数据挖掘作业 1 数据探索性分析与预处理

姓名：韩林洁　　学号：2620170052　　日期：2018-4-15

数据分析要求

1、数据可视化和摘要

数据摘要

对标称属性，给出每个可能取值的频数，

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。

2、数据的可视化

针对数值属性，

绘制直方图，用 qq 图检验其分布是否为正态分布。

绘制盒图，对离群值进行识别

3、数据缺失的处理

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

用最高频率值来填补缺失值

通过属性的相关关系来填补缺失值

通过数据对象之间的相似性来填补缺失值

处理后，可视化地对比新旧数据集。

解答内容

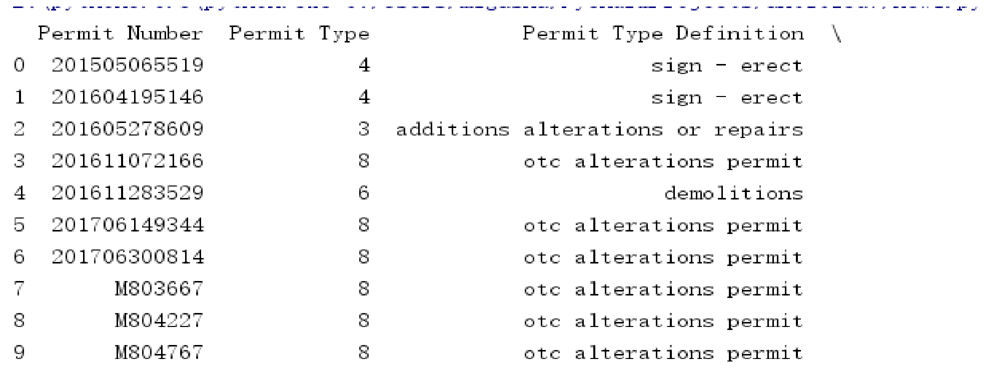
用 python，用到的库有：

```
import operator
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from pandas import Series,DataFrame
import matplotlib
```

一、对于 Building_Permits 数据集（new1.py）：

Step1 读取数据

```
db=' Building_Permits.csv'
#读取 csv 文件，生成 data frame
data=pd.read_csv(db,low_memory=False)
# 定义两类数据：标称型和数值型
frame1=DataFrame(data,columns=['Date'])
frame2=DataFrame(data,columns=['Number of Existing Stories'])
#查看前十条数据内容（截图未显示全部数据）
print(data.iloc[:10])
```

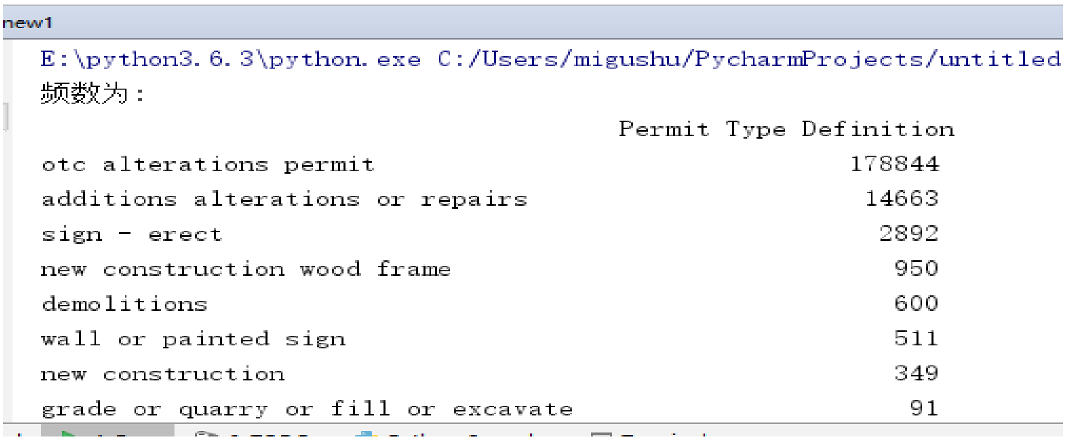


```
  Permit Number  Permit Type            Permit Type Definition  \
0  201505065519            4                       sign - erect
1  201604195146            4                       sign - erect
2  201605278609            3  additions alterations or repairs
3  201611072166            8            otc alterations permit
4  201611283529            6                        demolitions
5  201706149344            8            otc alterations permit
6  201706300814            8            otc alterations permit
7       M803667            8            otc alterations permit
8       M804227            8            otc alterations permit
9       M804767            8            otc alterations permit
```

Step 2 数据摘要

对标称属性，给出每个可能取值的频数

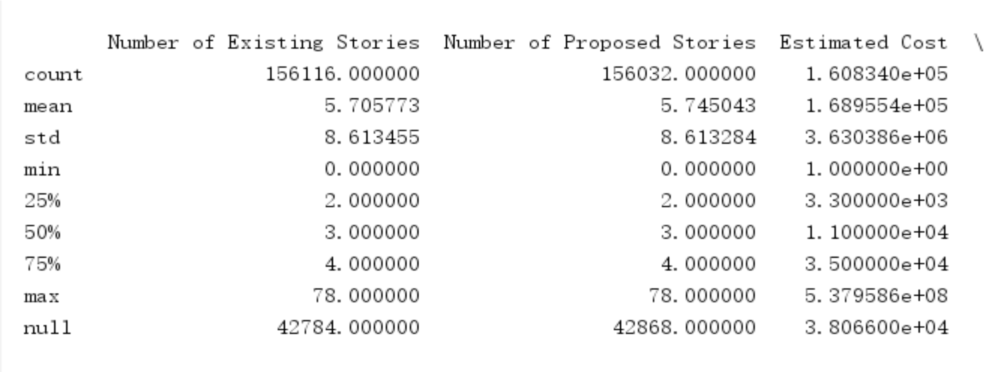
```
for i in range(35):
    print('频数为:\n',frame1.iloc[:,[i]].apply(pd.value_counts),'\n')
```



```
E:\python3.6.3\python.exe C:/Users/migushu/PycharmProjects/untitled
频数为：
                                            Permit Type Definition
otc alterations permit                                      178844
additions alterations or repairs                             14663
sign - erect                                                  2892
new construction wood frame                                    950
demolitions                                                    600
wall or painted sign                                           511
new construction                                               349
grade or quarry or fill or excavate                             91
```

