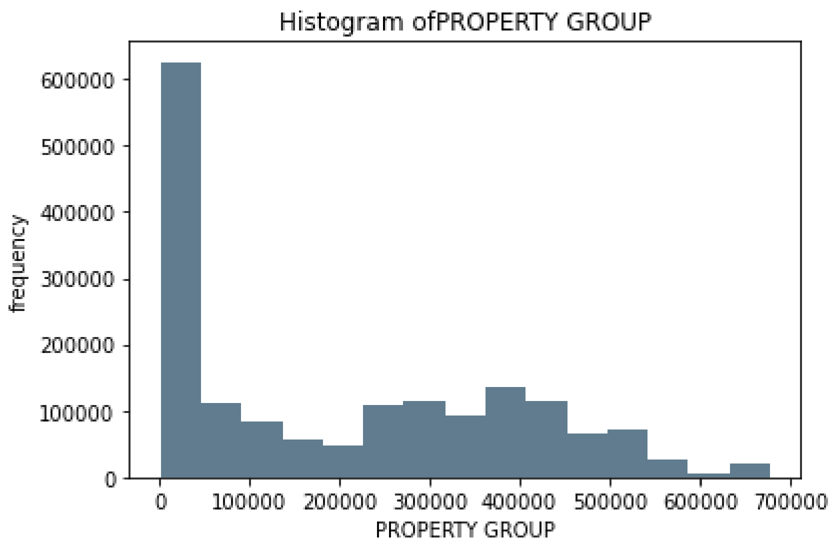
```
att = ['INSPECTION NUMBER']
show_hist(data, att,100)
show_boxplot(data,att)
```

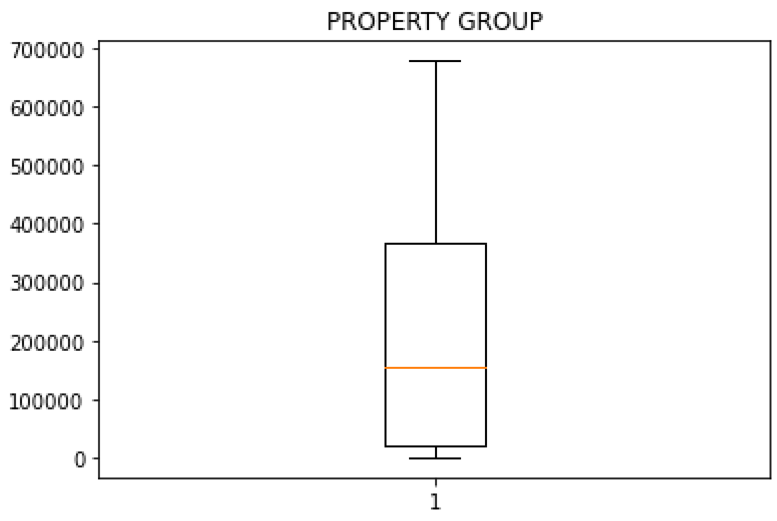




对 PROPERTY GROUP数据属性绘制直方图与盒图的代码与结果如下：

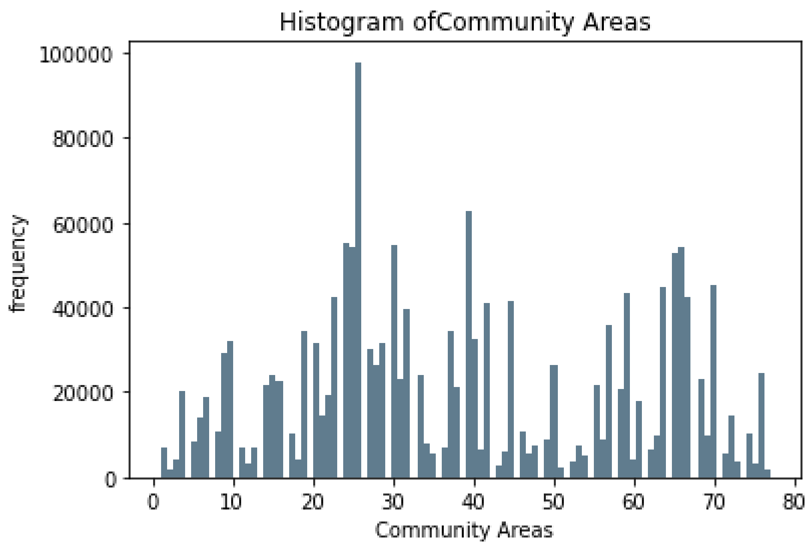
```
att = ['PROPERTY GROUP']
show_hist(data, att, 15)
show_boxplot(data,att)
```

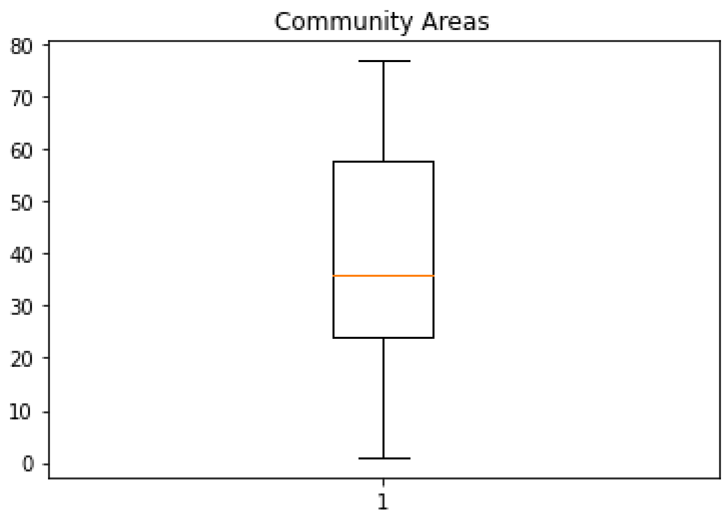




对Community Areas数据属性绘制直方图与盒图的代码与结果如下：

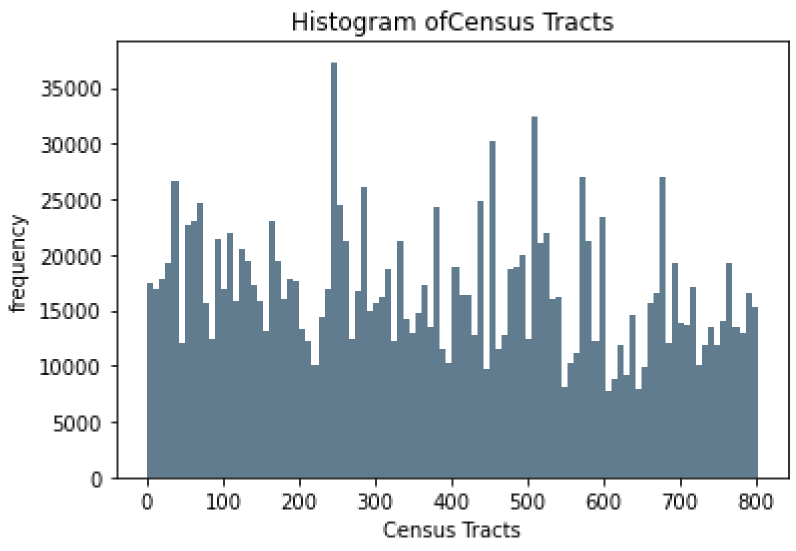
```
att = ['Community Areas']
show_hist(data, att,100)
show_boxplot(data,att)
```

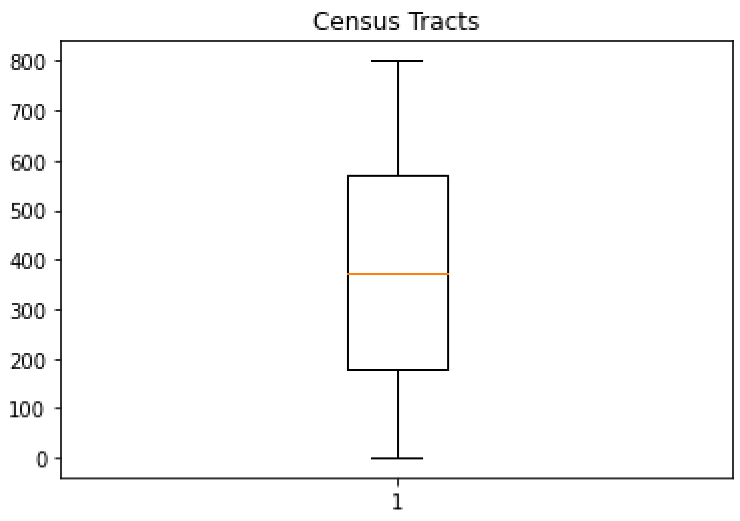




对Census Tracts数据属性绘制直方图与盒图的代码与结果如下：

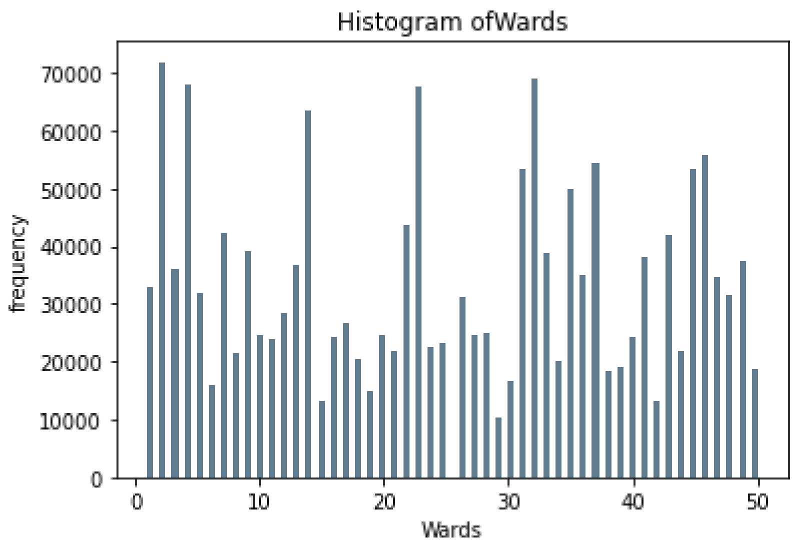
```
att = ['Census Tracts']
show_hist(data, att,100)
show_boxplot(data,att)
```

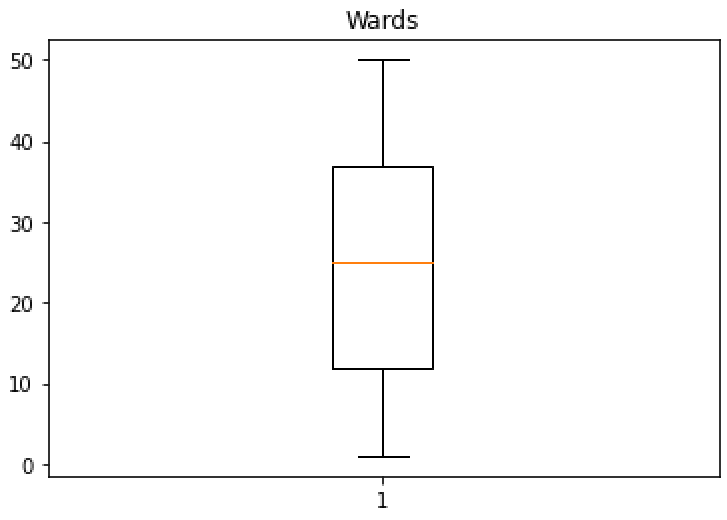




对Wards数据属性绘制直方图与盒图的代码与结果如下：

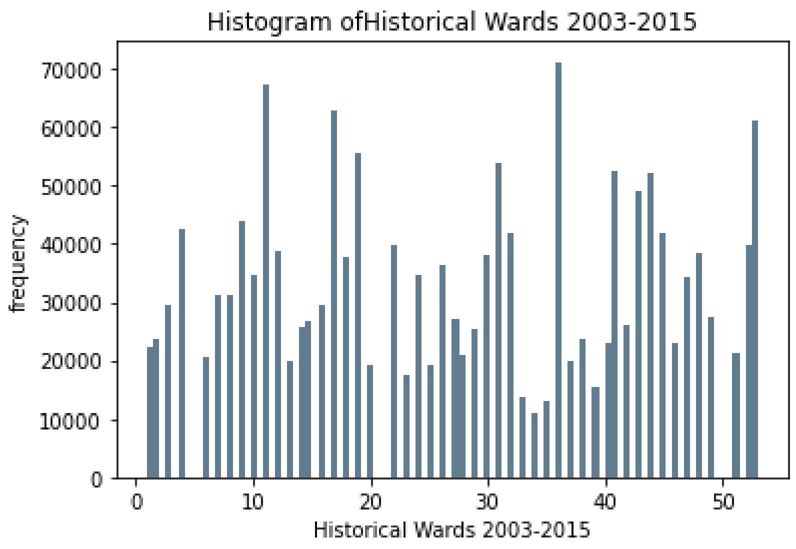
```
att = ['Wards']
show_hist(data, att,100)
show_boxplot(data,att)
```

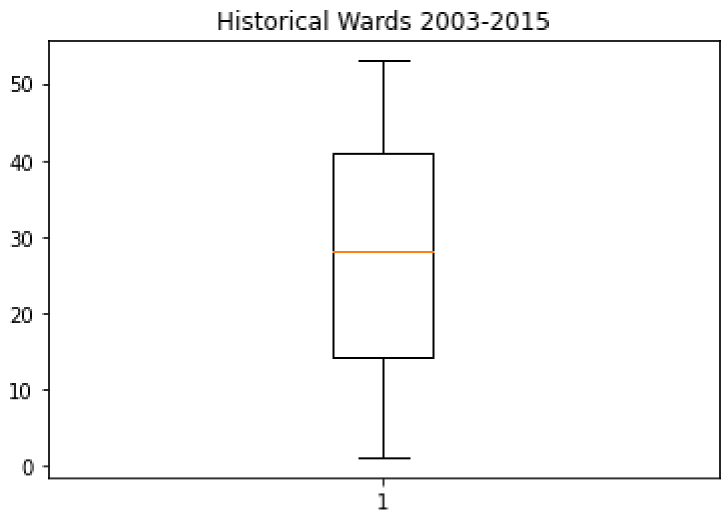




对Historical Wards 2003-2015数据属性绘制直方图与盒图的代码与结果如下：

```
att = ['Historical Wards 2003-2015']
show_hist(data, att,100)
show_boxplot(data,att)
```





通过直方图可以看出这些属性的分布不符合正态分布，但是通过盒图可以看出这些数据的分布较为均匀且没有离群点，

数据缺失处理

观察数据集中缺失数据，分析其缺失的原因，分别使用下列四种策略对缺失值进行处理

将缺失部分剔除

我们将所有缺失数据全部剔除并保存。对处理之后的数据五数概括的代码及结果为：

```
def lost_edition1():
    data_lost1 = data.copy()
    data_lost1 = data_lost1.dropna()
    return data_lost1

save_or_not = False
data_lost1 = lost_edition1()
print(data_lost1.isnull().sum())
if save_or_not:
    data_lost1.to_csv('data_delete.csv')
```

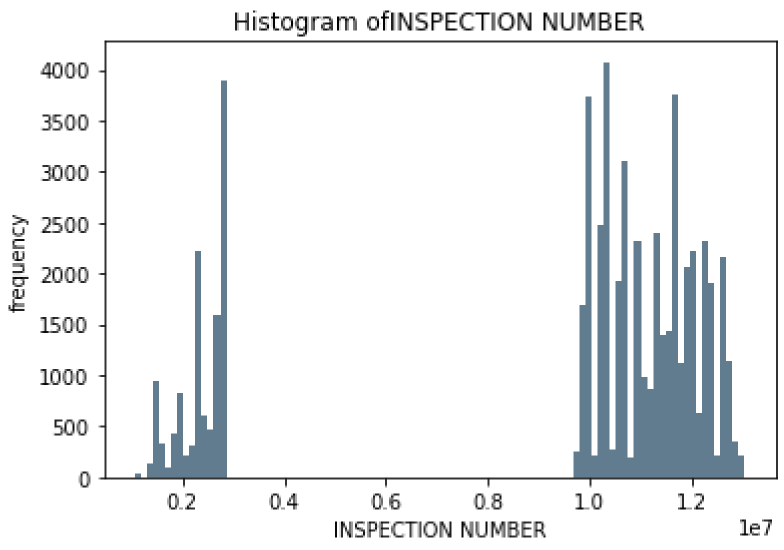
```python
for i in att_num:
    data_i_np = data_lost1.loc[:,i].values
    min_temp, Q1, median, Q3, max_temp = fiveNumber(data_i_np)
    print(i+' five number:')
    print(min_temp,Q1,median,Q3,max_temp)
```

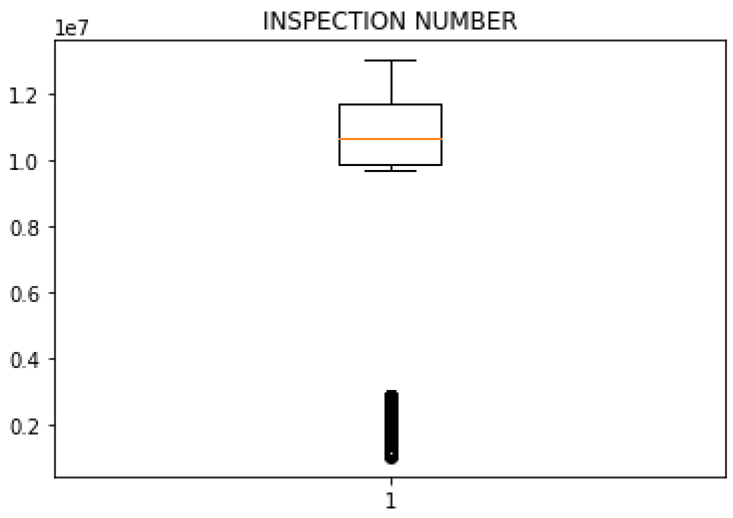
```
ID                                0
VIOLATION LAST MODIFIED DATE      0
VIOLATION DATE                    0
VIOLATION CODE                    0
VIOLATION STATUS                  0
VIOLATION STATUS DATE             0
VIOLATION DESCRIPTION             0
VIOLATION LOCATION                0
VIOLATION INSPECTOR COMMENTS      0
VIOLATION ORDINANCE               0
INSPECTOR ID                      0
INSPECTION NUMBER                 0
INSPECTION STATUS                 0
INSPECTION WAIVED                 0
INSPECTION CATEGORY               0
DEPARTMENT BUREAU                 0
ADDRESS                           0
STREET NUMBER                     0
STREET DIRECTION                  0
STREET NAME                       0
STREET TYPE                       0
PROPERTY GROUP                    0
SSA                               0
LATITUDE                          0
LONGITUDE                         0
LOCATION                          0
Community Areas                   0
Zip Codes                         0
Boundaries - ZIP Codes            0
Census Tracts                     0
Wards                             0
Historical Wards 2003-2015        0
dtype: int64
INSPECTION NUMBER five number:
1061448 9929462.0 10707030.0 11716234.0 13035353
PROPERTY GROUP five number:
1000 18144.0 26200.0 350611.0 674421
Community Areas five number:
2.0 25.0 40.0 59.0 76.0
Census Tracts five number:
1.0 188.0 360.0 516.0 801.0
Wards five number:
1.0 13.0 31.0 37.0 50.0
Historical Wards 2003-2015 five number:
1.0 13.0 31.0 42.0 53.0
```

原数据的INSPECTION NUMBER属性的五数概括为：[265575 2304416.0 10418746.0 11687280.0 13050915], 数据处理之后的五数概括为[1061448 9929462.0 10707030.0 11716234.0 13035353]。
原数据的PROPERTY GROUP属性的五数概括为：[1000 20560.0 154323.0 366984.0 677975], 数据处理之后的五数概括为[1000 18144.0 26200.0 350611.0 674421]。
原数据的Community Areas属性的五数概括为：[1.0 24.0 36.0 58.0 77.0], 数据处理之后的五数概括为[2.0 25.0 40.0 59.0 76.0]。
原数据的Census Tracts属性的五数概括为：[1.0 179.0 374.0 572.0 801.0], 数据处理之后的五数概括为[1.0 188.0 360.0 516.0 801.0]。
原数据的Wards属性的五数概括为：[1.0 12.0 25.0 37.0 50.0], 数据处理之后的五数概括为[1.0 13.0 31.0 37.0 50.0]。
原数据的PHistorical Wards 2003-2015属性的五数概括为：[1.0 14.0 28.0 41.0 53.0], 数据处理之后的五数概括为[1.0 13.0 31.0 42.0 53.0]。

对数据处理之后INSPECTION NUMBER的直方图与盒图的代码及结果为：

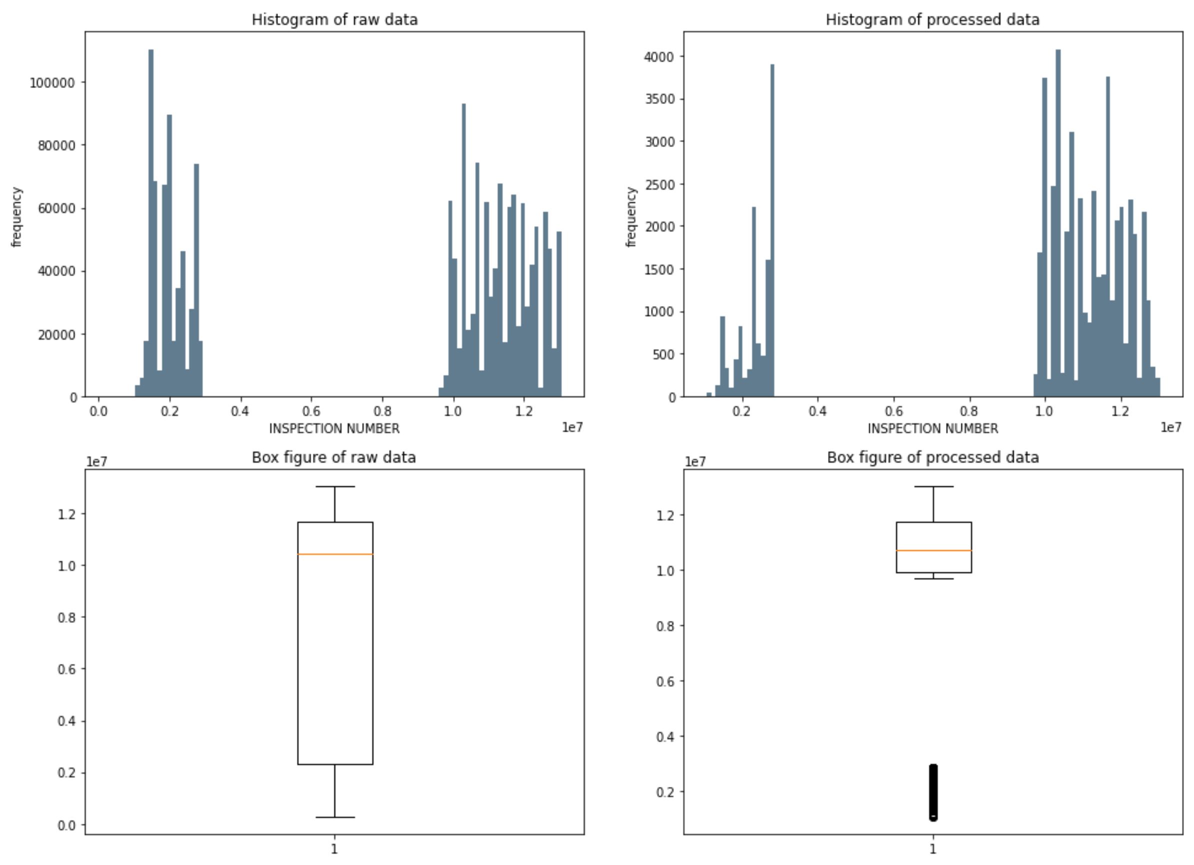
```
att = ['INSPECTION NUMBER']
show_hist(data_lost1, att, 100)
show_boxplot(data_lost1,att)
```





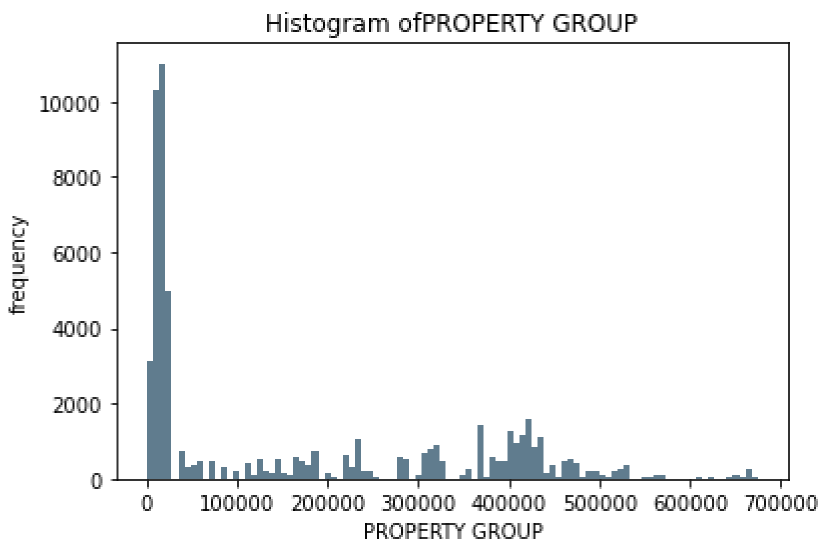
与原数据的对比结果如下：

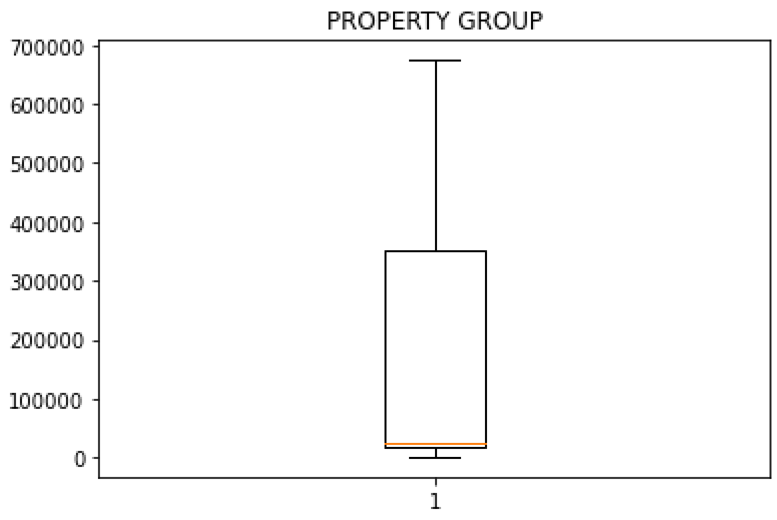
```python
def subplot_show(data1, att1, data2, att2, bin_num, title1, title2):

    data1_temp = data1.copy()

    data_i = data1_temp.dropna(subset=[att1])
    data1_att = data_i.loc[:,att1].values
    data2_att = data2.loc[:,att2].values
    plt.figure(figsize=(16, 12))
    plt.subplot(221)
    plt.hist(data1_att,bins = bin_num, color = '#607c8e', rwidth=5)
    plt.xlabel(att1)
    plt.ylabel('frequency')
    plt.title('Histogram of '+ title1 )

    plt.subplot(222)
    plt.hist(data2_att,bins = bin_num, color = '#607c8e', rwidth=5)
    plt.xlabel(att2)
    plt.ylabel('frequency')
    plt.title('Histogram of '+ title2 )

    plt.subplot(223)
    plt.boxplot(data1_att, notch=False, sym='o', vert=True)

    plt.title('Box figure of '+title1)

    plt.subplot(224)
    plt.boxplot(data2_att, notch=False, sym='o', vert=True)
    plt.title('Box figure of '+title2)

    plt.show()
att = 'INSPECTION NUMBER'
subplot_show(data, att, data_lost1, att, 100, 'raw data', 'processed data')
```



对数据处理之后PROPERTY GROUP的直方图与盒图的代码及结果为:

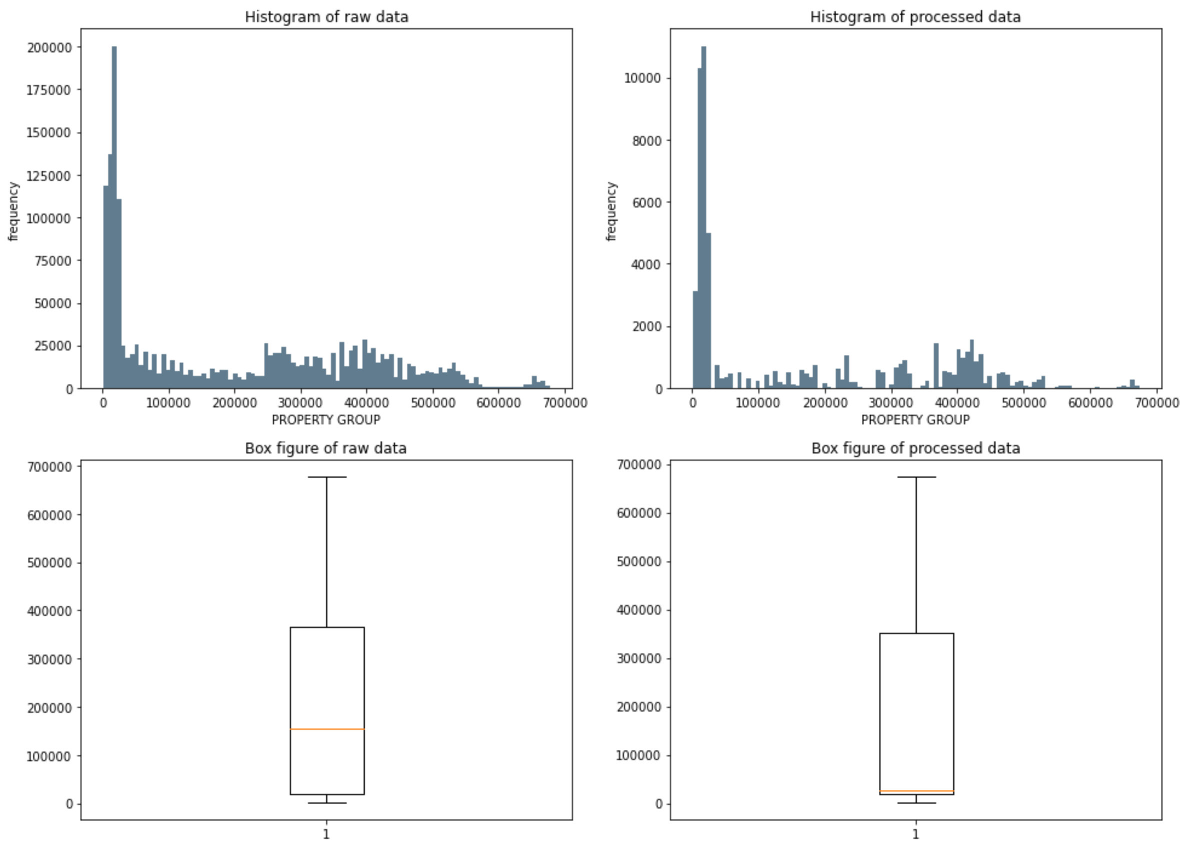
```
att = ['PROPERTY GROUP']
show_hist(data_lost1, att, 100)
show_boxplot(data_lost1,att)
```





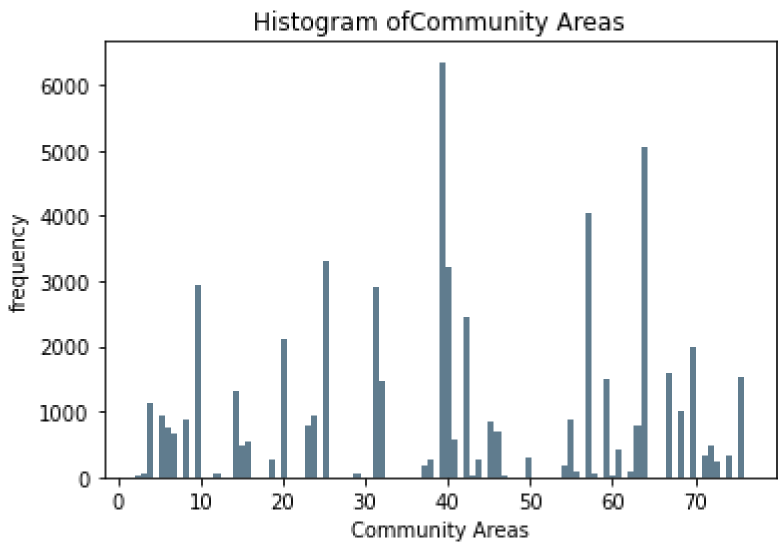
与原数据代码及结果为:

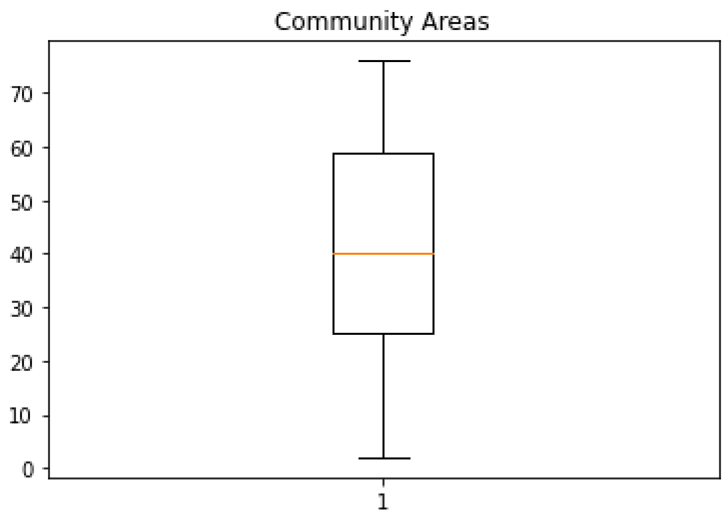
```
att = 'PROPERTY GROUP'
subplot_show(data, att, data_lost1, att, 100, 'raw data', 'processed data')
```



对数据处理之后Community Areas的直方图与盒图的代码及结果为:

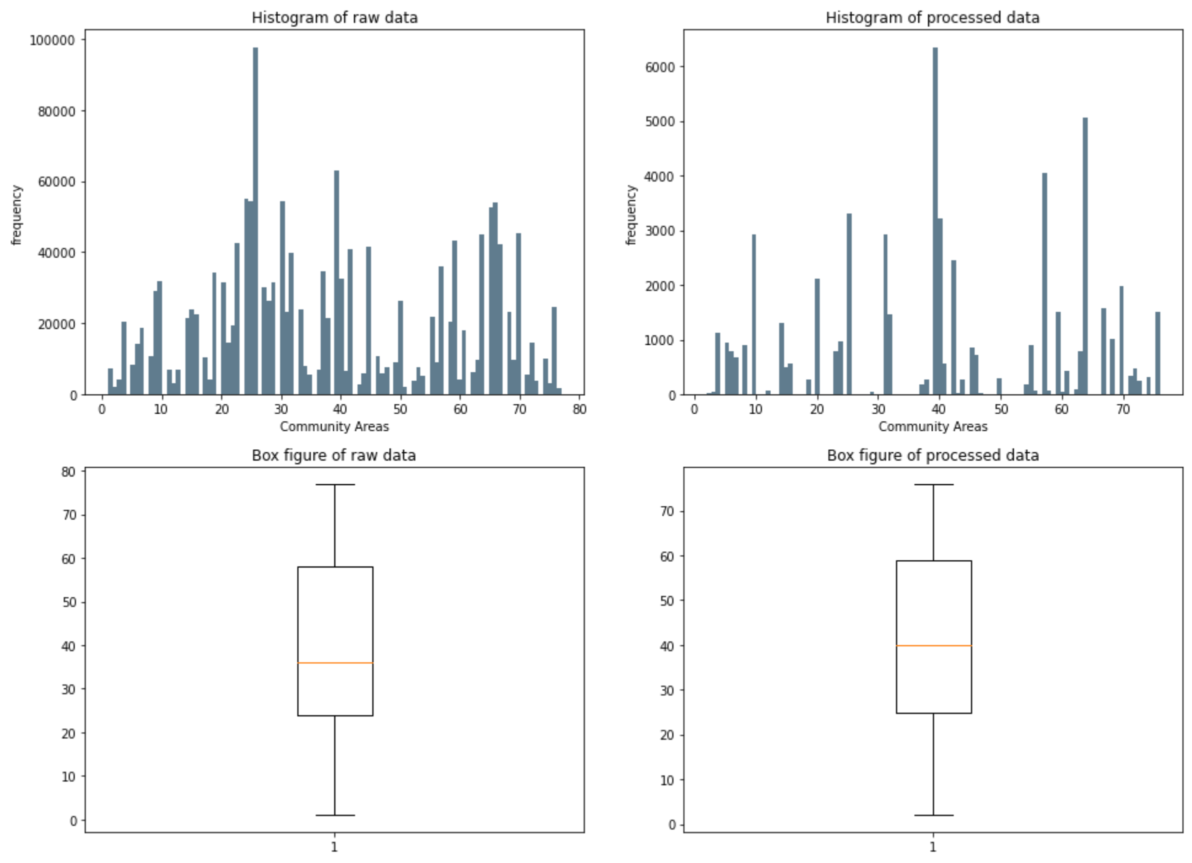
```
att = ['Community Areas']
show_hist(data_lost1, att, 100)
show_boxplot(data_lost1,att)
```





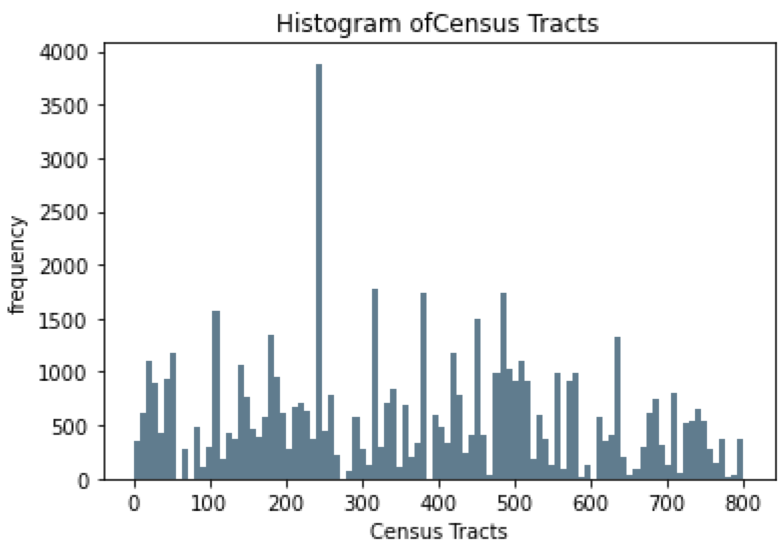
与原数据比较代码及结果为：

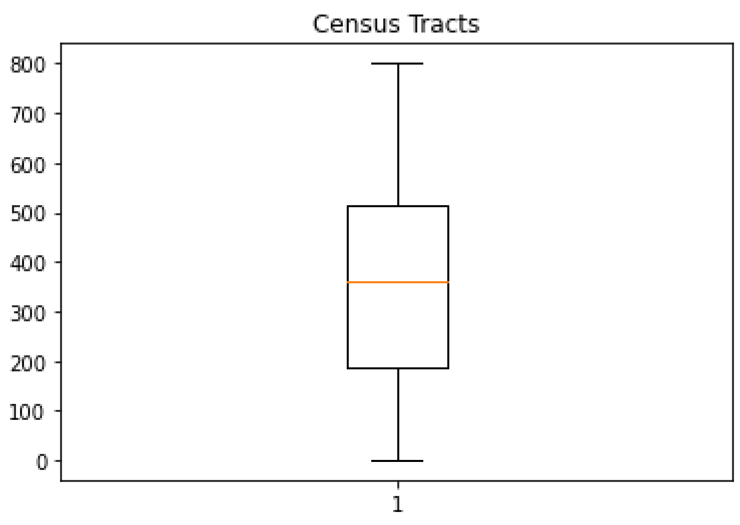
```
att = 'Community Areas'
subplot_show(data, att, data_lost1, att, 100, 'raw data', 'processed data')
```



对数据处理之后Census Tracts的直方图与盒图的代码及结果为：

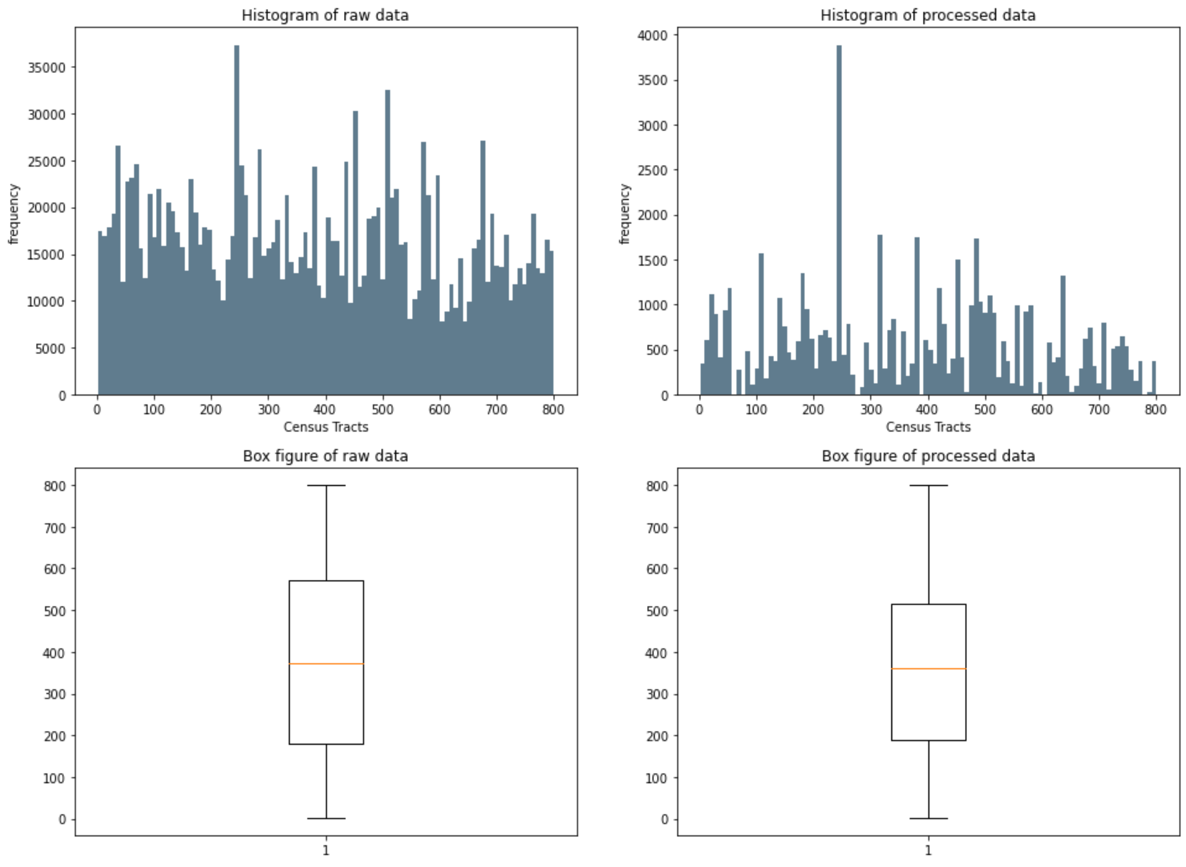
```
att = ['Census Tracts']
show_hist(data_lost1, att, 100)
show_boxplot(data_lost1,att)
```





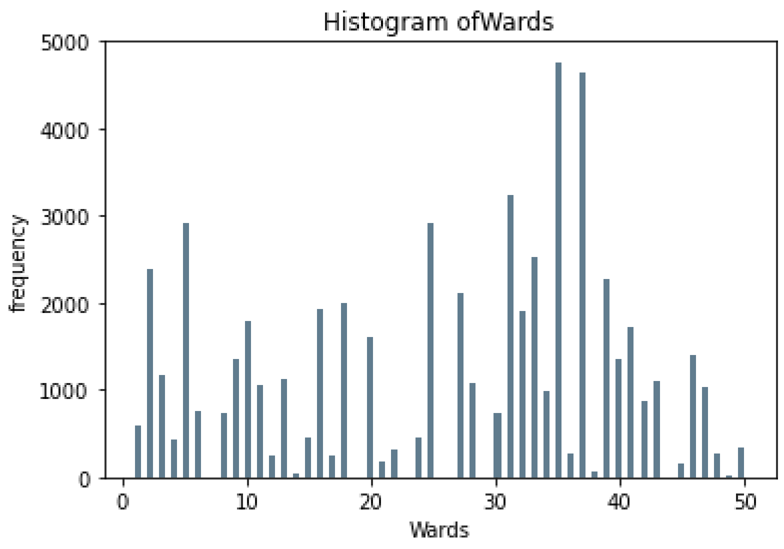
与原数据比较代码及结果为：

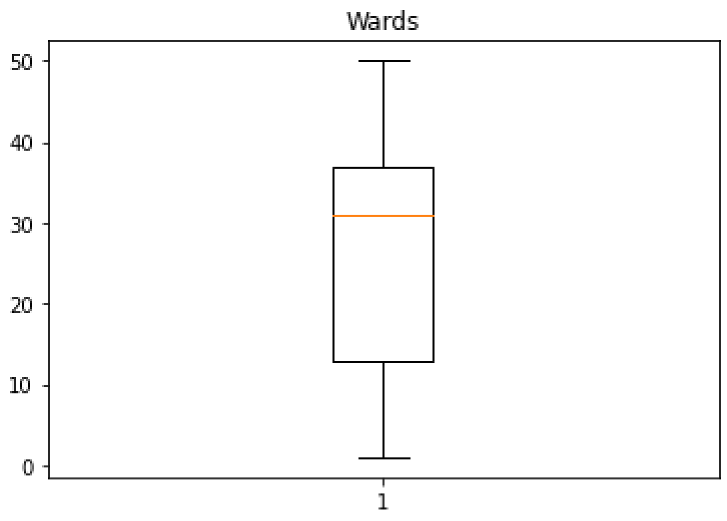
```python
att = 'Census Tracts'
subplot_show(data, att, data_lost1, att, 100, 'raw data', 'processed data')
```



对数据处理之后Wards的直方图与盒图的代码及结果为：

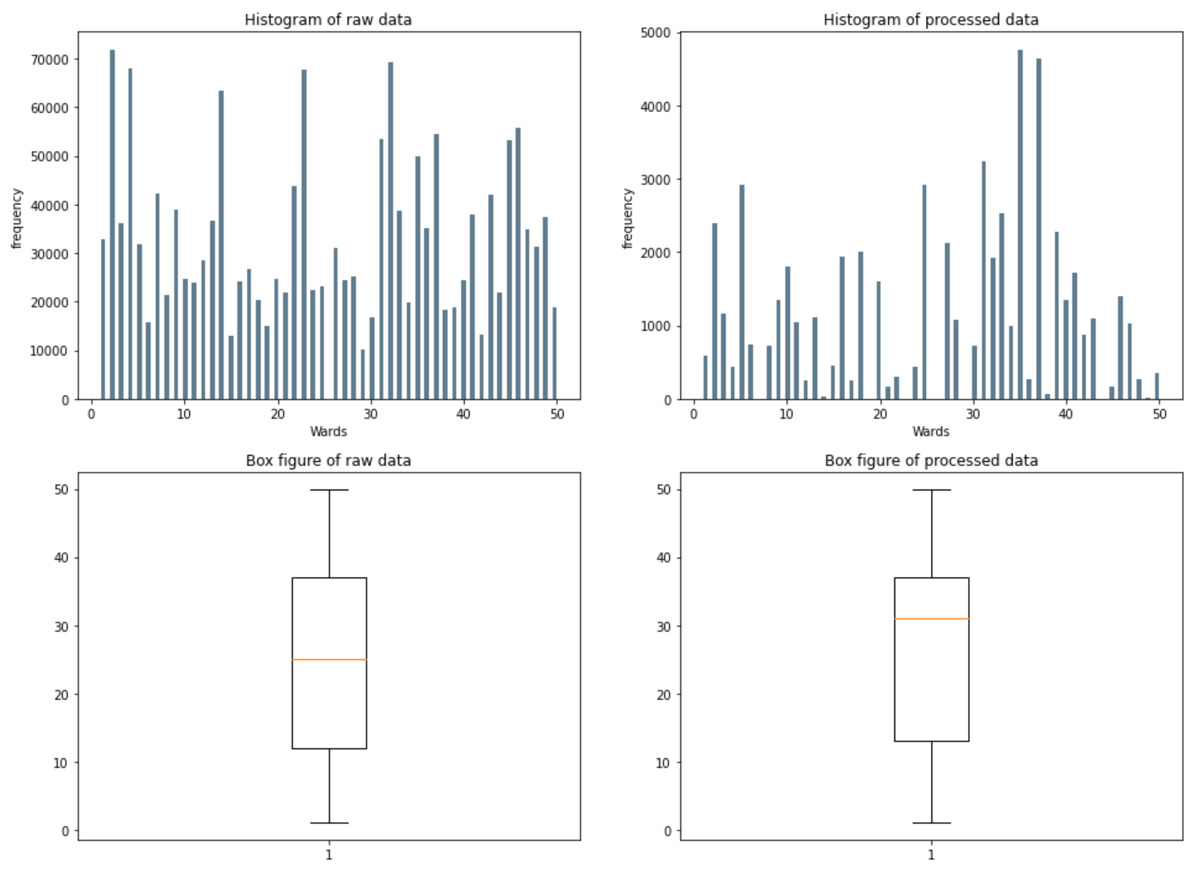
```python
att = ['Wards']
show_hist(data_lost1, att, 100)
show_boxplot(data_lost1,att)
```





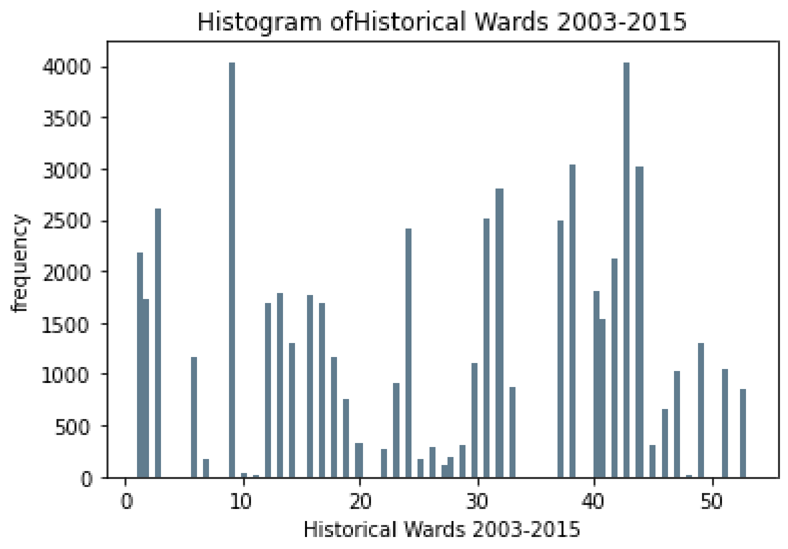
与原数据比较代码及结果为:

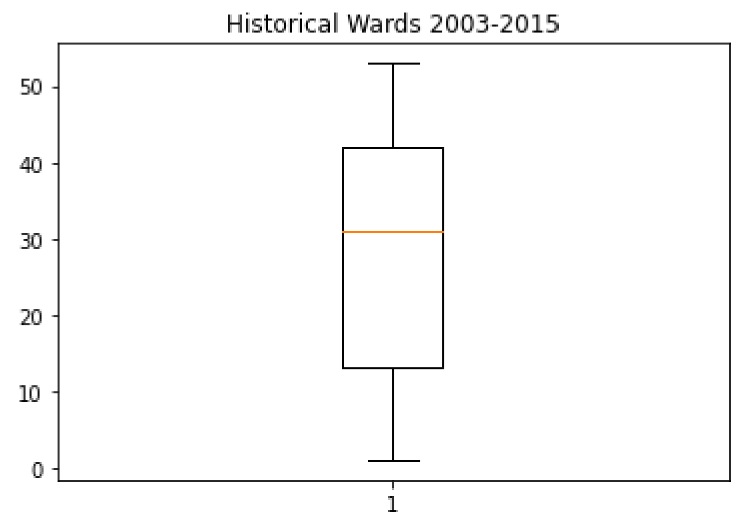
```
att = 'Wards'
subplot_show(data, att, data_lost1, att, 100, 'raw data', 'processed data')
```



对数据处理之后Historical Wards 2003-2015的直方图与盒图的代码及结果为：

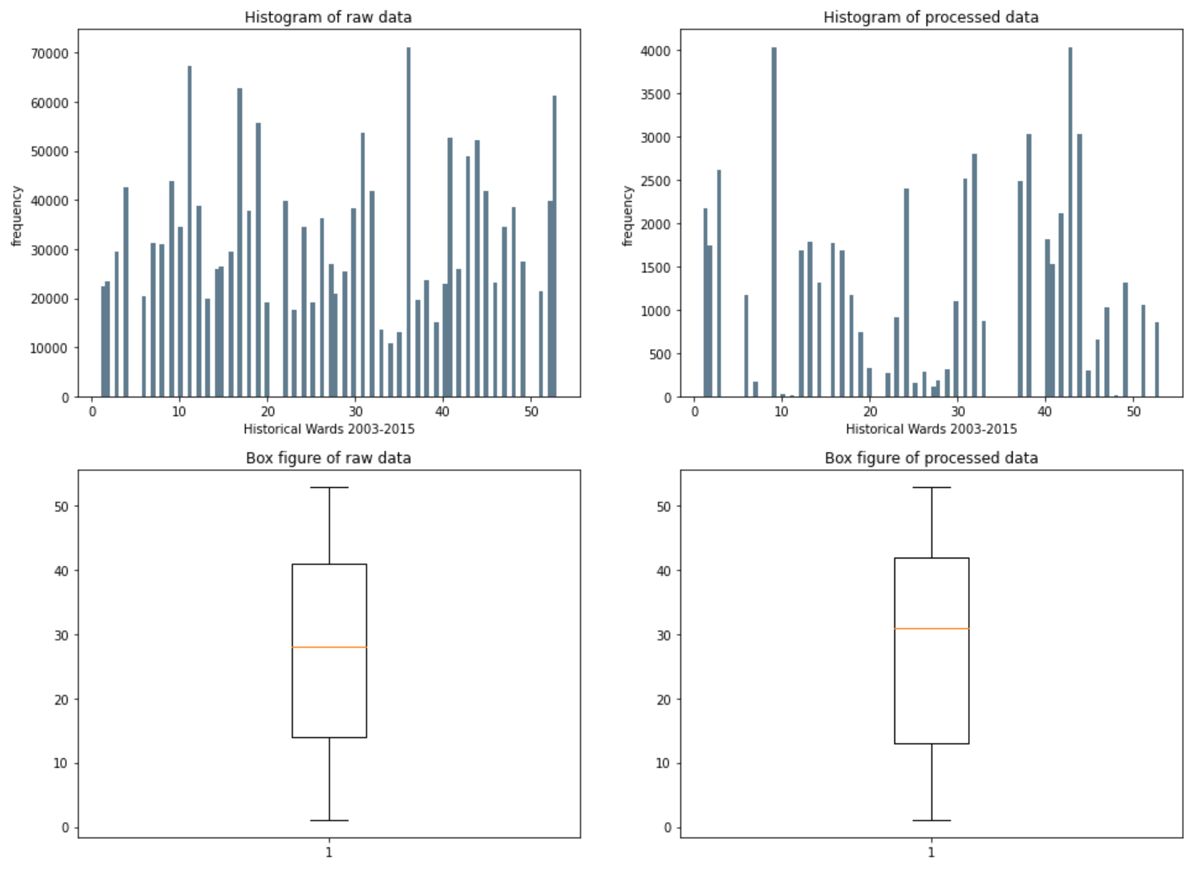
```
att = ['Historical Wards 2003-2015']
show_hist(data_lost1, att, 100)
show_boxplot(data_lost1,att)
```





与原数据比较代码及结果为：

```
att = 'Historical Wards 2003-2015'
subplot_show(data, att, data_lost1, att, 100, 'raw data', 'processed data')
```



由此可以看出，在去掉全部的缺失数据之后，INSPECTION NUMBER的数据的分布多出了许多离群点，但是其他的数据诸如Census Tracts的数据分布更趋向于正态分布，删除缺失数据的数据处理方式不能保证所有的数据分布均匀，因此可以看出这不是一种比较好的处理方式。

用最高频率值来填补缺失值

我们首先计算price属性最高频率的数值，将空缺值替换为该值并保存。对处理之后的数据五数概括的代码及结果为：

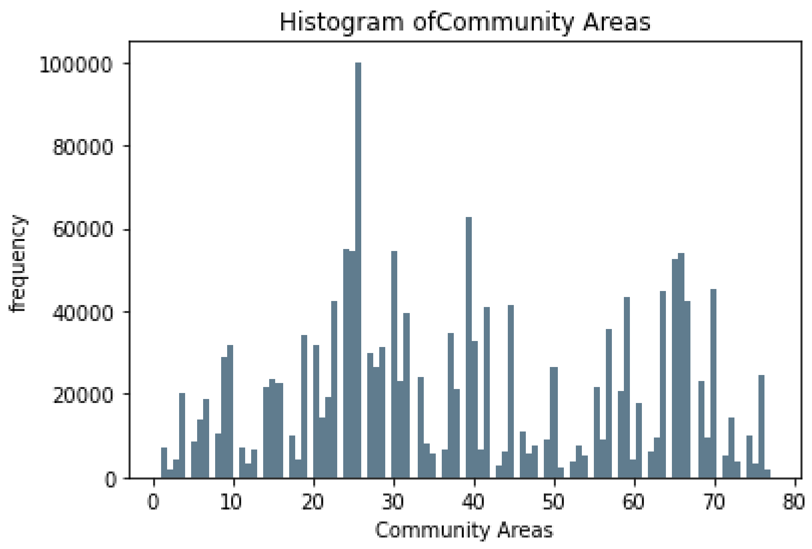
```
def lost_edition2(att_list):

    data_lost2 = data.copy()


    for i in att_list:
        i_temp_np = data_lost2[data_lost2[i] != np.NaN].loc[:,i].values
        i_temp_mode = mode(i_temp_np)[0].tolist()[0]
        data_lost2[i] = data_lost2[i].fillna(i_temp_mode)

    return data_lost2







save_or_not = False
data_lost2 = lost_edition2(att_num)
print(data_lost2.isnull().sum())
if save_or_not:
    data_lost2.to_csv('data_mode.csv')


for i in att_num:
    data_i_np = data_lost2.loc[:,i].values
    min_temp, Q1, median, Q3, max_temp = fiveNumber(data_i_np)
    print(i+' five number:')
    print(min_temp,Q1,median,Q3,max_temp)
```

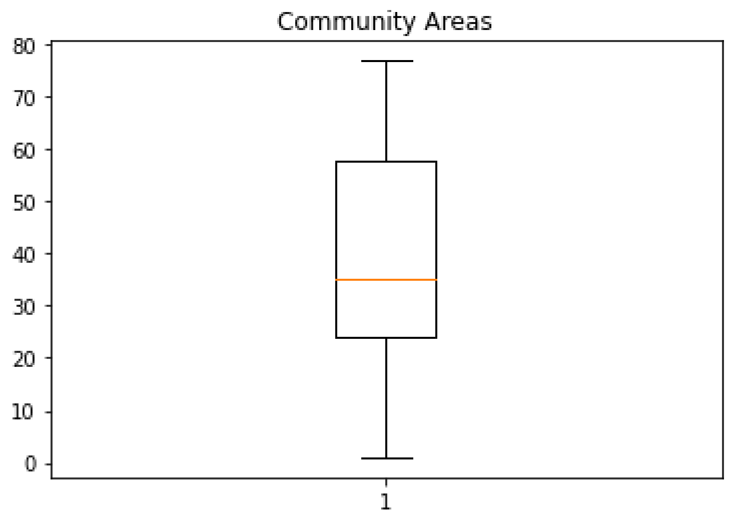
```
ID                                  0
VIOLATION LAST MODIFIED DATE        0
VIOLATION DATE                      0
VIOLATION CODE                      0
VIOLATION STATUS                    0
VIOLATION STATUS DATE         1036199
VIOLATION DESCRIPTION           10768
VIOLATION LOCATION             897282
VIOLATION INSPECTOR COMMENTS   175463
VIOLATION ORDINANCE             47581
INSPECTOR ID                        0
INSPECTION NUMBER                   0
INSPECTION STATUS                  16
INSPECTION WAIVED                   0
INSPECTION CATEGORY                 0
DEPARTMENT BUREAU                   0
ADDRESS                             0
STREET NUMBER                       0
STREET DIRECTION                    0
STREET NAME                         0
STREET TYPE                      13541
PROPERTY GROUP                       0
SSA                            1356267
LATITUDE                          1510
LONGITUDE                         1510
LOCATION                          1510
Community Areas                      0
Zip Codes                         1510
Boundaries - ZIP Codes            2279
Census Tracts                        0
Wards                                0
Historical Wards 2003-2015           0
dtype: int64
INSPECTION NUMBER five number:
265575 2304416.0 10418746.0 11687280.0 13050915
PROPERTY GROUP five number:
1000 20560.0 154323.0 366984.0 677975
Community Areas five number:
1.0 24.0 35.0 58.0 77.0
Census Tracts five number:
1.0 178.0 374.0 572.0 801.0
Wards five number:
1.0 12.0 25.0 37.0 50.0
Historical Wards 2003-2015 five number:
1.0 14.0 28.0 41.0 53.0
```

原数据的INSPECTION NUMBER属性的五数概括为：[265575 2304416.0 10418746.0 11687280.0 13050915], 数据处理之后的五数概括为[265575 2304416.0 10418746.0 11687280.0 13050915]。
原数据的PROPERTY GROUP属性的五数概括为：[1000 20560.0 154323.0 366984.0 677975], 数据处理之后的五数概括为[1000 20560.0 154323.0 366984.0 677975]。
原数据的Community Areas属性的五数概括为：[1.0 24.0 36.0 58.0 77.0], 数据处理之后的五数概括为[1.0 24.0 36.0 58.0 77.0]。
原数据的Census Tracts属性的五数概括为：[1.0 179.0 374.0 572.0 801.0], 数据处理之后的五数概括为[1.0 178.0 374.0 572.0 801.0]。
原数据的Wards属性的五数概括为：[1.0 12.0 25.0 37.0 50.0], 数据处理之后的五数概括为[1.0 12.0 25.0 37.0 50.0]。
原数据的PHistorical Wards 2003-2015属性的五数概括为：[1.0 14.0 28.0 41.0 53.0], 数据处理之后的五数概括为[1.0 14.0 28.0 41.0 53.0]。

对数据处理之后Community Areas的直方图与盒图的代码及结果为：

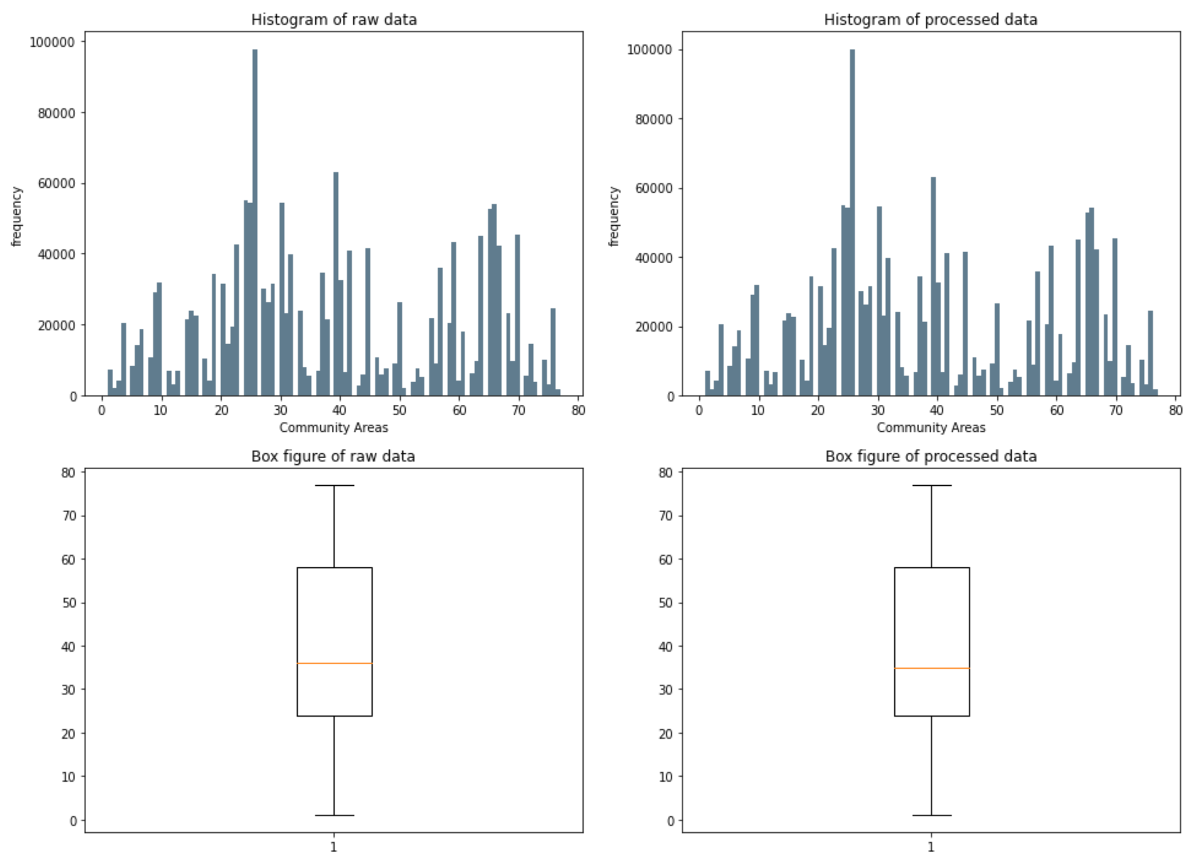
```
att = ['Community Areas']
show_hist(data_lost2, att, 100)
show_boxplot(data_lost2,att)
```





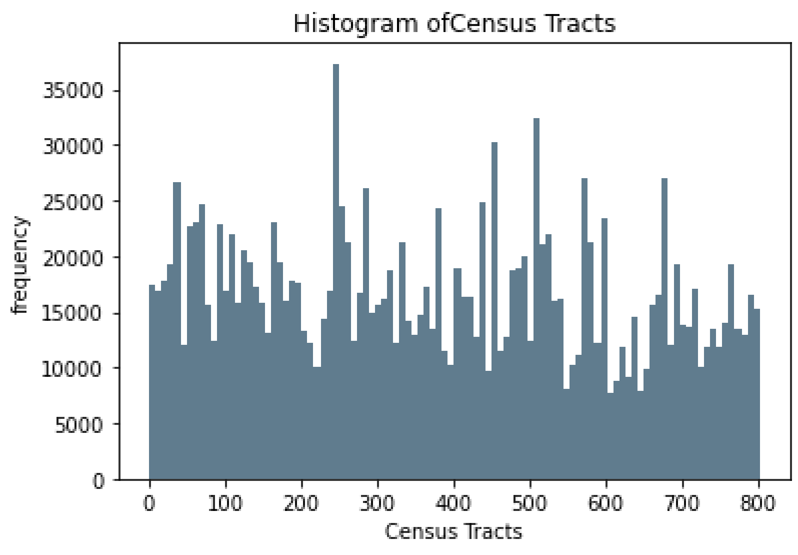
与原数据比较代码及结果为：

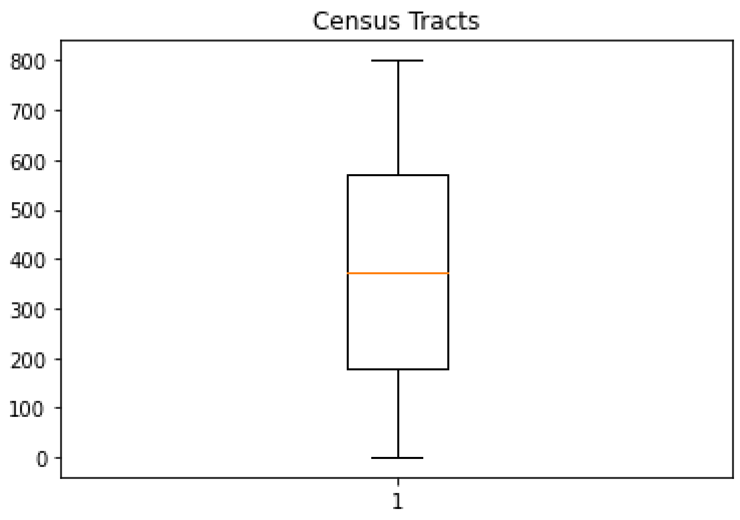
```
att = 'Community Areas'
subplot_show(data, att, data_lost2, att, 100, 'raw data', 'processed data')
```



对数据处理之后Census Tracts的直方图与盒图的代码及结果为:

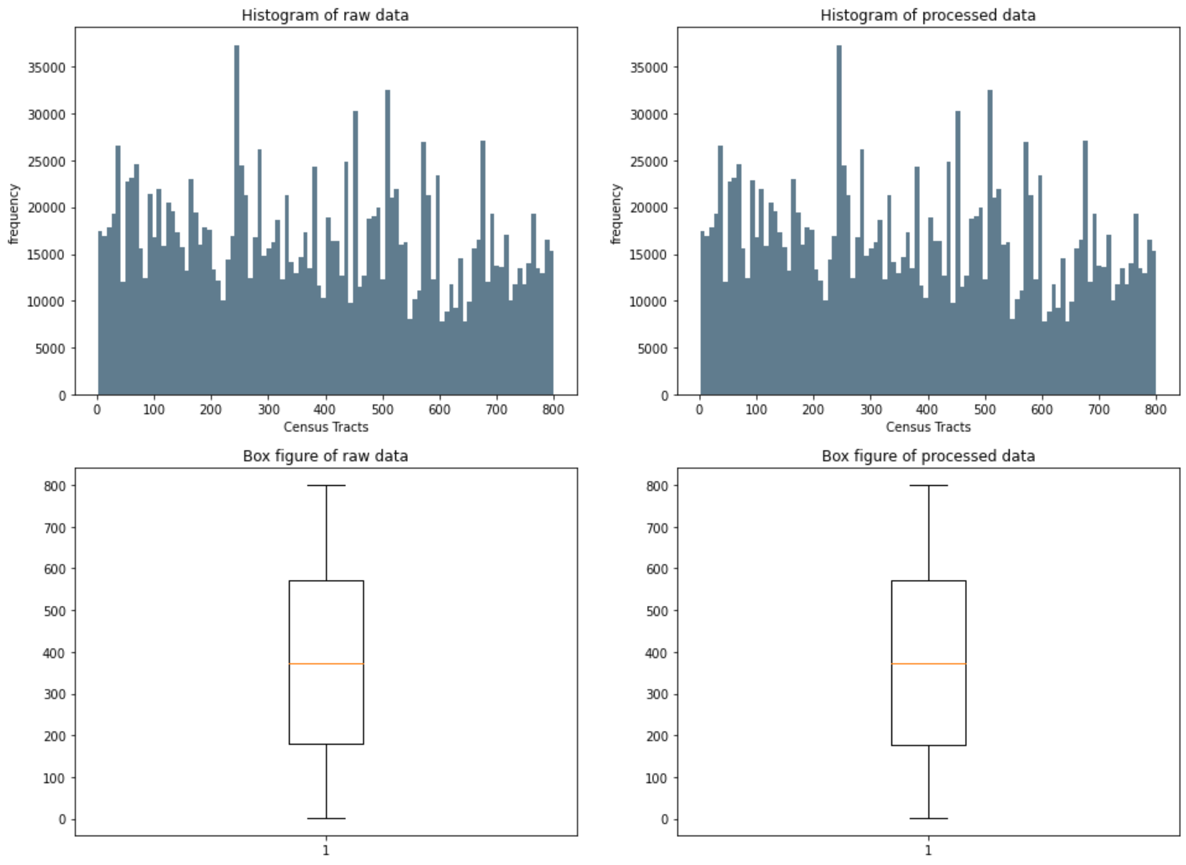
```
att = ['Census Tracts']
show_hist(data_lost2, att, 100)
show_boxplot(data_lost2,att)
```





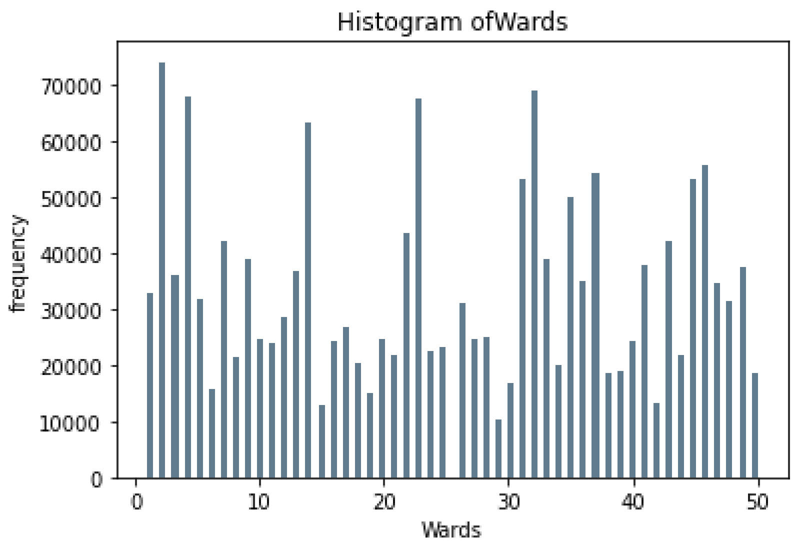
与原数据比较代码及结果为：

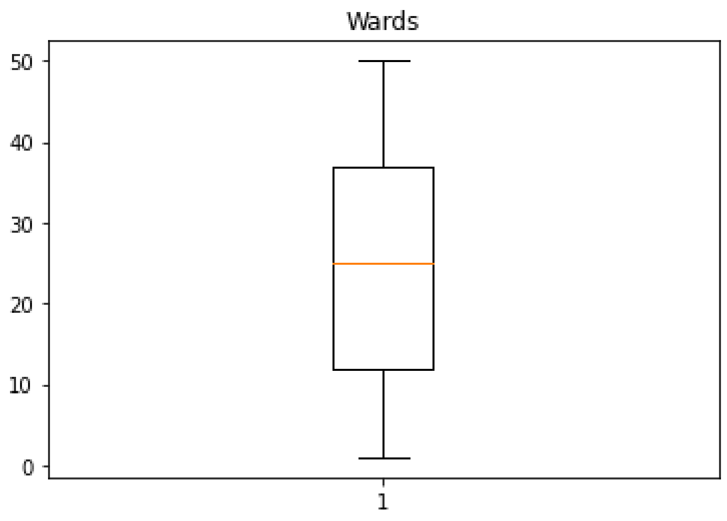
```
att = 'Census Tracts'
subplot_show(data, att, data_lost2, att, 100, 'raw data', 'processed data')
```



对数据处理之后Wards的直方图与盒图的代码及结果为：

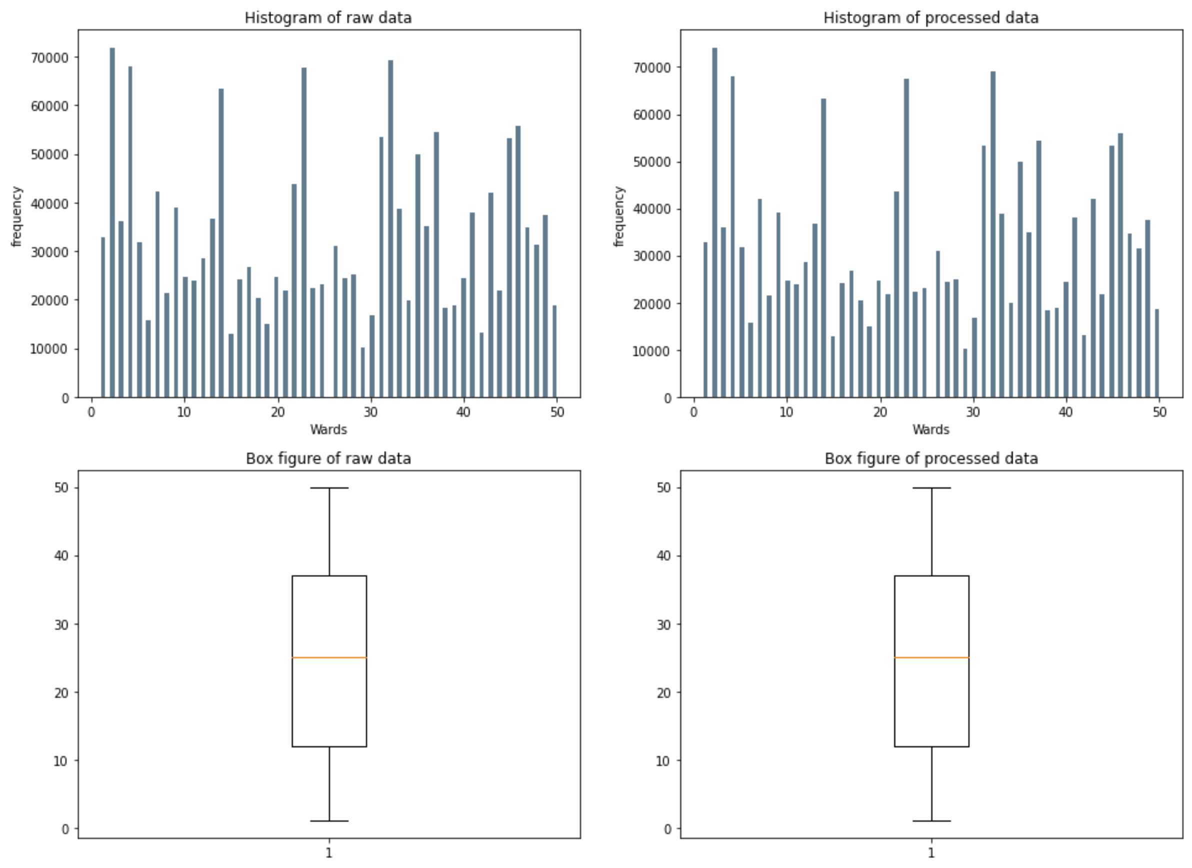
```
att = ['Wards']
show_hist(data_lost2, att, 100)
show_boxplot(data_lost2,att)
```





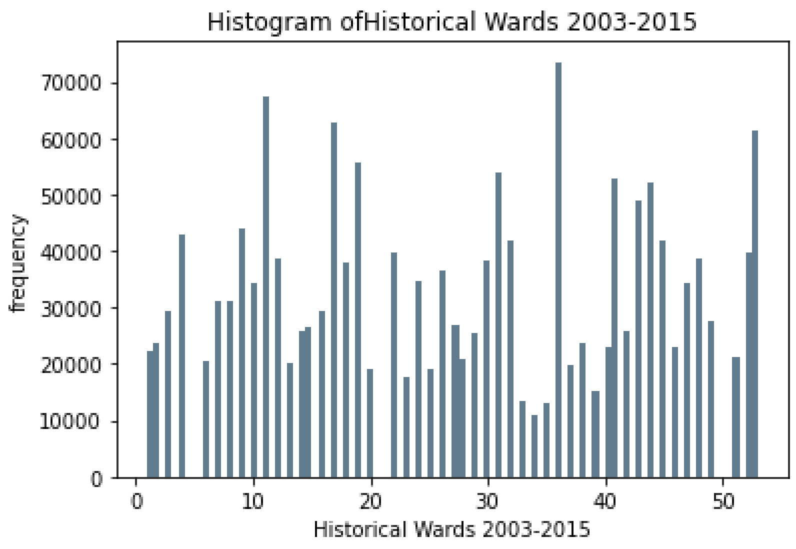
与原数据比较代码及结果为:

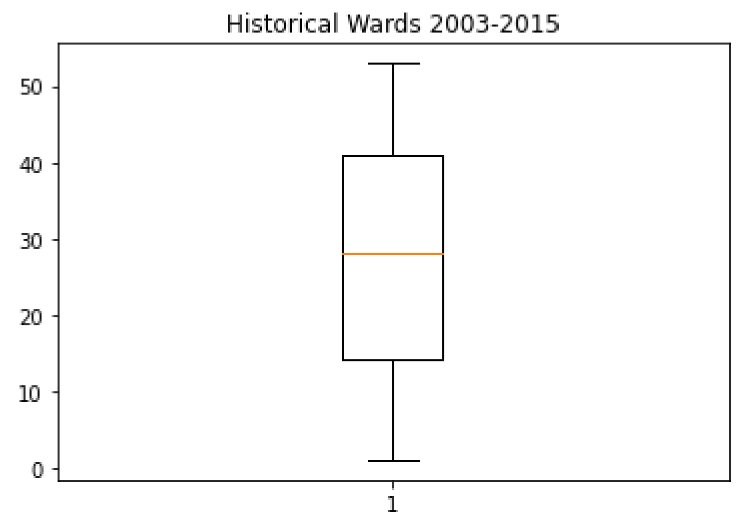
```
att = 'Wards'
subplot_show(data, att, data_lost2, att, 100, 'raw data', 'processed data')
```



对数据处理之后Historical Wards 2003-2015的直方图与盒图的代码及结果为：

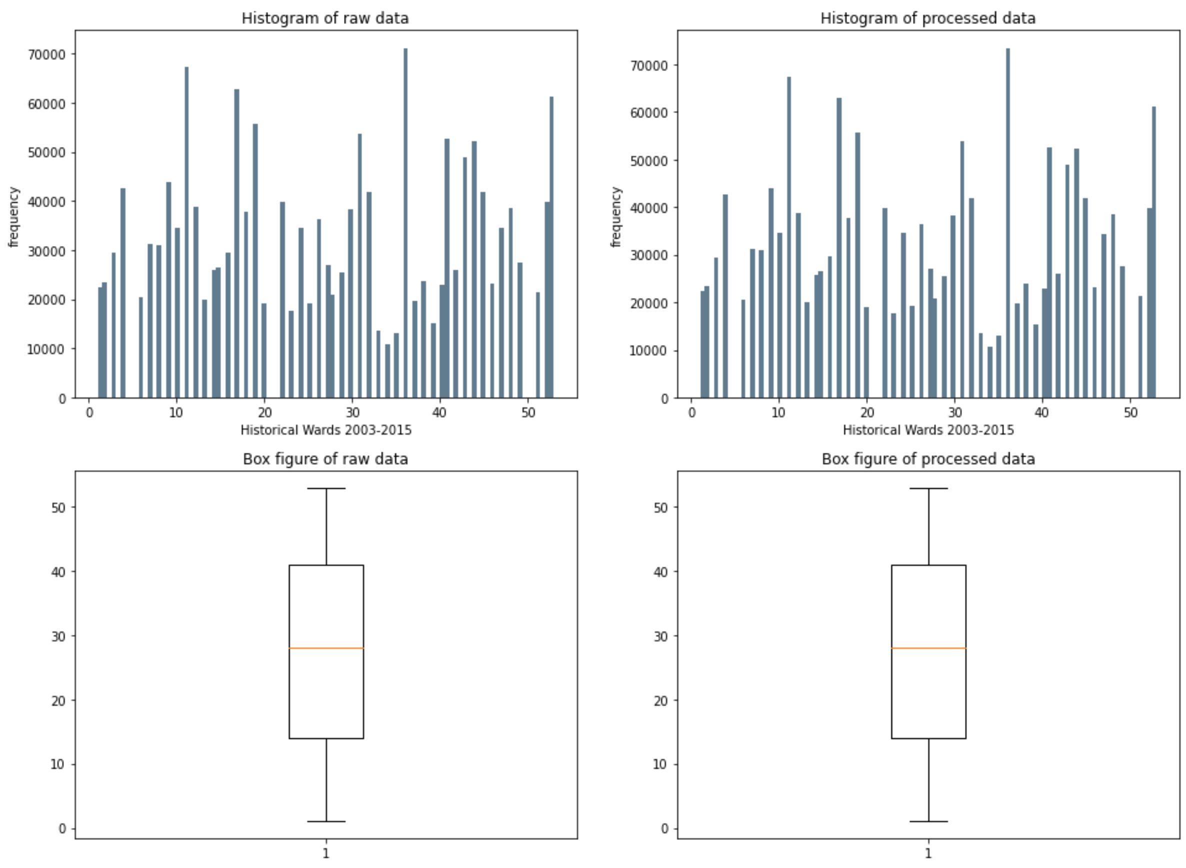
```
att = ['Historical Wards 2003-2015']
show_hist(data_lost2, att, 100)
show_boxplot(data_lost2,att)
```





与原数据比较代码及结果为：

```
att = 'Historical Wards 2003-2015'
subplot_show(data, att, data_lost2, att, 100, 'raw data', 'processed data')
```



通过属性的相关关系来填补缺失值

我们使用随机森林的方法对空缺的数据进行填充。在这里，我们采用了两种方式，分别为，利用完整的数据属性'INSPECTION NUMBER', 'PROPERTY GROUP'分别对空缺的数据属性进行填充，第二种是利用完整的数据属性'INSPECTION NUMBER', 'PROPERTY GROUP'依次对空缺的数据属性数据进行填充，并将填充的数据与'INSPECTION NUMBER', 'PROPERTY GROUP'一起作为参考数据对不完整的属性数据填充。由于数据量较大，随机森林方法需要的计算资源与存储资源较大，所以我们采用在服务器实现、保存数据，对数据分析的方式。按照第一种方式处理之后的数据五数概括的代码及结果为：

```python
def set_missing_prices(data_temp, att_list, target):

    att_list = att_list.copy()
    att_list.insert(0,target)
    print('----------------------')
    print(att_list)

    data_lost3 = data_temp.copy()

    # 把已有的数值型特征取出来丢进Random Forest Regressor中

    # 首先根据INSPECTION NUMBER', 'PROPERTY GROUP'填充 'Community Areas'的数据
    Community_df = data_lost3[att_list]
    goal = att_list[0]

    # 乘客分成已知年龄和未知年龄两部分
    temp = Community_df[Community_df[goal].notnull()]
    print(temp.isnull().sum())

    known_Community = Community_df[Community_df[goal].notnull()].values
    unknown_Community = Community_df[Community_df[goal].isnull()].values

    # y即目标值
    y = known_Community[:, 0]

    # X即特征属性值
    X = known_Community[:, 1:]

    # fit到RandomForestRegressor之中
    rfr = sklearn.ensemble.RandomForestRegressor(random_state=0,
n_estimators=2000, n_jobs=-1)
    rfr.fit(X, y)

    # 用得到的模型进行未知年龄结果预测
    predictedprice = rfr.predict(unknown_Community[:, 1:])
#     print predictedAges
    # 用得到的预测结果填补原缺失数据
    data_lost3.loc[ (data_lost3[goal].isnull()), goal ] = predictedprice


    return data_lost3


def lost_edition3(complete_list, uncomplete_list ):
    data_lost3 = data.copy()

    for i in uncomplete_list:


        print(complete_list)
        data_lost3 = set_missing_prices(data_lost3, complete_list, i)
        print(data_lost3.isnull().sum())

    return data_lost3
```

```
save_or_not = False

if save_or_not:
    data_lost3 = lost_edition3(att_num)
    print(data_lost2.isnull().sum())
    data_lost2.to_csv('data_mode.csv')

data_lost3_name = r'C:\Users\Xiaomeng Fan\Desktop\研究生的各种作业\数据挖掘\作业四
\building\data_forest_2.csv'
file_lost3 = pd.read_csv(data_lost3_name)
data_lost3 = pd.DataFrame(file_lost3)
print(data_lost3.isnull().sum())
for i in att_num:
    data_i_np = data_lost3.loc[:,i].values
    min_temp, Q1, median, Q3, max_temp = fiveNumber(data_i_np)
    print(i+' five number:')
    print(min_temp,Q1,median,Q3,max_temp)
```

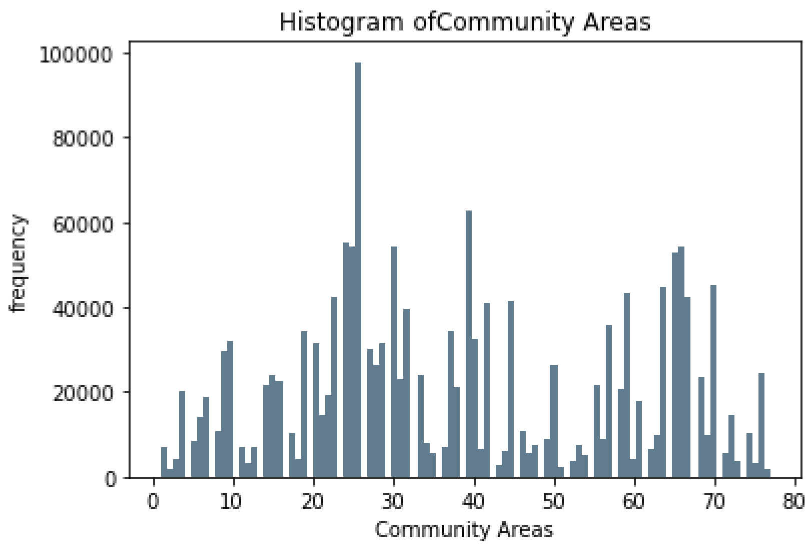
```
Unnamed: 0                            0
ID                                    0
VIOLATION LAST MODIFIED DATE          0
VIOLATION DATE                        0
VIOLATION CODE                        0
VIOLATION STATUS                      0
VIOLATION STATUS DATE           1036199
VIOLATION DESCRIPTION             10768
VIOLATION LOCATION               897282
VIOLATION INSPECTOR COMMENTS     175463
VIOLATION ORDINANCE               47581
INSPECTOR ID                          0
INSPECTION NUMBER                     0
INSPECTION STATUS                    16
INSPECTION WAIVED                     0
INSPECTION CATEGORY                   0
DEPARTMENT BUREAU                     0
ADDRESS                               0
STREET NUMBER                         0
STREET DIRECTION                      0
STREET NAME                           0
STREET TYPE                       13541
PROPERTY GROUP                        0
SSA                             1356267
LATITUDE                           1510
LONGITUDE                          1510
LOCATION                           1510
Community Areas                       0
Zip Codes                          1510
Boundaries - ZIP Codes             2279
Census Tracts                         0
Wards                                 0
Historical Wards 2003-2015            0
dtype: int64
INSPECTION NUMBER five number:
265575 2304416.0 10418746.0 11687280.0 13050915
PROPERTY GROUP five number:
```

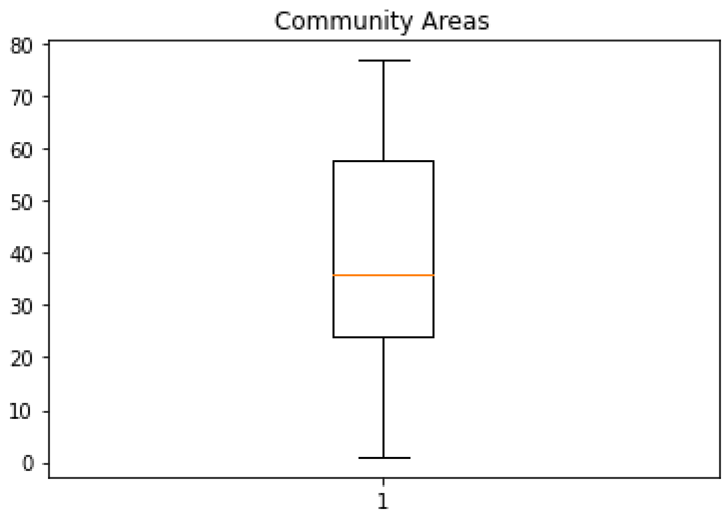
```
1000 20560.0 154323.0 366984.0 677975
Community Areas five number:
1.0 24.0 36.0 58.0 77.0
Census Tracts five number:
1.0 179.0 374.0 572.0 801.0
Wards five number:
1.0 12.0 25.0 37.0 50.0
Historical Wards 2003-2015 five number:
1.0 14.0 28.0 41.0 53.0
```

原数据的INSPECTION NUMBER属性的五数概括为：[265575 2304416.0 10418746.0 11687280.0 13050915], 数据处理之后的五数概括为[265575 2304416.0 10418746.0 11687280.0 13050915]。
原数据的PROPERTY GROUP属性的五数概括为：[1000 20560.0 154323.0 366984.0 677975], 数据处理之后的五数概括为[1000 20560.0 154323.0 366984.0 677975]。
原数据的Community Areas属性的五数概括为：[1.0 24.0 36.0 58.0 77.0], 数据处理之后的五数概括为[1.0 24.0 36.0 58.0 77.0]。
原数据的Census Tracts属性的五数概括为：[1.0 179.0 374.0 572.0 801.0], 数据处理之后的五数概括为[1.0 179.0 374.0 572.0 801.0]。
原数据的Wards属性的五数概括为：[1.0 12.0 25.0 37.0 50.0], 数据处理之后的五数概括为[1.0 12.0 25.0 37.0 50.0]。
原数据的PHistorical Wards 2003-2015属性的五数概括为：[1.0 14.0 28.0 41.0 53.0], 数据处理之后的五数概括为[1.0 14.0 28.0 41.0 53.0]。

对数据处理之后Community Areas的直方图与盒图的代码及结果为：

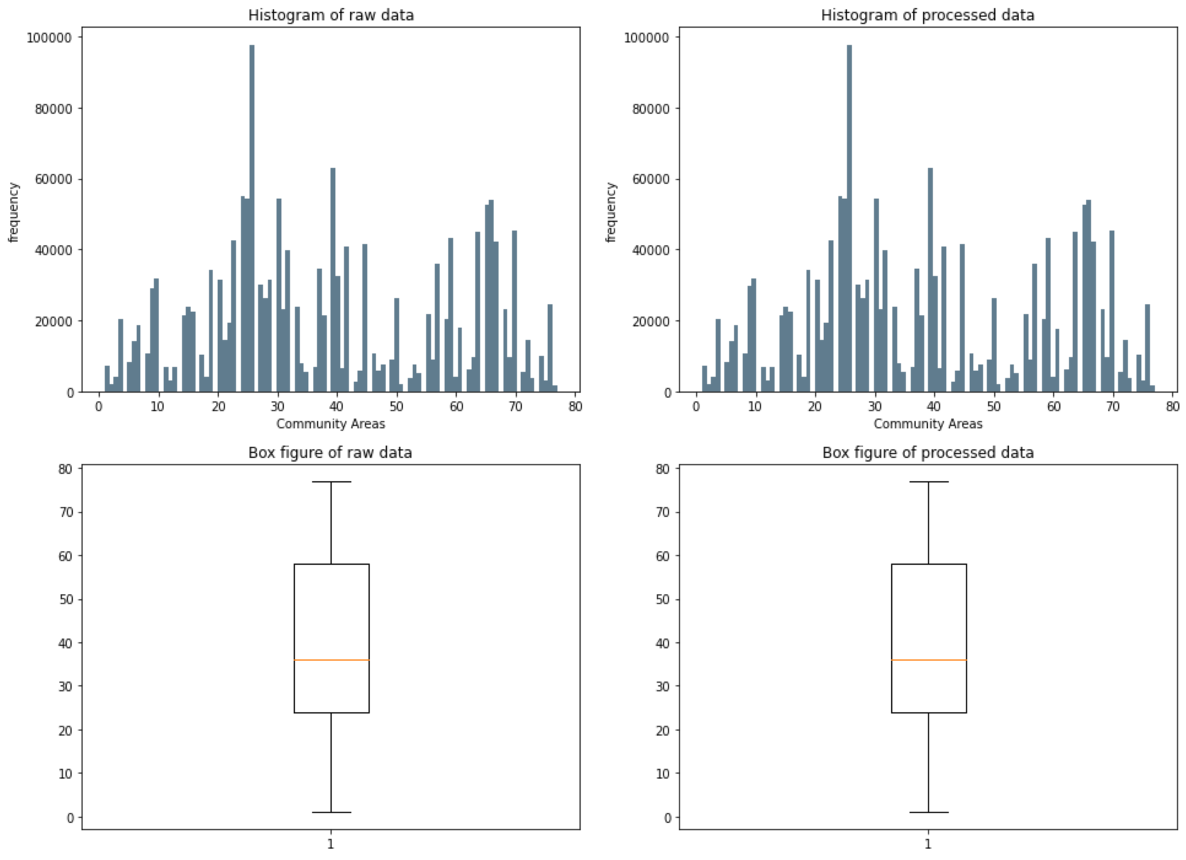
```
att = ['Community Areas']
show_hist(data_lost3, att, 100)
show_boxplot(data_lost3,att)
```





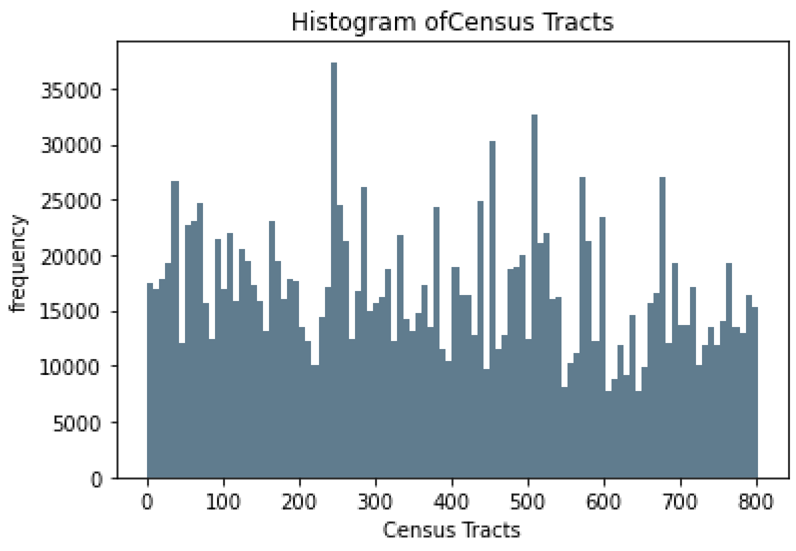
与原数据比较代码及结果为：

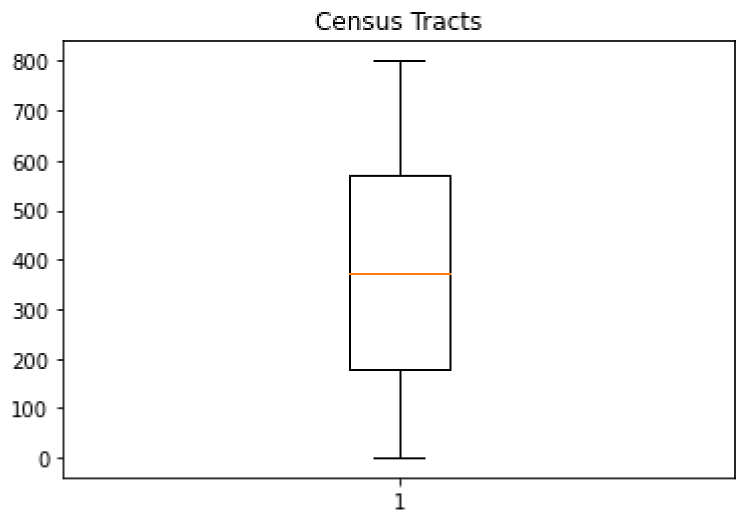
```
att = 'Community Areas'
subplot_show(data, att, data_lost3, att, 100, 'raw data', 'processed data')
```



对数据处理之后Census Tracts的直方图与盒图的代码及结果为：

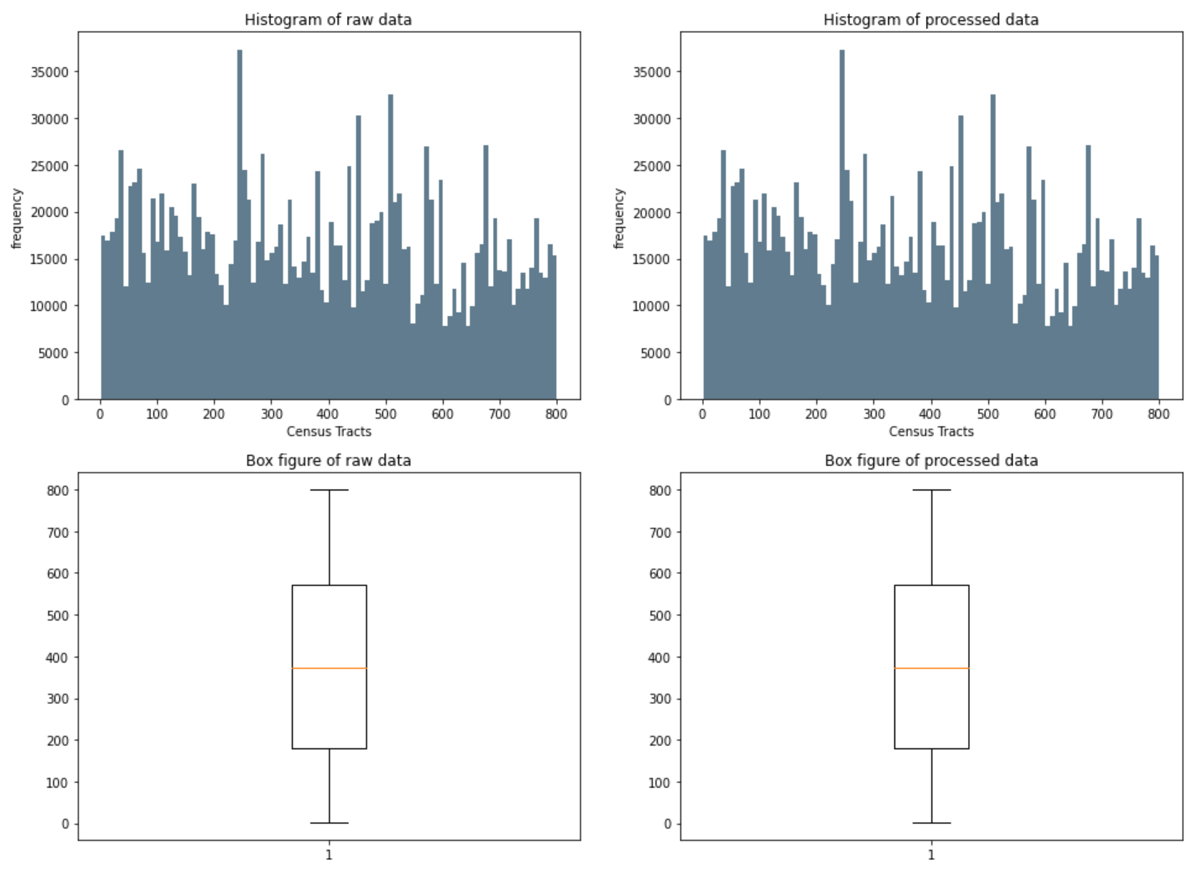
```
att = ['Census Tracts']
show_hist(data_lost3, att, 100)
show_boxplot(data_lost3,att)
```





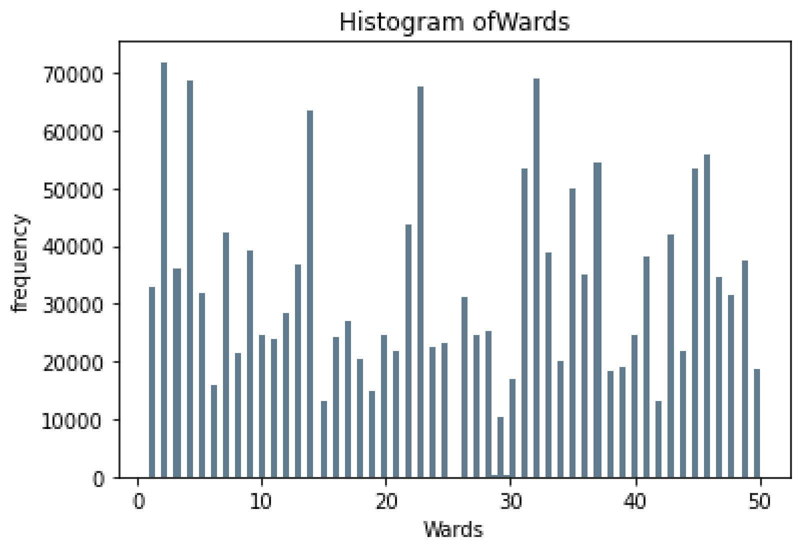
与原数据比较代码及结果为:

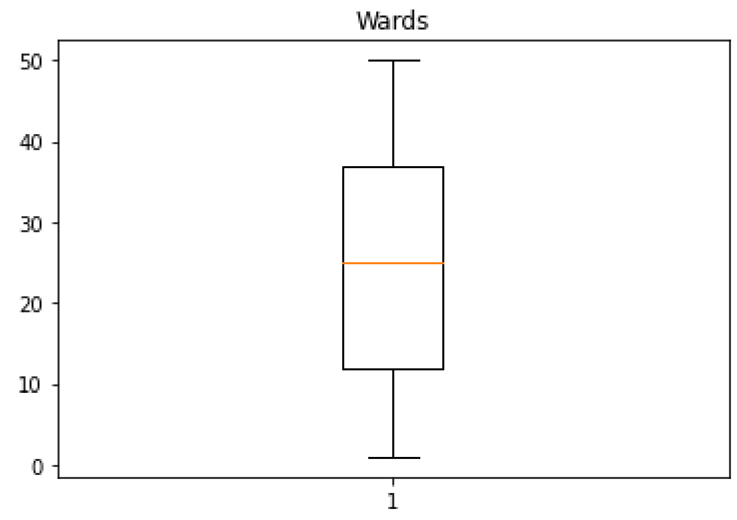
```
att = 'Census Tracts'
subplot_show(data, att, data_lost3, att, 100, 'raw data', 'processed data')
```



对数据处理之后Wards的直方图与盒图的代码及结果为:

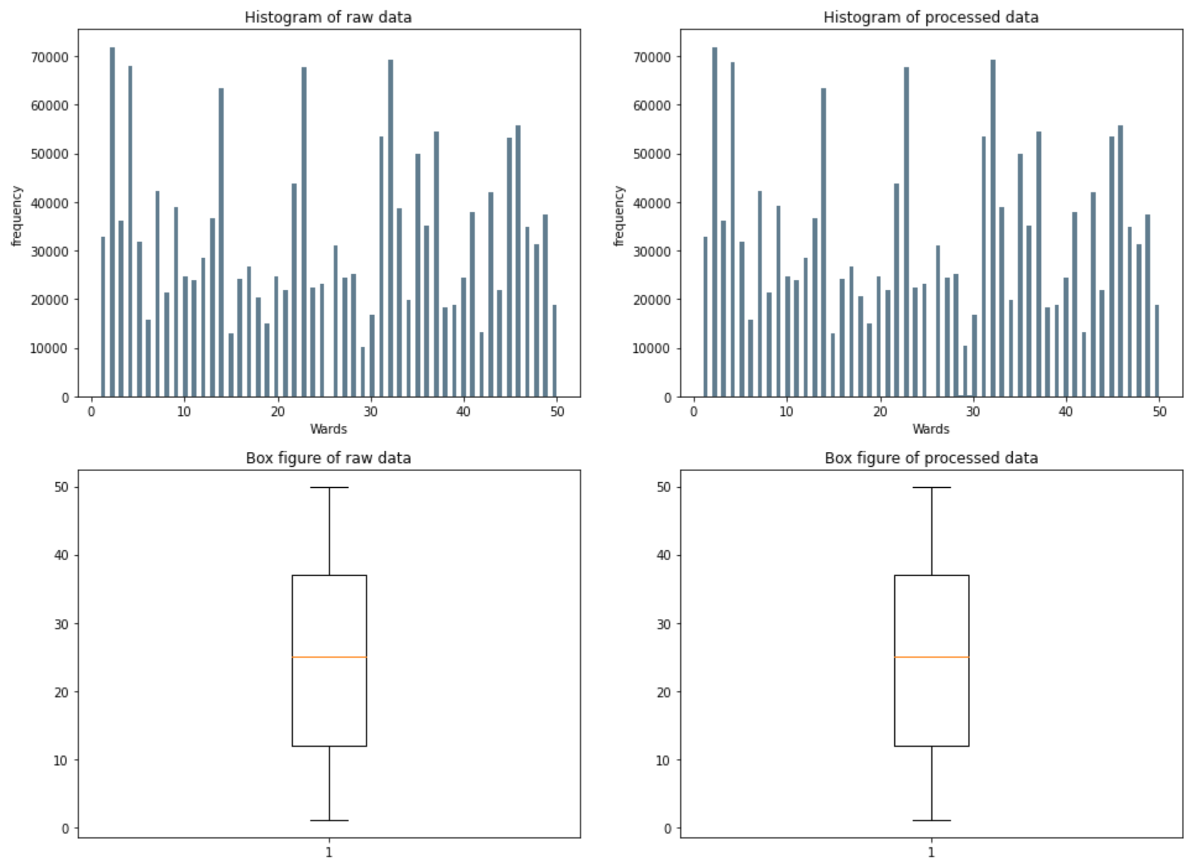
```
att = ['Wards']
show_hist(data_lost3, att, 100)
show_boxplot(data_lost3,att)
```





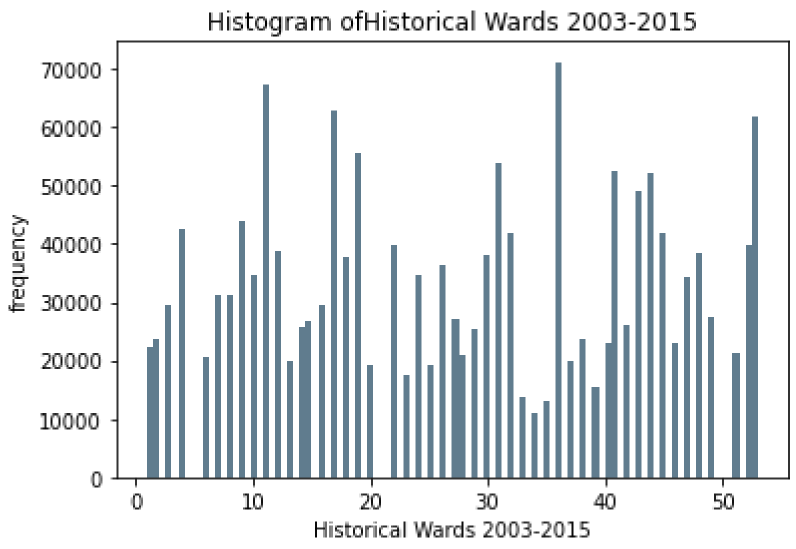
与原数据比较代码及结果为:

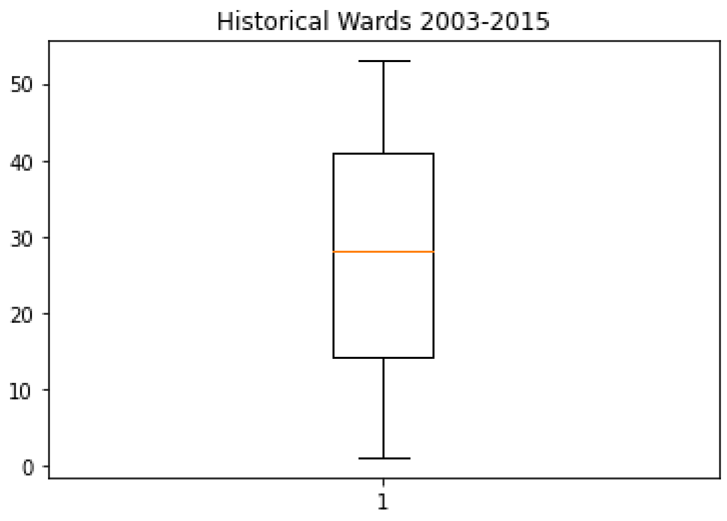
```
att = 'Wards'
subplot_show(data, att, data_lost3, att, 100, 'raw data', 'processed data')
```



对数据处理之后Historical Wards 2003-2015的直方图与盒图的代码及结果为:

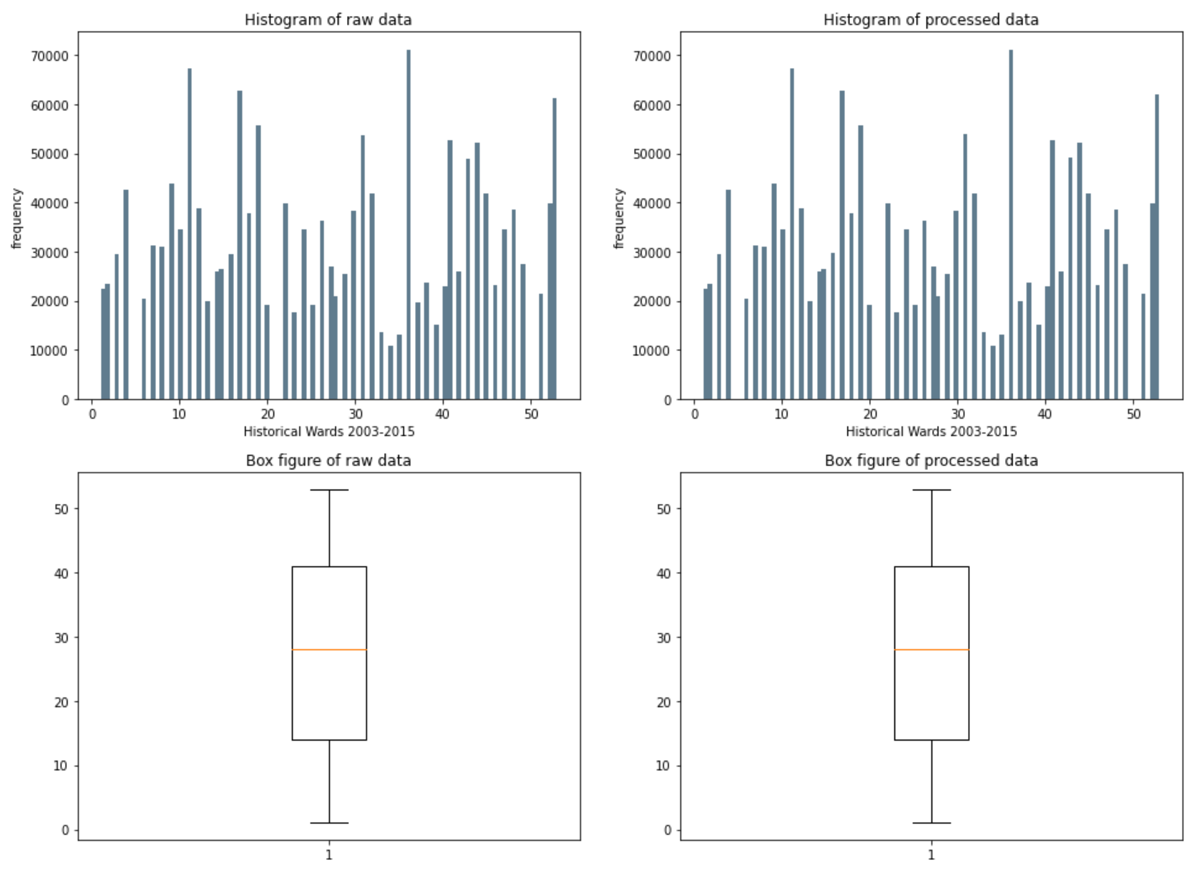
```
att = ['Historical Wards 2003-2015']
show_hist(data_lost3, att, 100)
show_boxplot(data_lost3,att)
```





与原数据比较代码及结果为：

```
att = 'Historical Wards 2003-2015'
subplot_show(data, att, data_lost3, att, 100, 'raw data', 'processed data')
```



通过属性的相关关系来填补缺失值

我们使用随机森林的方法对空缺的数据进行填充。按照第二种方式处理之后的数据五数概括的代码及结果为：

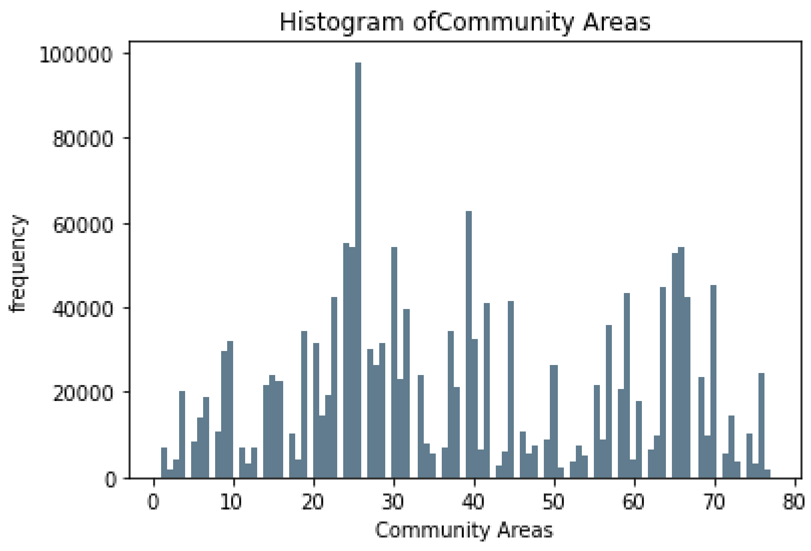
```
def set_missing_prices(data_temp, att_list):

    data_lost3 = data_temp.copy()

    # 把已有的数值型特征取出来丢进Random Forest Regressor中

    # 首先根据INSPECTION NUMBER', 'PROPERTY GROUP'填充 'Community Areas'的数据
    Community_df = data_lost3[att_list]
    goal = att_list[0]

    # 乘客分成已知年龄和未知年龄两部分
    temp = Community_df[Community_df[goal].notnull()]
    print(temp.isnull().sum())

    known_Community = Community_df[Community_df[goal].notnull()].values
    unknown_Community = Community_df[Community_df[goal].isnull()].values

    # y即目标值
    y = known_Community[:, 0]

    # X即特征属性值
    X = known_Community[:, 1:]

    # fit到RandomForestRegressor之中
    rfr = sklearn.ensemble.RandomForestRegressor(random_state=0,
n_estimators=2000, n_jobs=-1)
    rfr.fit(X, y)

    # 用得到的模型进行未知年龄结果预测
    predictedprice = rfr.predict(unknown_Community[:, 1:])
#     print predictedAges
    # 用得到的预测结果填补原缺失数据
    data_lost3.loc[ (data_lost3[goal].isnull()), goal ] = predictedprice


    return data_lost3


def lost_edition3(complete_list, uncomplete_list ):
    data_lost3 = data.copy()

    for i in uncomplete_list:

        complete_list.insert(0,i)
        print(complete_list)
        data_lost3 = set_missing_prices(data_lost3, complete_list)
        print(data_lost3.isnull().sum())

    return data_lost3




save_or_not = False

if save_or_not:
    data_lost3 = lost_edition3(att_num)
    print(data_lost2.isnull().sum())
    data_lost2.to_csv('data_mode.csv')

data_lost3_name_1 = r'C:\Users\Xiaomeng Fan\Desktop\研究生的各种作业\数据挖掘\作业四
\building\data_forest.csv'
file_lost3_1 = pd.read_csv(data_lost3_name_1)
data_lost3_1 = pd.DataFrame(file_lost3_1)
print(data_lost3_1.isnull().sum())
for i in att_num:
    data_i_np = data_lost3_1.loc[:,i].values
    min_temp, Q1, median, Q3, max_temp = fiveNumber(data_i_np)
    print(i+' five number:')
    print(min_temp,Q1,median,Q3,max_temp)
```

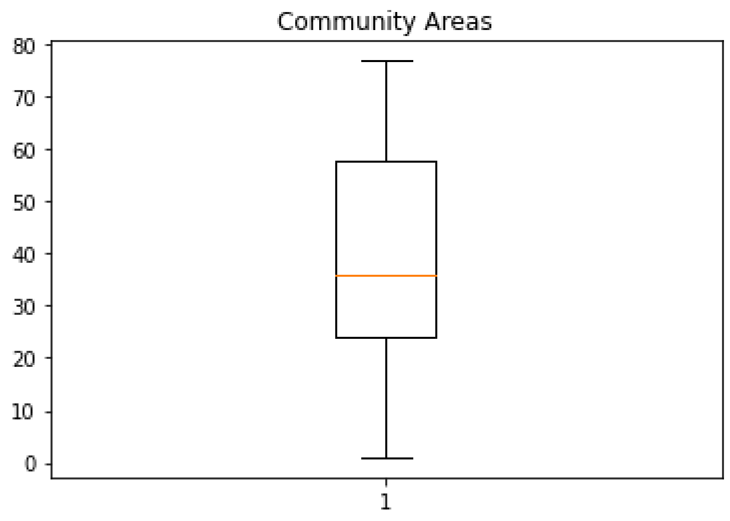
```
Unnamed: 0                        0
ID                                0
VIOLATION LAST MODIFIED DATE      0
VIOLATION DATE                    0
VIOLATION CODE                    0
VIOLATION STATUS                  0
VIOLATION STATUS DATE       1036199
VIOLATION DESCRIPTION         10768
VIOLATION LOCATION           897282
VIOLATION INSPECTOR COMMENTS      175463
VIOLATION ORDINANCE                47581
INSPECTOR ID                           0
INSPECTION NUMBER                      0
INSPECTION STATUS                     16
INSPECTION WAIVED                      0
INSPECTION CATEGORY                    0
DEPARTMENT BUREAU                      0
ADDRESS                                0
STREET NUMBER                          0
STREET DIRECTION                       0
STREET NAME                            0
STREET TYPE                        13541
PROPERTY GROUP                         0
SSA                              1356267
LATITUDE                            1510
LONGITUDE                           1510
LOCATION                            1510
Community Areas                        0
Zip Codes                           1510
Boundaries - ZIP Codes              2279
Census Tracts                          0
Wards                                  0
Historical Wards 2003-2015             0
dtype: int64
INSPECTION NUMBER five number:
265575 2304416.0 10418746.0 11687280.0 13050915
PROPERTY GROUP five number:
1000 20560.0 154323.0 366984.0 677975
Community Areas five number:
1.0 24.0 36.0 58.0 77.0
Census Tracts five number:
1.0 179.0 374.0 572.0 801.0
Wards five number:
1.0 12.0 25.0 37.0 50.0
Historical Wards 2003-2015 five number:
1.0 14.0 28.0 41.0 53.0
```

原数据的INSPECTION NUMBER属性的五数概括为：[265575 2304416.0 10418746.0 11687280.0 13050915], 数据处理之后的五数概括为[265575 2304416.0 10418746.0 11687280.0 13050915]。
原数据的PROPERTY GROUP属性的五数概括为：[1000 20560.0 154323.0 366984.0 677975], 数据处理之后的五数概括为[1000 20560.0 154323.0 366984.0 677975]。
原数据的Community Areas属性的五数概括为：[1.0 24.0 36.0 58.0 77.0], 数据处理之后的五数概括为[1.0 24.0 36.0 58.0 77.0]。
原数据的Census Tracts属性的五数概括为：[1.0 179.0 374.0 572.0 801.0], 数据处理之后的五数概括为[1.0 179.0 374.0 572.0 801.0]。
原数据的Wards属性的五数概括为：[1.0 12.0 25.0 37.0 50.0], 数据处理之后的五数概括为[1.0 12.0 25.0 37.0 50.0]。
原数据的PHistorical Wards 2003-2015属性的五数概括为：[1.0 14.0 28.0 41.0 53.0], 数据处理之后的五数概括为[1.0 14.0 28.0 41.0 53.0]。

对数据处理之后Community Areas的直方图与盒图的代码及结果为：

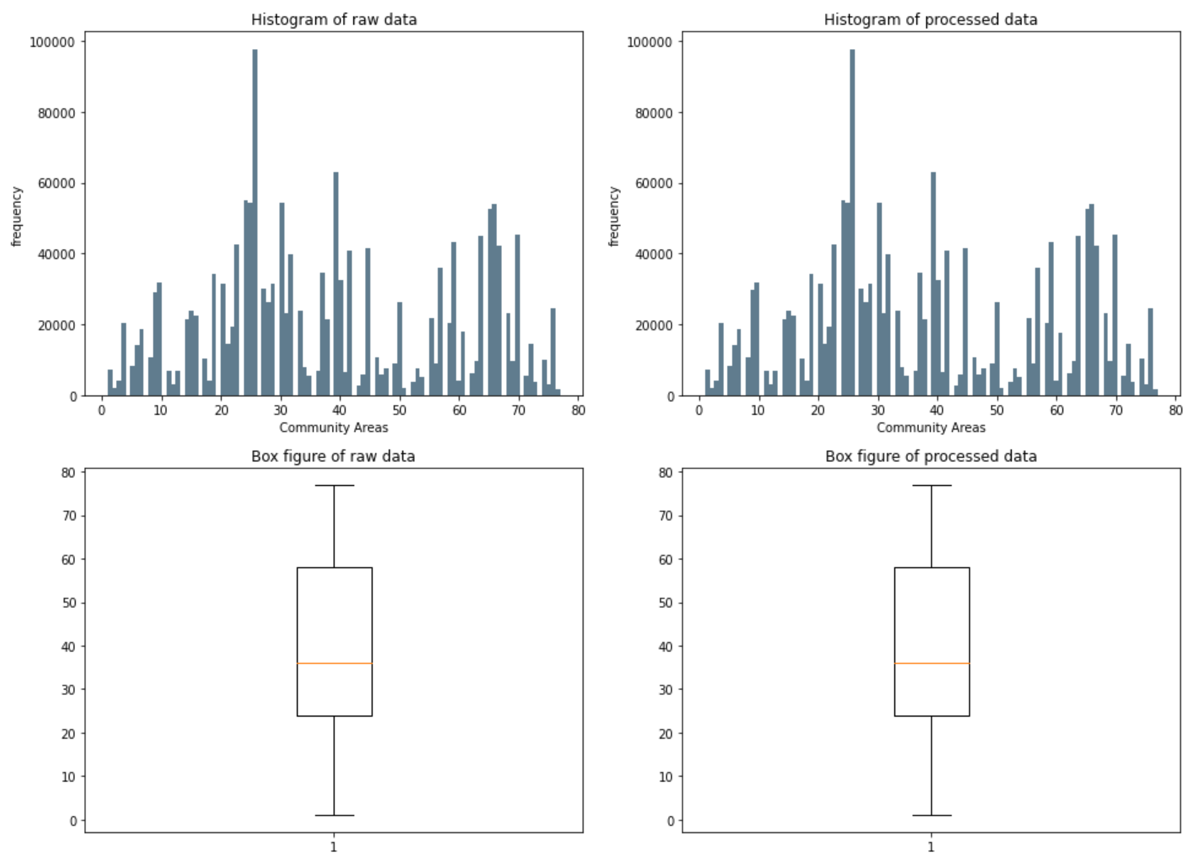
```
att = ['Community Areas']
show_hist(data_lost3_1, att, 100)
show_boxplot(data_lost3_1,att)
```





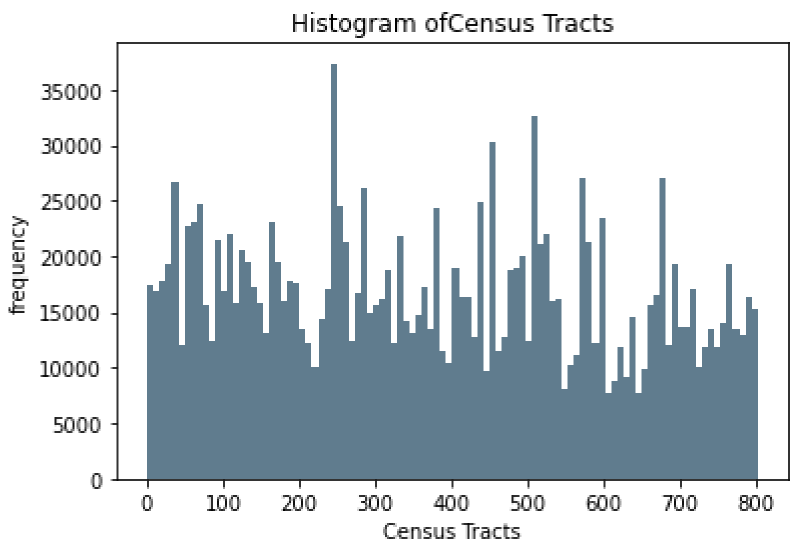
与原数据比较代码及结果为：

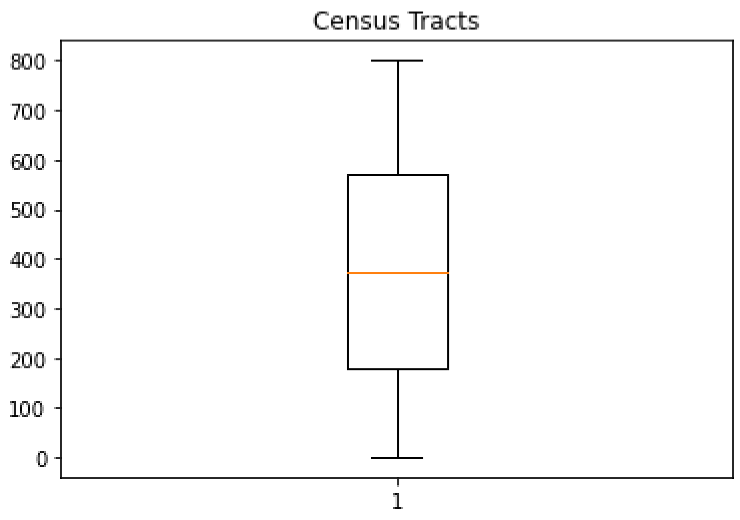
```
att = 'Community Areas'
subplot_show(data, att, data_lost3_1, att, 100, 'raw data', 'processed data')
```



对数据处理之后Census Tracts的直方图与盒图的代码及结果为：

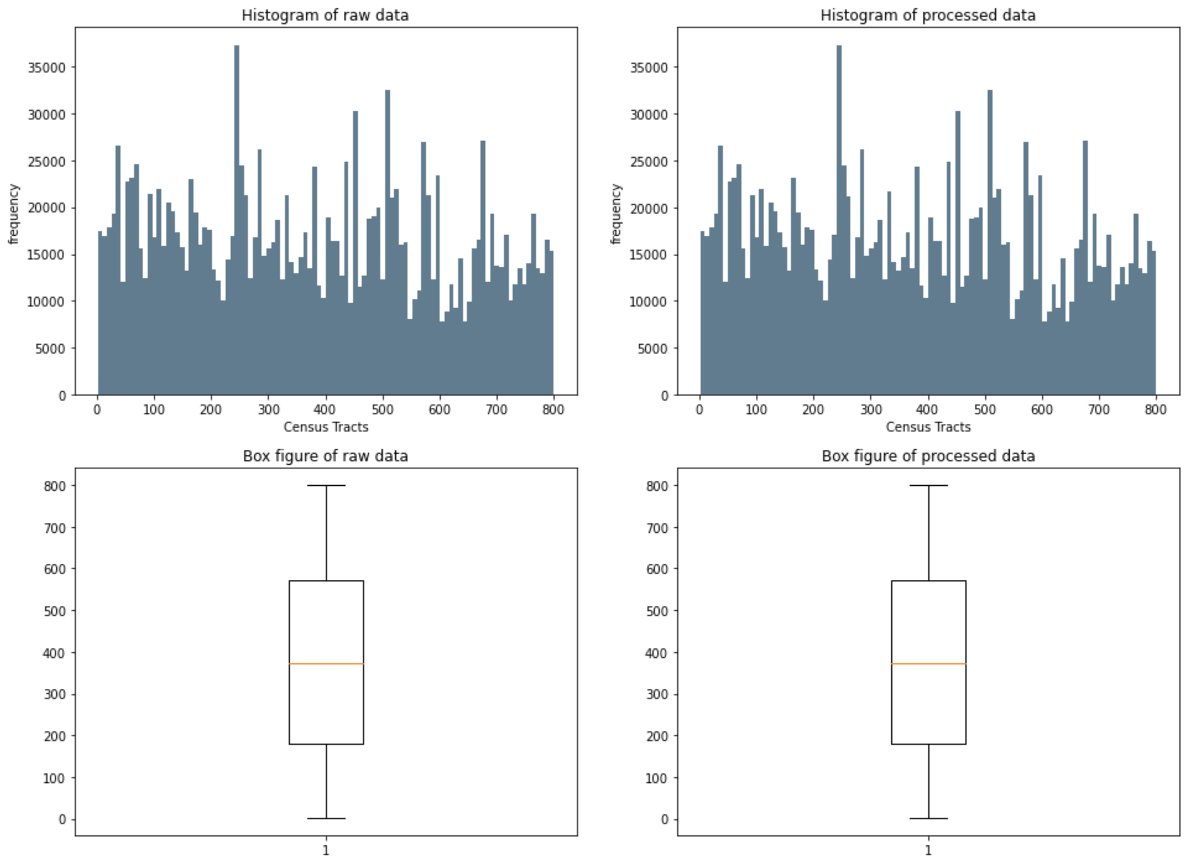
```
att = ['Census Tracts']
show_hist(data_lost3_1, att, 100)
show_boxplot(data_lost3_1,att)
```





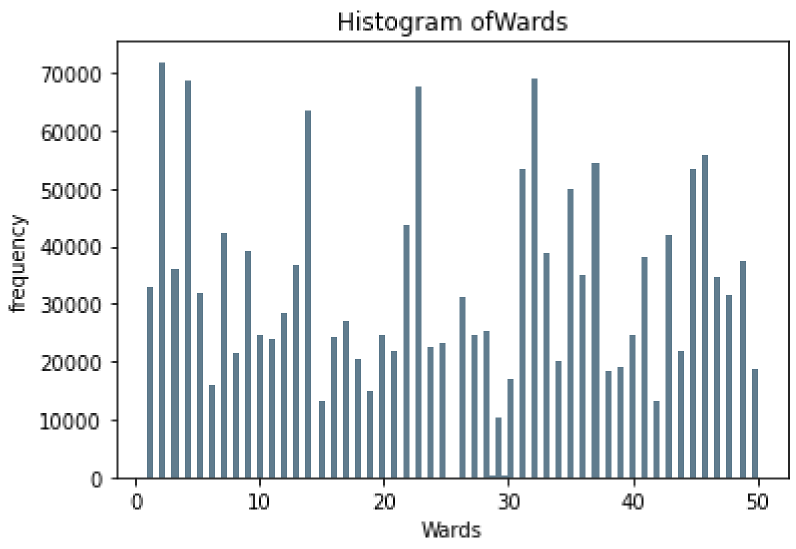
与原数据比较代码及结果为：

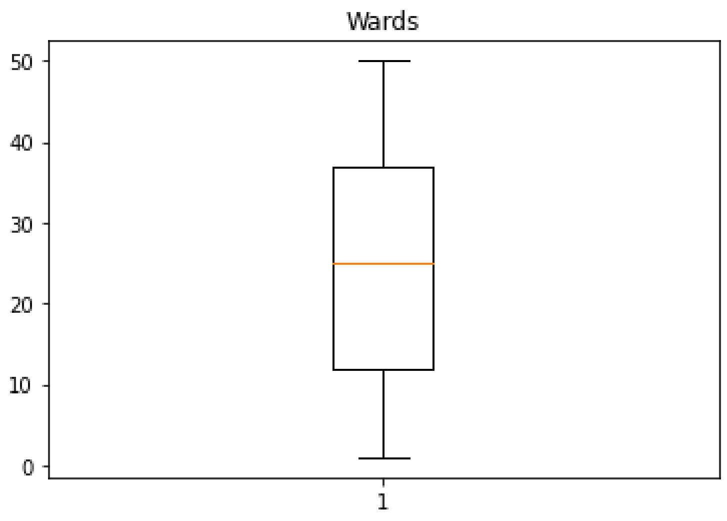
```
att = 'Census Tracts'
subplot_show(data, att, data_lost3_1, att, 100, 'raw data', 'processed data')
```



对数据处理之后Wards的直方图与盒图的代码及结果为：

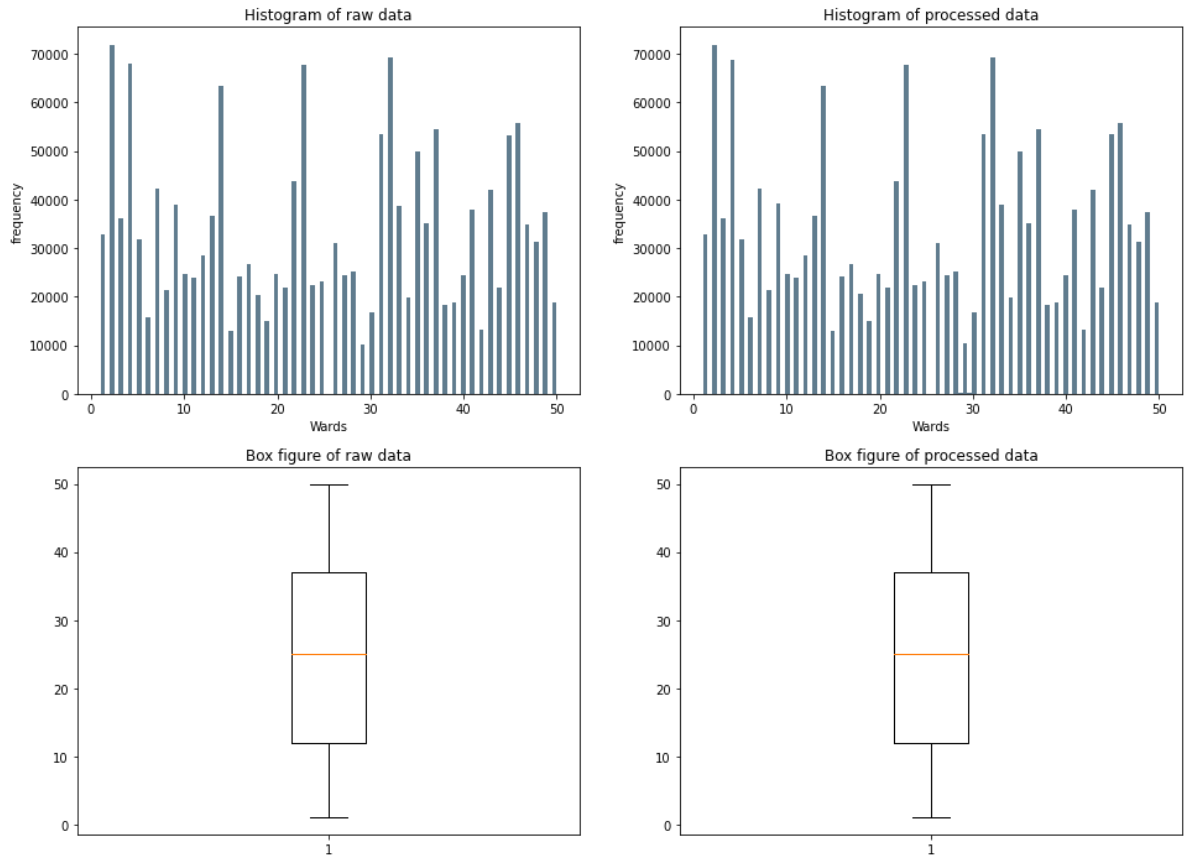
```
att = ['Wards']
show_hist(data_lost3_1, att, 100)
show_boxplot(data_lost3_1,att)
```





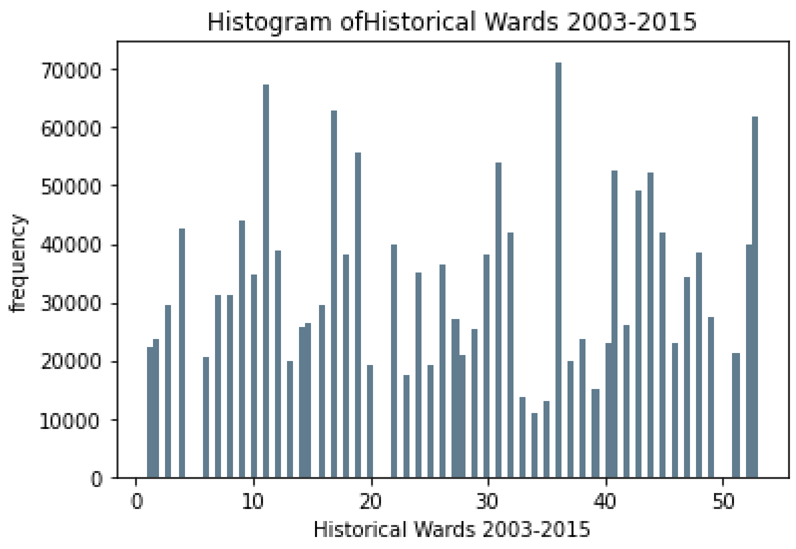
与原数据比较代码及结果为：

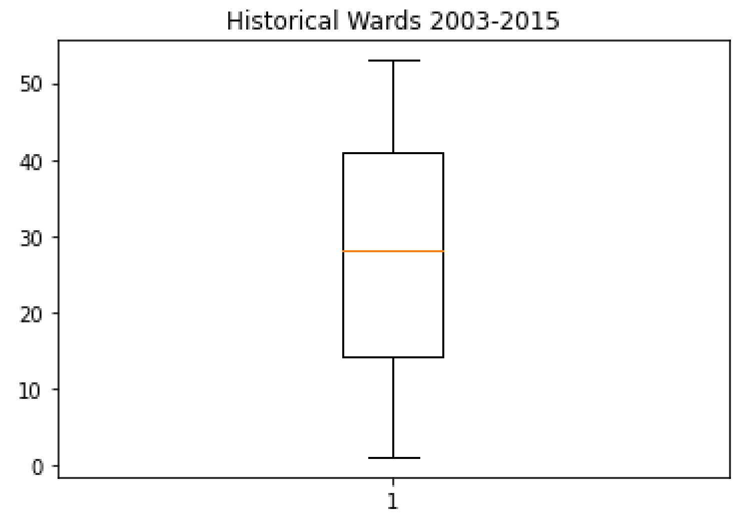
```python
att = 'Wards'
subplot_show(data, att, data_lost3_1, att, 100, 'raw data', 'processed data')
```



对数据处理之后Historical Wards 2003-2015的直方图与盒图的代码及结果为：

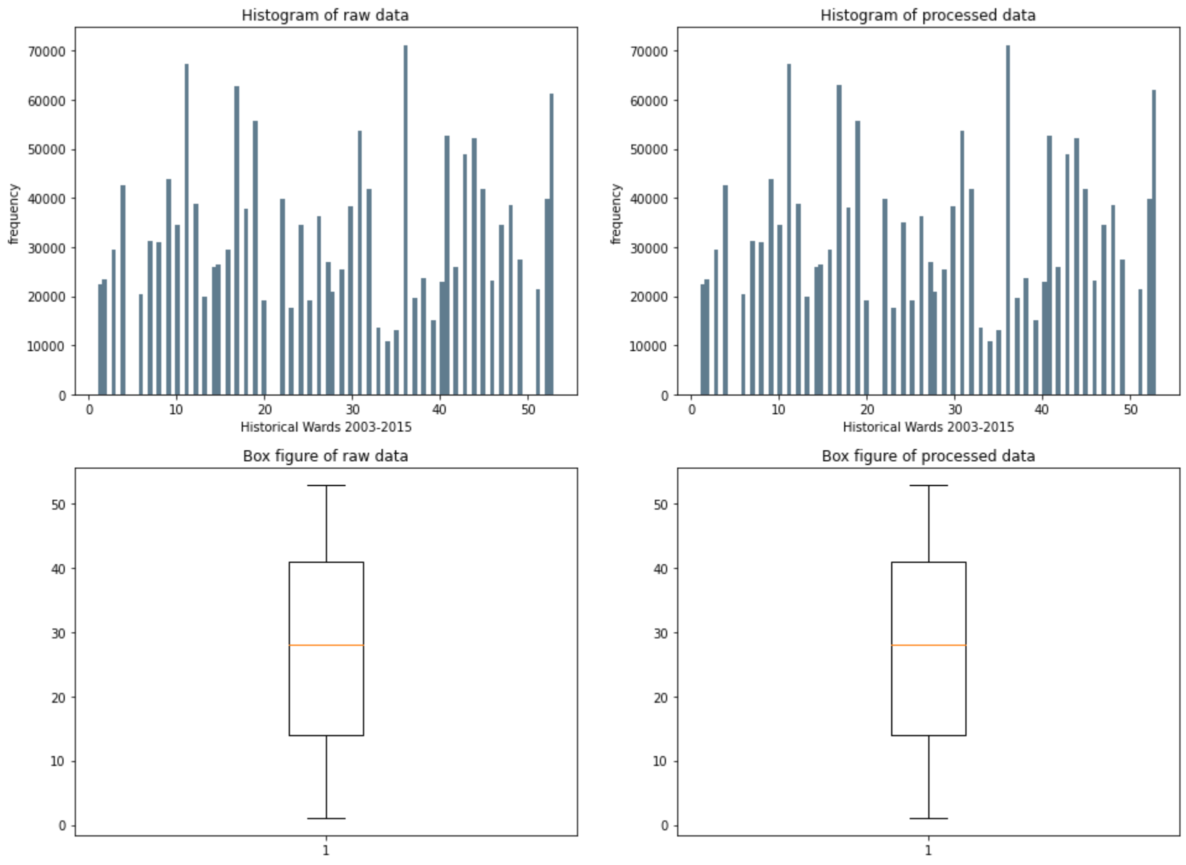
```
att = ['Historical Wards 2003-2015']
show_hist(data_lost3_1, att, 100)
show_boxplot(data_lost3_1,att)
```





与原数据比较代码及结果为：

```
att = 'Historical Wards 2003-2015'
subplot_show(data, att, data_lost3_1, att, 100, 'raw data', 'processed data')
```



通过数据对象之间的相似性来填补缺失值

我们使用knn的方法对空缺的数据进行填充。在这里，我们采用了两种方式，分别为，利用完整的数据属性'INSPECTION NUMBER', 'PROPERTY GROUP'分别对空缺的数据属性进行填充，第二种是利用完整的数据属性'INSPECTION NUMBER', 'PROPERTY GROUP'依次对空缺的数据属性数据进行填充，并将填充的数据与'INSPECTION NUMBER', 'PROPERTY GROUP'一起作为参考数据对不完整的属性数据填充。由于数据量较大，随机森林方法需要的计算资源与存储资源较大，所以我们采用在服务器实现、保存数据，对数据分析的方式。按照第一种方式处理之后的数据五数概括的代码及结果为：

```
def knn_missing_filled(target, complete_list, data, k = 3, dispersed = True):



    data_temp = data.copy()

    complete_list_temp = complete_list.copy()
    complete_list_temp.insert(0, target)
    print(complete_list_temp)

    data_temp_temp = data_temp[complete_list_temp]

    x_train = data_temp_temp[data_temp_temp[target].notnull()].values[:,1:]

    y_train =
data_temp_temp[data_temp_temp[target].notnull()].values[:,0].reshape(-1,1)

    test = data_temp_temp[data_temp_temp[target].isnull()].values[:,1:]

    if dispersed:
        clf = KNeighborsClassifier(n_neighbors = k, weights = "distance")
    else:
        clf = KNeighborsRegressor(n_neighbors = k, weights = "distance")

    clf.fit(x_train, y_train)

    data_temp.loc[ (data_temp[target].isnull()), target ] = clf.predict(test)

    return data_temp


def lost_edition4(complete_list, uncomplete_list):

    data_lost4 = data.copy()

    for i in uncomplete_list:

        # complete_list.insert(0, i)
        data_lost4 = knn_missing_filled(i, complete_list, data_lost4)
        print(data_lost4.isnull().sum())

    return data_lost4




save_or_not = False

if save_or_not:
    data_lost4 = lost_edition4(att_num)
    print(data_lost4.isnull().sum())
    data_lost4.to_csv('data_mode.csv')

data_lost4_name = r'C:\Users\Xiaomeng Fan\Desktop\研究生的各种作业\数据挖掘\作业四
\building\data_knn_3.csv'
file_lost4 = pd.read_csv(data_lost4_name)
data_lost4 = pd.DataFrame(file_lost4)
```

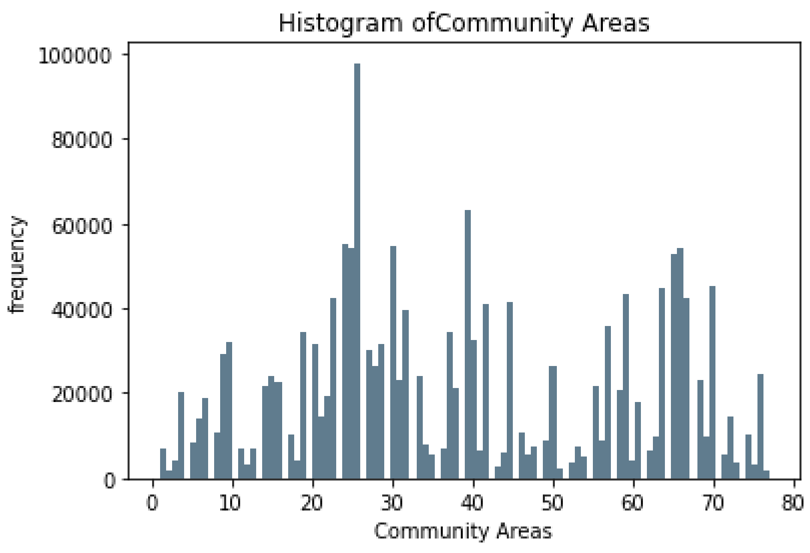
```
print(data_lost4.isnull().sum())
for i in att_num:
    data_i_np = data_lost4.loc[:,i].values
    min_temp, Q1, median, Q3, max_temp = fiveNumber(data_i_np)
    print(i+' five number:')
    print(min_temp,Q1,median,Q3,max_temp)
```

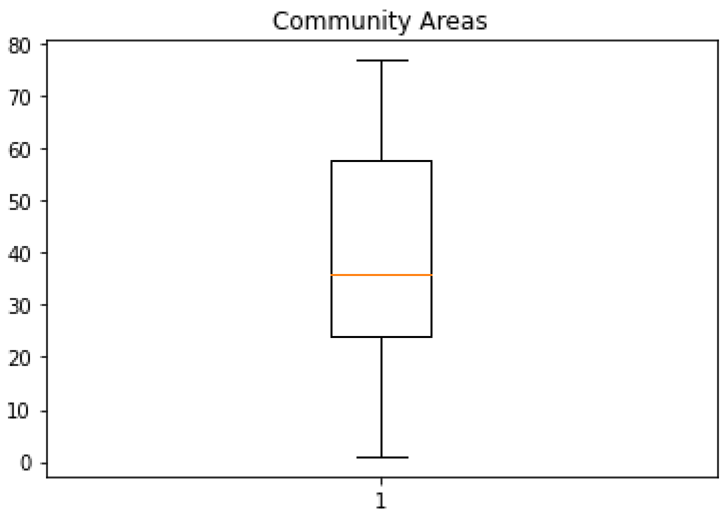
```
Unnamed: 0                           0
ID                                   0
VIOLATION LAST MODIFIED DATE         0
VIOLATION DATE                       0
VIOLATION CODE                       0
VIOLATION STATUS                     0
VIOLATION STATUS DATE          1036199
VIOLATION DESCRIPTION            10768
VIOLATION LOCATION              897282
VIOLATION INSPECTOR COMMENTS    175463
VIOLATION ORDINANCE              47581
INSPECTOR ID                         0
INSPECTION NUMBER                    0
INSPECTION STATUS                   16
INSPECTION WAIVED                    0
INSPECTION CATEGORY                  0
DEPARTMENT BUREAU                    0
ADDRESS                              0
STREET NUMBER                        0
STREET DIRECTION                     0
STREET NAME                          0
STREET TYPE                      13541
PROPERTY GROUP                       0
SSA                            1356267
LATITUDE                          1510
LONGITUDE                         1510
LOCATION                          1510
Community Areas                      0
Zip Codes                         1510
Boundaries - ZIP Codes            2279
Census Tracts                        0
Wards                                0
Historical Wards 2003-2015           0
dtype: int64
INSPECTION NUMBER five number:
265575 2304416.0 10418746.0 11687280.0 13050915
PROPERTY GROUP five number:
1000 20560.0 154323.0 366984.0 677975
Community Areas five number:
1.0 24.0 36.0 58.0 77.0
Census Tracts five number:
1.0 179.0 374.0 572.0 801.0
Wards five number:
1.0 12.0 25.0 37.0 50.0
Historical Wards 2003-2015 five number:
1.0 14.0 28.0 41.0 53.0
```

原数据的INSPECTION NUMBER属性的五数概括为：[265575 2304416.0 10418746.0 11687280.0 13050915], 数据处理之后的五数概括为[265575 2304416.0 10418746.0 11687280.0 13050915]。
原数据的PROPERTY GROUP属性的五数概括为：[1000 20560.0 154323.0 366984.0 677975], 数据处理之后的五数概括为[1000 20560.0 154323.0 366984.0 677975]。
原数据的Community Areas属性的五数概括为：[1.0 24.0 36.0 58.0 77.0], 数据处理之后的五数概括为[1.0 24.0 36.0 58.0 77.0]。
原数据的Census Tracts属性的五数概括为：[1.0 179.0 374.0 572.0 801.0], 数据处理之后的五数概括为[1.0 179.0 374.0 572.0 801.0]。
原数据的Wards属性的五数概括为：[1.0 12.0 25.0 37.0 50.0], 数据处理之后的五数概括为[1.0 12.0 25.0 37.0 50.0]。
原数据的PHistorical Wards 2003-2015属性的五数概括为：[1.0 14.0 28.0 41.0 53.0], 数据处理之后的五数概括为[1.0 14.0 28.0 41.0 53.0]。

对数据处理之后Community Areas的直方图与盒图的代码及结果为：

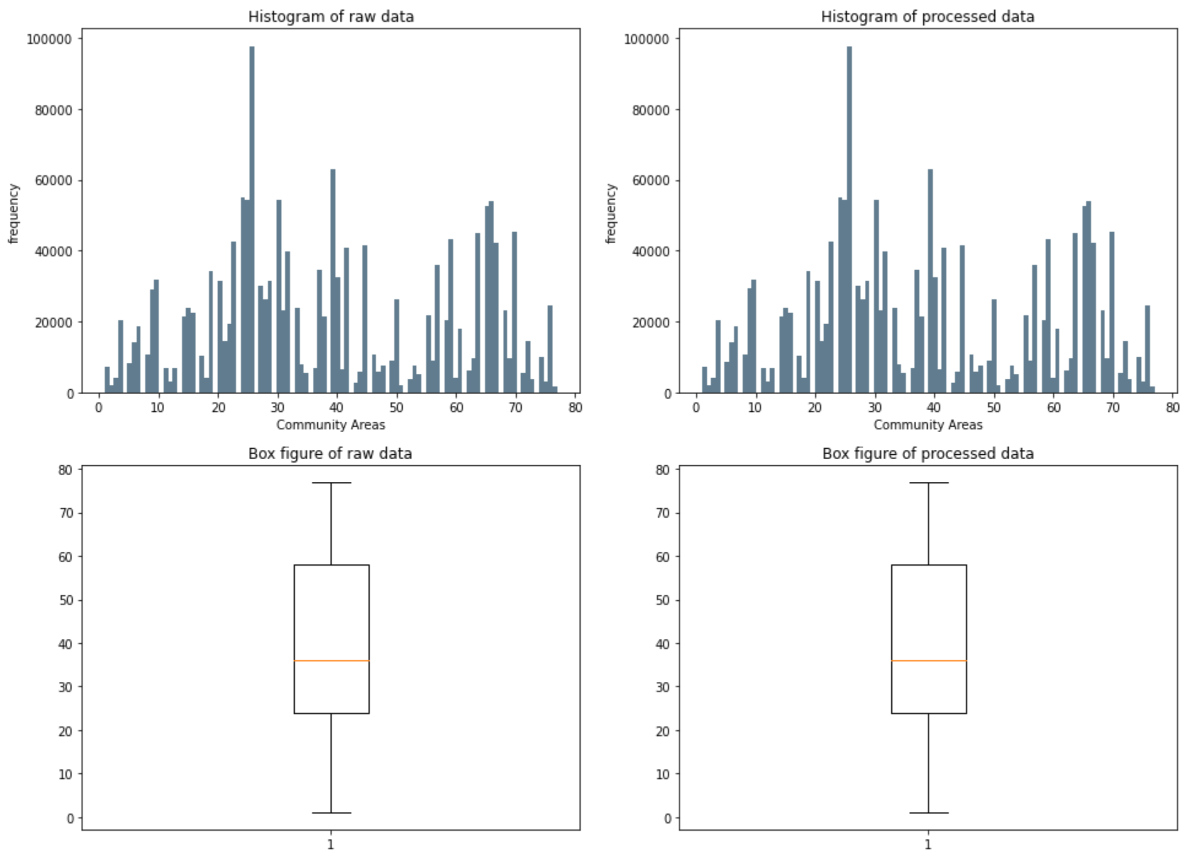
```
att = ['Community Areas']
show_hist(data_lost4, att, 100)
show_boxplot(data_lost4,att)
```





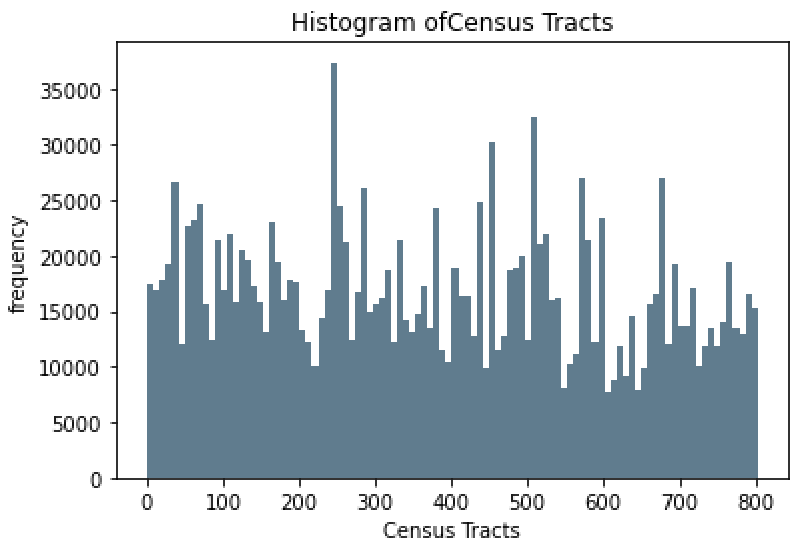
与原数据比较代码及结果为：

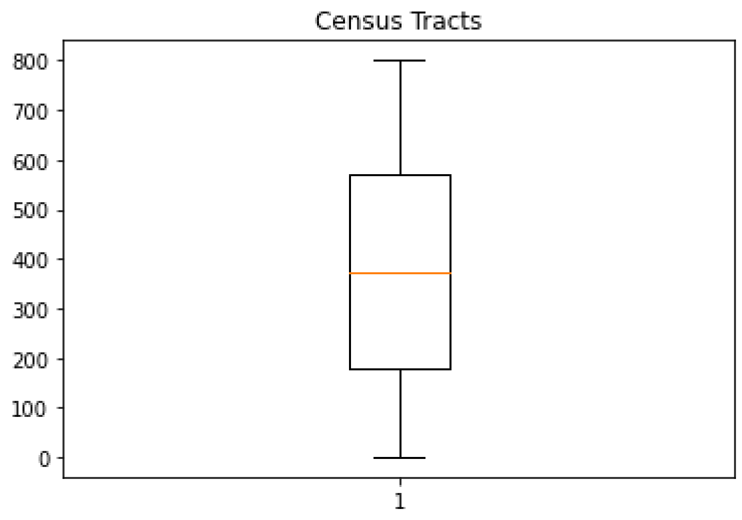
```
att = 'Community Areas'
subplot_show(data, att, data_lost4, att, 100, 'raw data', 'processed data')
```



对数据处理之后Census Tracts的直方图与盒图的代码及结果为：

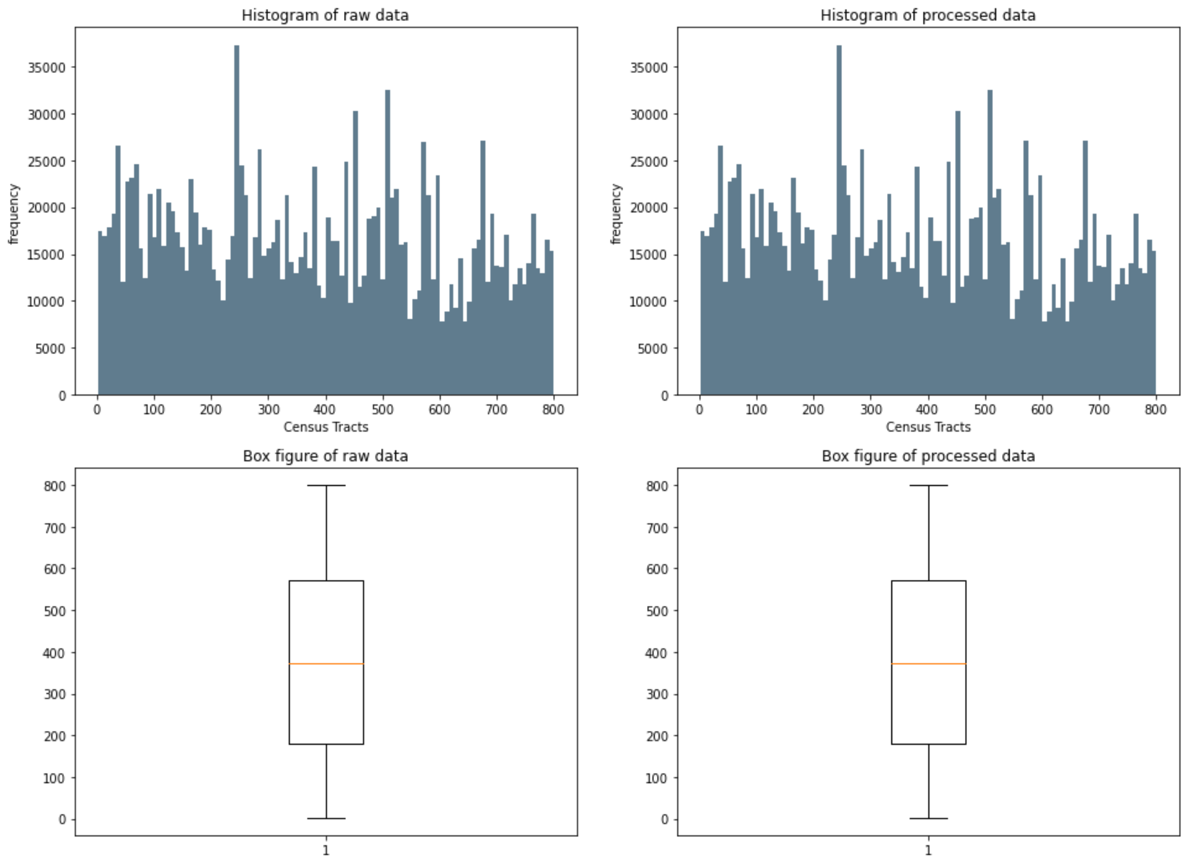
```
att = ['Census Tracts']
show_hist(data_lost4, att, 100)
show_boxplot(data_lost4,att)
```





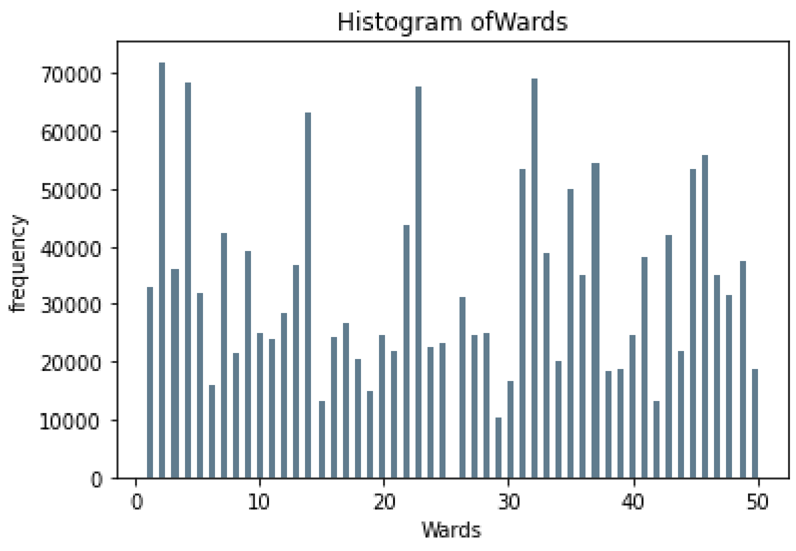
与原数据比较代码及结果为:

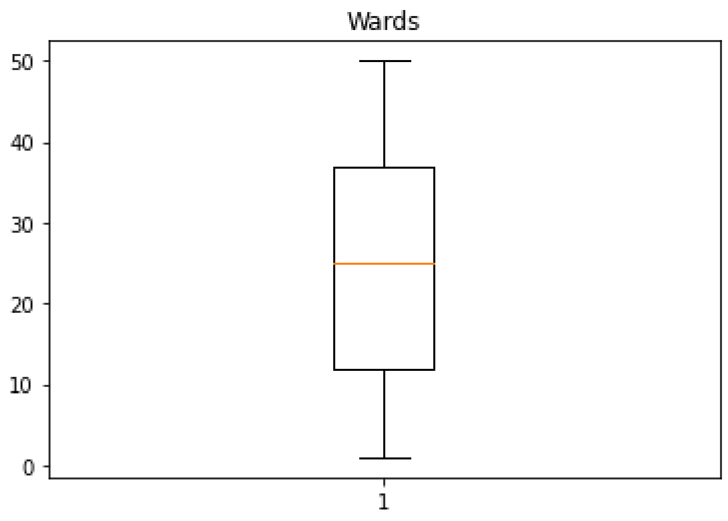
```
att = 'Census Tracts'
subplot_show(data, att, data_lost4, att, 100, 'raw data', 'processed data')
```



对数据处理之后Wards的直方图与盒图的代码及结果为:

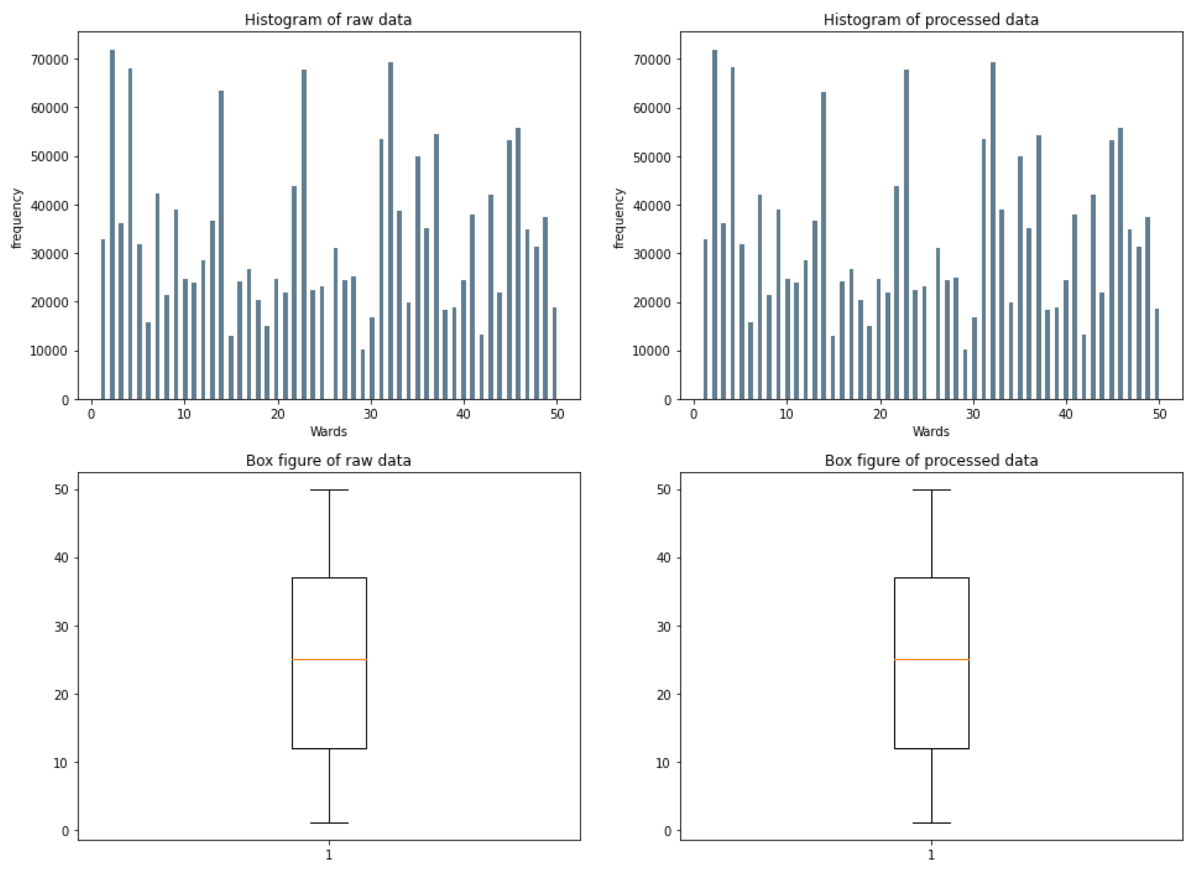
```
att = ['Wards']
show_hist(data_lost4, att, 100)
show_boxplot(data_lost4,att)
```





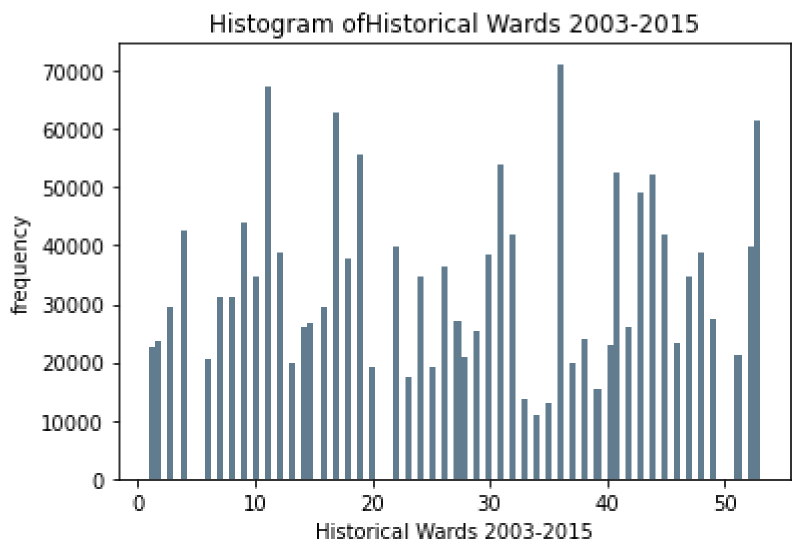
与原数据比较代码及结果为：

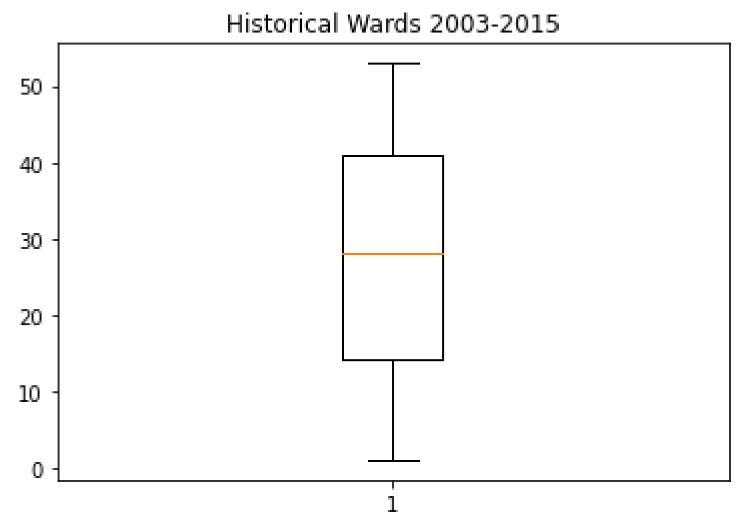
```
att = 'Wards'
subplot_show(data, att, data_lost4, att, 100, 'raw data', 'processed data')
```



对数据处理之后Historical Wards 2003-2015的直方图与盒图的代码及结果为:

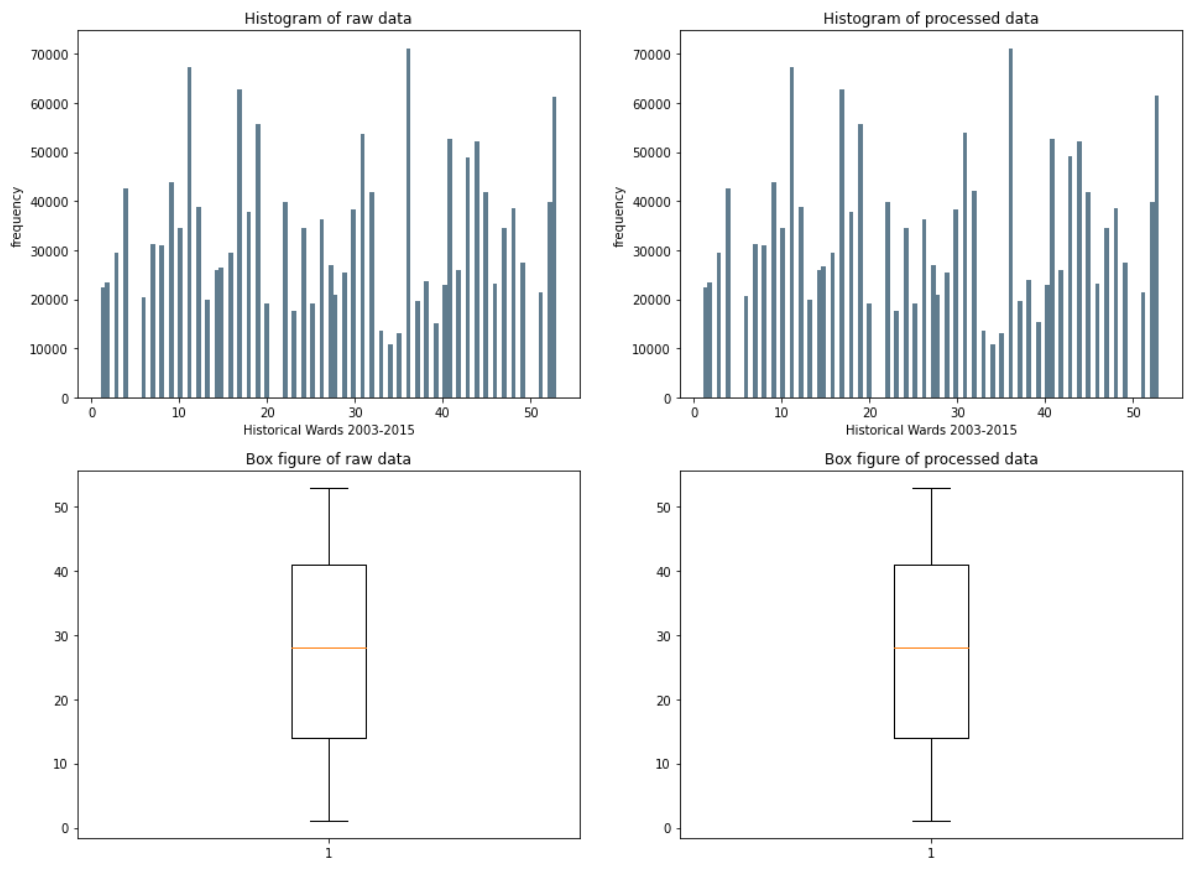
```
att = ['Historical Wards 2003-2015']
show_hist(data_lost4, att, 100)
show_boxplot(data_lost4,att)
```





与原数据比较代码及结果为：

```
att = 'Historical Wards 2003-2015'
subplot_show(data, att, data_lost4, att, 100, 'raw data', 'processed data')
```



按照第二种方式处理之后的数据五数概括的代码及结果为：

```
def knn_missing_filled(target, complete_list, data, k = 3, dispersed = True):


    data_temp = data.copy()
    complete_list_temp.insert(0, target)
    print(complete_list_temp)

    data_temp_temp = data_temp[complete_list_temp]

    x_train = data_temp_temp[data_temp_temp[target].notnull()].values[:,1:]

    y_train = 
data_temp_temp[data_temp_temp[target].notnull()].values[:,0].reshape(-1,1)

    test = data_temp_temp[data_temp_temp[target].isnull()].values[:,1:]

    if dispersed:
        clf = KNeighborsClassifier(n_neighbors = k, weights = "distance")
    else:
        clf = KNeighborsRegressor(n_neighbors = k, weights = "distance")

    clf.fit(x_train, y_train)

    data_temp.loc[ (data_temp[target].isnull()), target ] = clf.predict(test)

    return data_temp


def lost_edition4(complete_list, uncomplete_list):

    data_lost4 = data.copy()

    for i in uncomplete_list:

        # complete_list.insert(0, i)
        data_lost4 = knn_missing_filled(i, complete_list, data_lost4)
        print(data_lost4.isnull().sum())

    return data_lost4




save_or_not = False

if save_or_not:
    data_lost4_1 = lost_edition4(att_num)
    print(data_lost4_1.isnull().sum())
    data_lost4_1.to_csv('data_mode.csv')

data_lost4_name_1 = r'C:\Users\Xiaomeng Fan\Desktop\研究生的各种作业\数据挖掘\作业四
\building\data_knn.csv'
file_lost4_1 = pd.read_csv(data_lost4_name_1)
data_lost4_1 = pd.DataFrame(file_lost4_1)
print(data_lost4_1.isnull().sum())
for i in att_num:
    data_i_np = data_lost4_1.loc[:,i].values
    min_temp, Q1, median, Q3, max_temp = fiveNumber(data_i_np)
    print(i+' five number:')
    print(min_temp,Q1,median,Q3,max_temp)
```

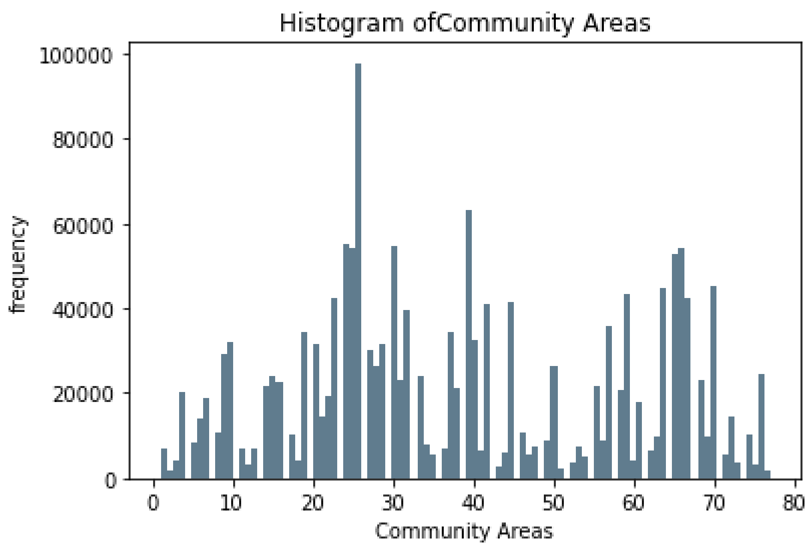
```
Unnamed: 0                              0
ID                                      0
VIOLATION LAST MODIFIED DATE            0
VIOLATION DATE                          0
VIOLATION CODE                          0
VIOLATION STATUS                        0
VIOLATION STATUS DATE             1036199
VIOLATION DESCRIPTION               10768
VIOLATION LOCATION                 897282
VIOLATION INSPECTOR COMMENTS       175463
VIOLATION ORDINANCE                 47581
INSPECTOR ID                            0
INSPECTION NUMBER                       0
INSPECTION STATUS                      16
INSPECTION WAIVED                       0
INSPECTION CATEGORY                     0
DEPARTMENT BUREAU                       0
ADDRESS                                 0
STREET NUMBER                           0
STREET DIRECTION                        0
STREET NAME                             0
STREET TYPE                         13541
PROPERTY GROUP                          0
SSA                               1356267
LATITUDE                             1510
LONGITUDE                            1510
LOCATION                             1510
Community Areas                         0
Zip Codes                            1510
Boundaries - ZIP Codes               2279
Census Tracts                           0
Wards                                   0
Historical Wards 2003-2015              0
dtype: int64
INSPECTION NUMBER five number:
265575 2304416.0 10418746.0 11687280.0 13050915
PROPERTY GROUP five number:
1000 20560.0 154323.0 366984.0 677975
Community Areas five number:
1.0 24.0 36.0 58.0 77.0
Census Tracts five number:
1.0 179.0 374.0 572.0 801.0
Wards five number:
1.0 12.0 25.0 37.0 50.0
Historical Wards 2003-2015 five number:
1.0 14.0 28.0 41.0 53.0
```

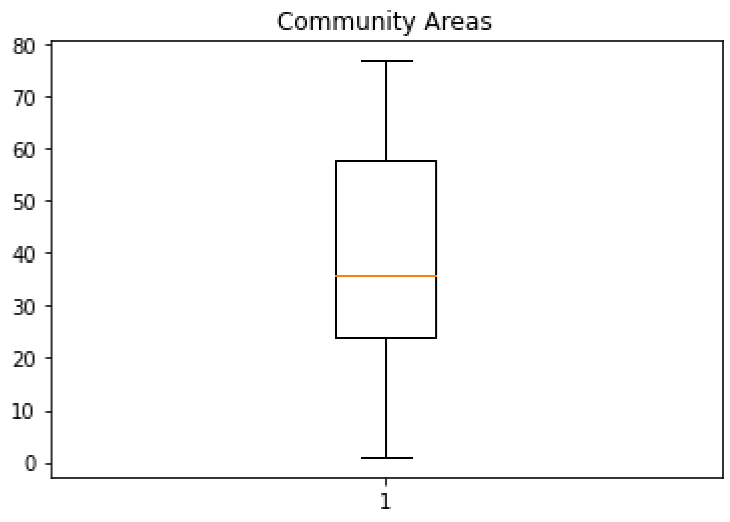
原数据的INSPECTION NUMBER属性的五数概括为：[265575 2304416.0 10418746.0 11687280.0 13050915], 数据处理之后的五数概括为[265575 2304416.0 10418746.0 11687280.0 13050915]。
原数据的PROPERTY GROUP属性的五数概括为：[1000 20560.0 154323.0 366984.0 677975], 数据处理之后的五数概括为[1000 20560.0 154323.0 366984.0 677975]。
原数据的Community Areas属性的五数概括为：[1.0 24.0 36.0 58.0 77.0], 数据处理之后的五数概括为[1.0 24.0 36.0 58.0 77.0]。
原数据的Census Tracts属性的五数概括为：[1.0 179.0 374.0 572.0 801.0], 数据处理之后的五数概括为[1.0 179.0 374.0 572.0 801.0]。

原数据的Wards属性的五数概括为：[1.0 12.0 25.0 37.0 50.0], 数据处理之后的五数概括为[1.0 12.0 25.0 37.0 50.0]。
原数据的PHistorical Wards 2003-2015属性的五数概括为：[1.0 14.0 28.0 41.0 53.0], 数据处理之后的五数概括为[1.0 14.0 28.0 41.0 53.0]。

对数据处理之后Community Areas的直方图与盒图的代码及结果为：

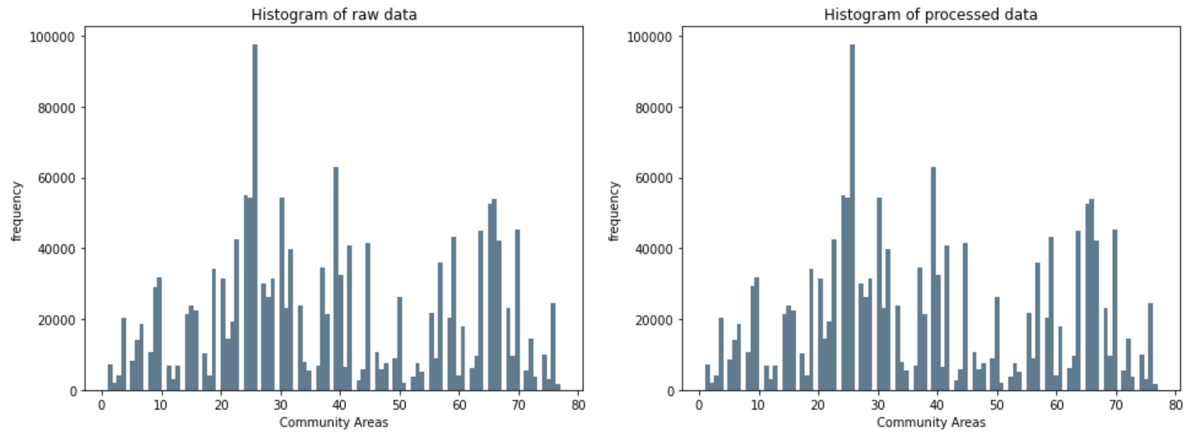
```
att = ['Community Areas']
show_hist(data_lost4_1, att, 100)
show_boxplot(data_lost4_1,att)
```

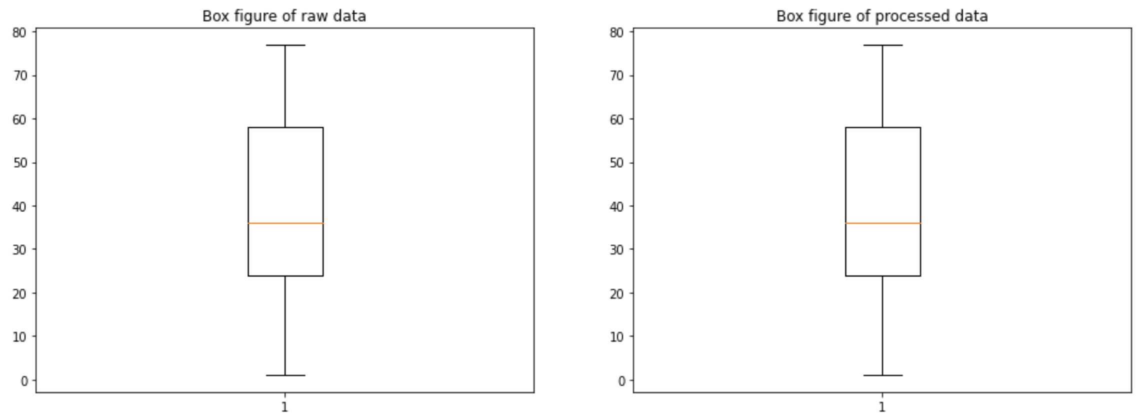




与原数据比较代码及结果为：

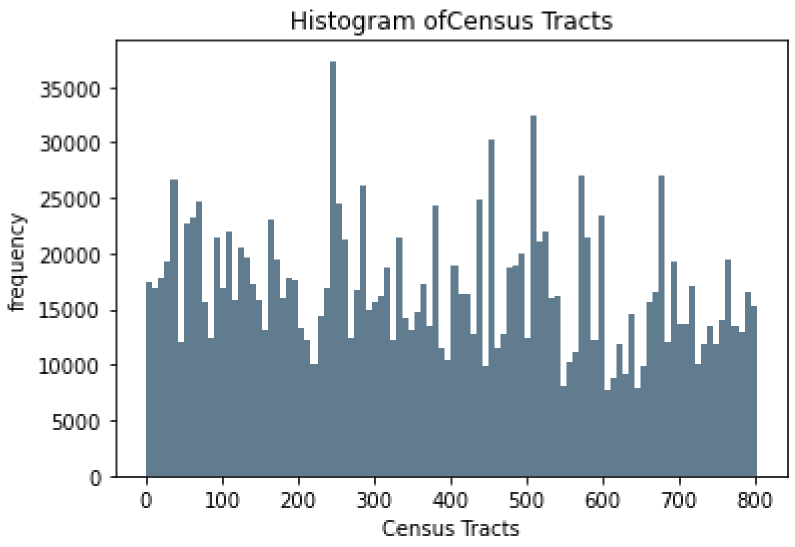
```
att = 'Community Areas'
subplot_show(data, att, data_lost4_1, att, 100, 'raw data', 'processed data')
```

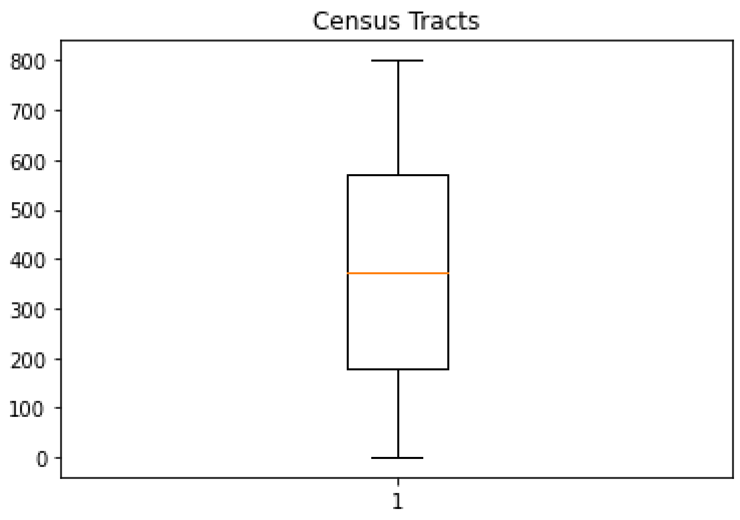




对数据处理之后Census Tracts的直方图与盒图的代码及结果为：

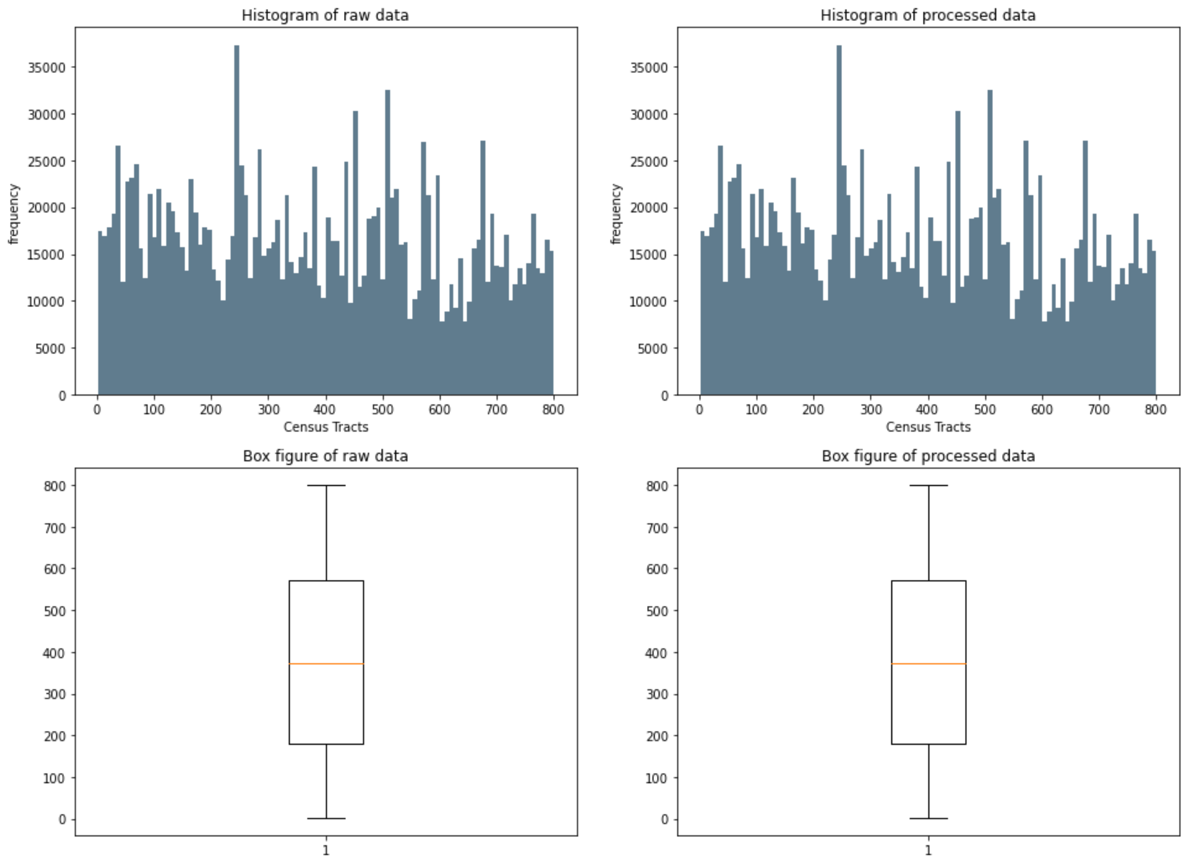
```python
att = ['Census Tracts']
show_hist(data_lost4_1, att, 100)
show_boxplot(data_lost4_1,att)
```





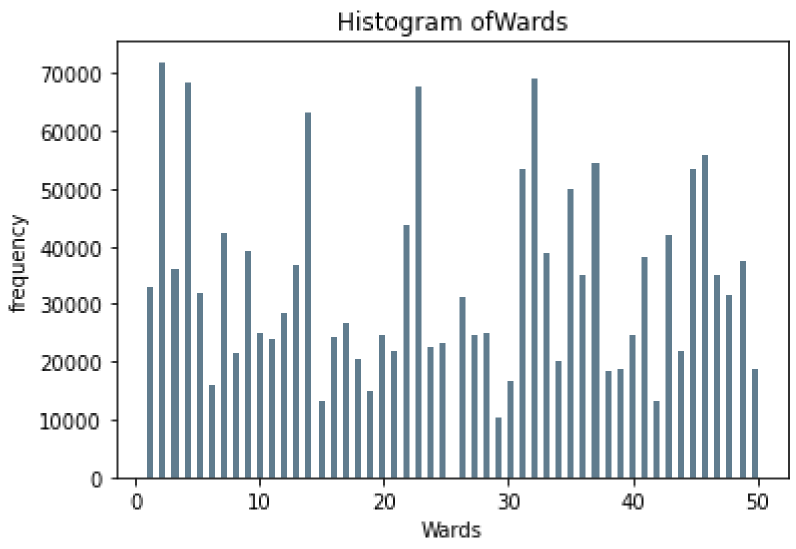
与原数据比较代码及结果为：

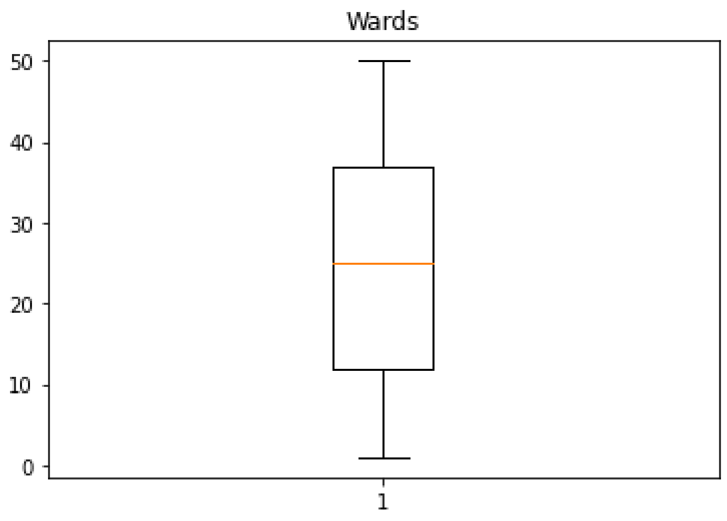
```
att = 'Census Tracts'
subplot_show(data, att, data_lost4_1, att, 100, 'raw data', 'processed data')
```



对数据处理之后Wards的直方图与盒图的代码及结果为：

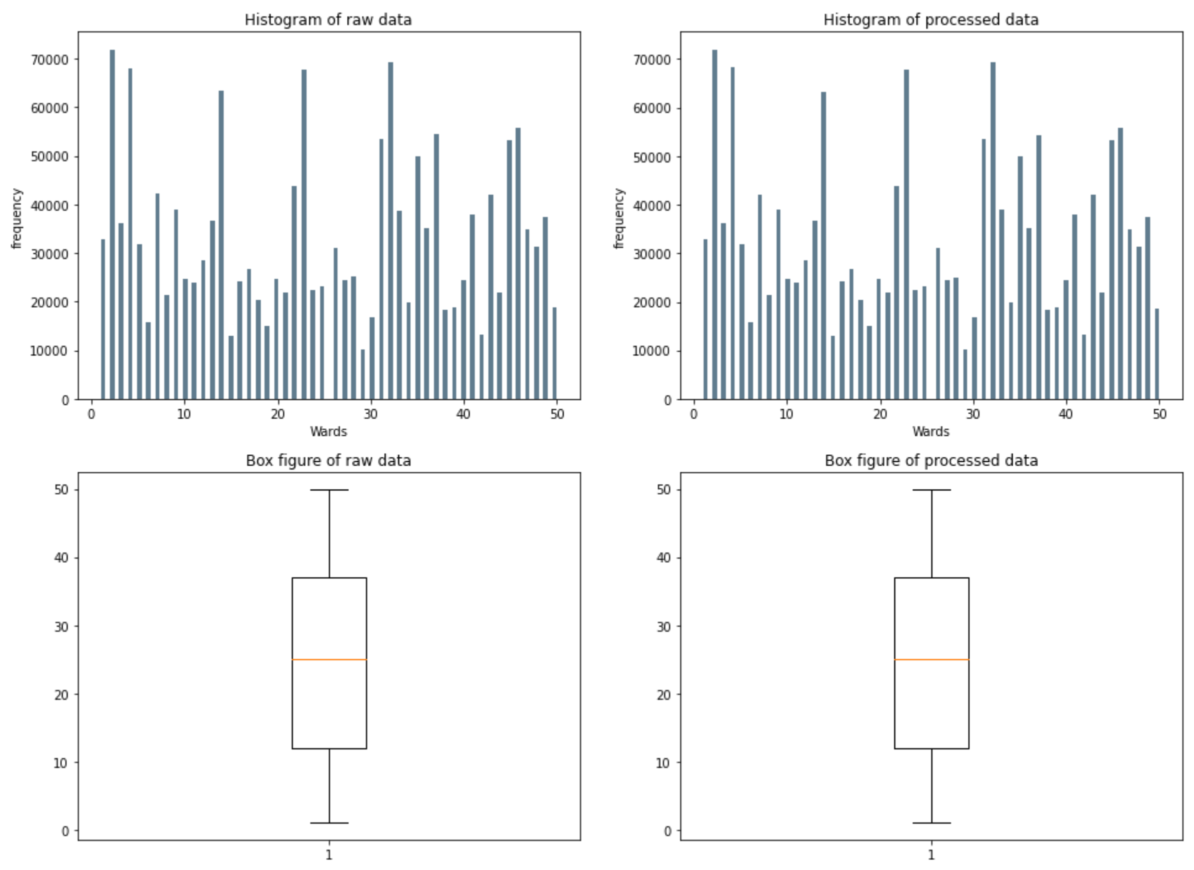
```
att = ['Wards']
show_hist(data_lost4_1, att, 100)
show_boxplot(data_lost4_1,att)
```





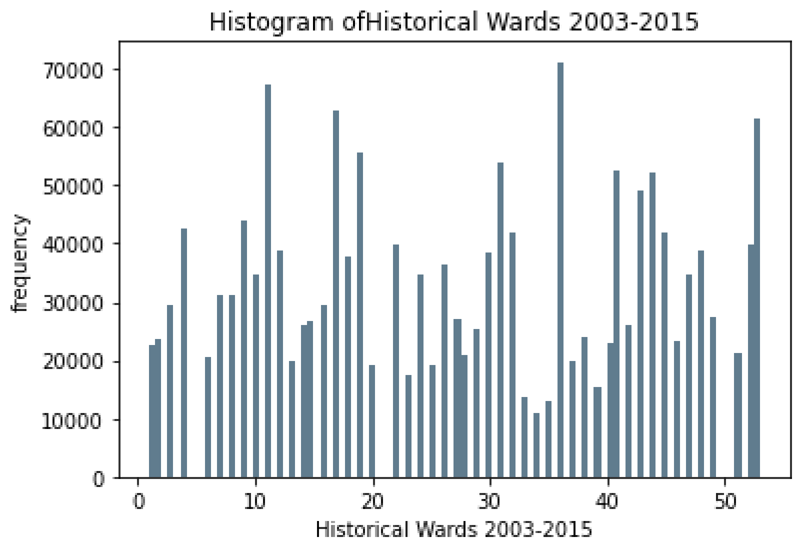
与原数据比较代码及结果为：

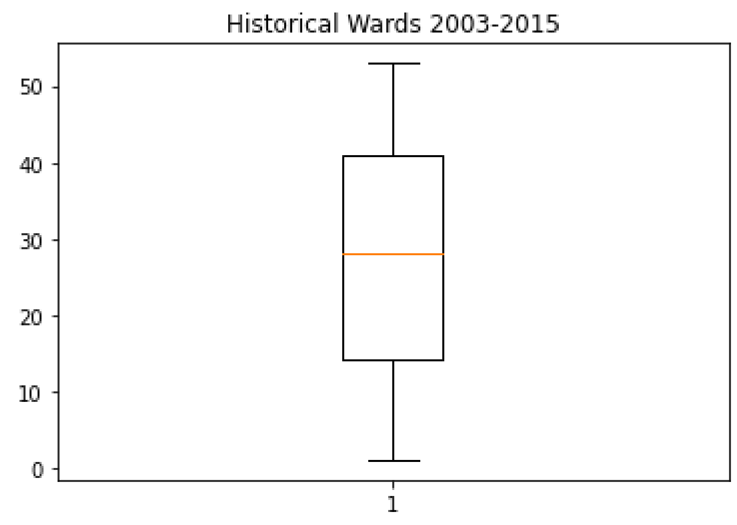
```
att = 'Wards'
subplot_show(data, att, data_lost4_1, att, 100, 'raw data', 'processed data')
```



对数据处理之后Historical Wards 2003-2015的直方图与盒图的代码及结果为：

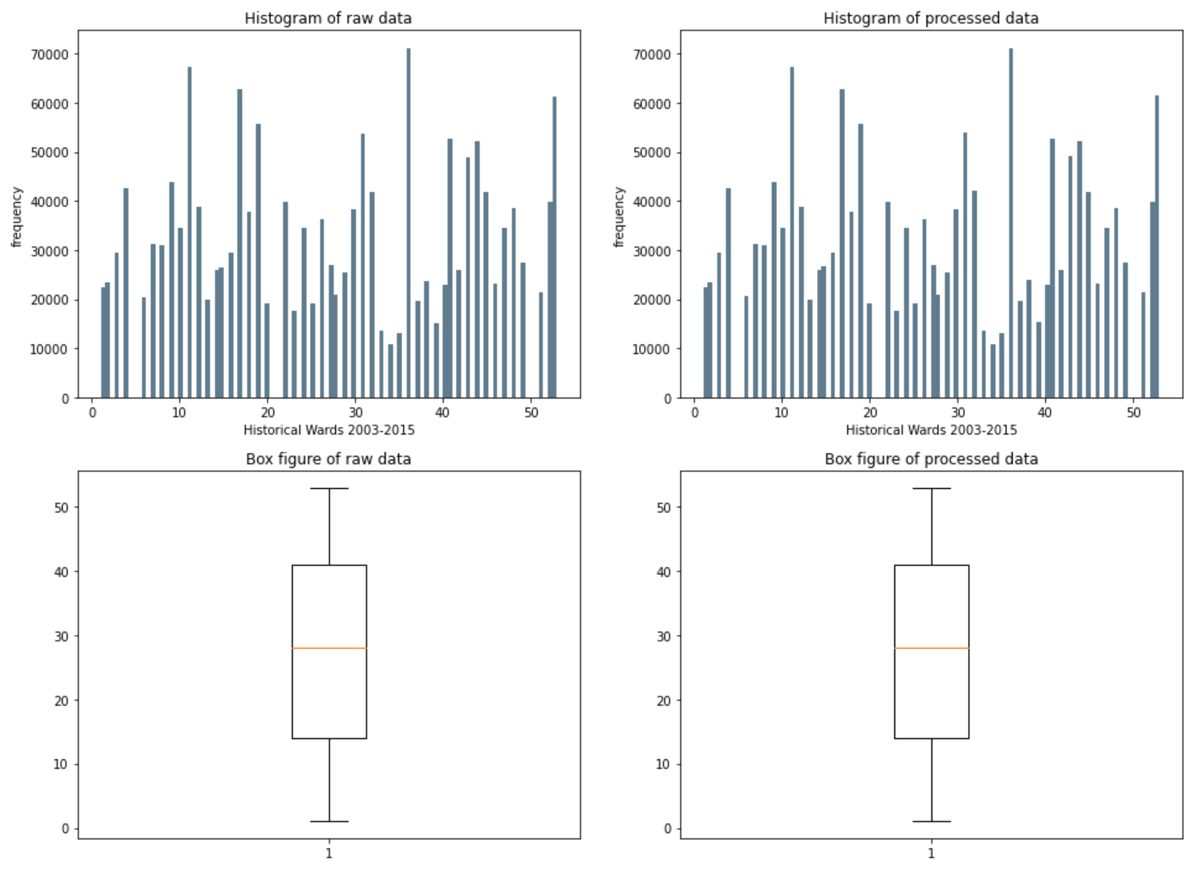
```
att = ['Historical Wards 2003-2015']
show_hist(data_lost4_1, att, 100)
show_boxplot(data_lost4_1,att)
```





与原数据比较代码及结果为：

```
att = 'Historical Wards 2003-2015'
subplot_show(data, att, data_lost4_1, att, 100, 'raw data', 'processed data')
```



注意：所有的代码，统计的词频文档以及清理之后的数据文件均可以在https://github.com/XiaomengFanmcislab/DataMining_1 进行查看。