针对数值属性，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，用 describe（）函数获取最大、最小、均值、中位数、四分位数

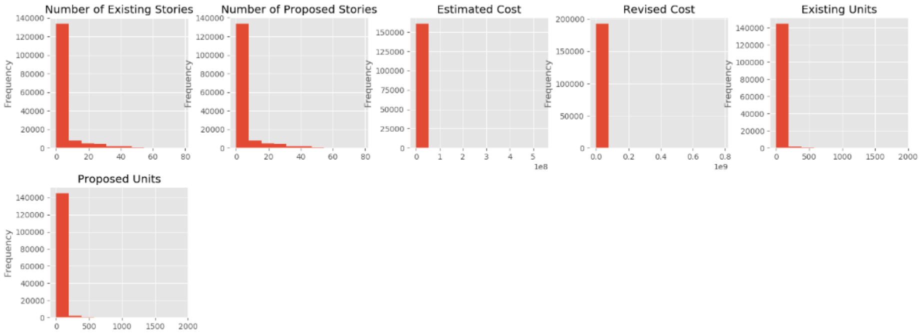
```
statistics=frame2.describe()
statistics.loc['null']=data.shape[0]-statistics.loc['count']
```



```
       Number of Existing Stories  Number of Proposed Stories  Estimated Cost  \
count              156116.000000               156032.000000    1.608340e+05
mean                    5.705773                    5.745043    1.689554e+05
std                     8.613455                    8.613284    3.630386e+06
min                     0.000000                    0.000000    1.000000e+00
25%                     2.000000                    2.000000    3.300000e+03
50%                     3.000000                    3.000000    1.100000e+04
75%                     4.000000                    4.000000    3.500000e+04
max                    78.000000                   78.000000    5.379586e+08
null                42784.000000                42868.000000    3.806600e+04
```

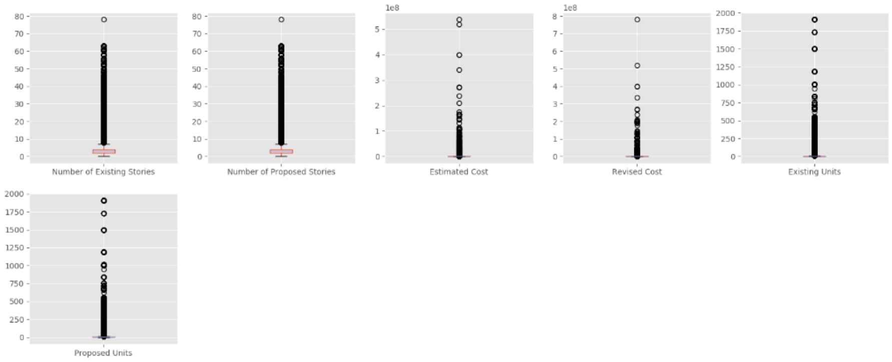
针对数值属性，绘制直方图 ./image1/histogram.png

```
fig = plt.figure(figsize = (20,11))
i = 1
for item in frame2:
    ax = fig.add_subplot(3, 5, i)
    data[item].plot(kind = 'hist', title = item, ax = ax)
    i += 1
plt.subplots_adjust(wspace = 0.3, hspace = 0.3)
```



针对数值属性，绘制盒图，对离群值进行识别 ./image1/boxplot.png

```
fig = plt.figure(figsize = (20,12))
i = 1
for item in frame2:
    ax = fig.add_subplot(3, 5, i)
    data[item].plot(kind = 'box')
    i += 1
```



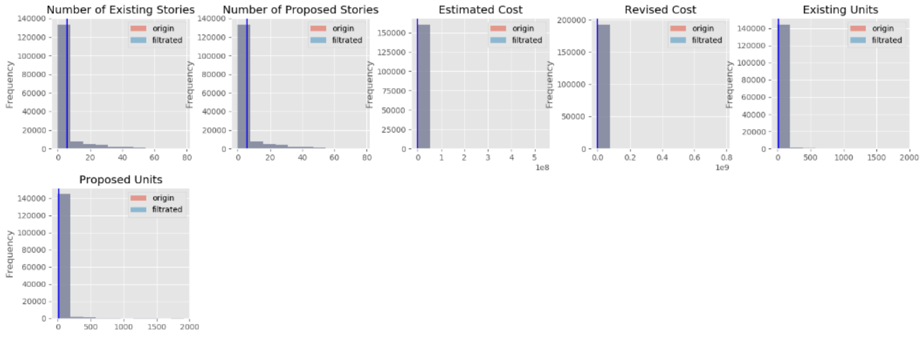
Step 3 处理缺失值

1. 将缺失部分剔除

```
DataTable_filtrated[i] = DataTable_filtrated[i].dropna()
```

绘制可视化图 ./image1/missing_data_delete.png

```
for i in frame2:
    ax = fig.add_subplot(3, 5, n)
    DataTable_filtrated[i] = DataTable_filtrated[i].dropna()    # 删除
    ax.set_title(i)
    data[i].plot(ax=ax, alpha=0.5, kind='hist', label='origin', legend=True)
    DataTable_filtrated[i].plot(ax=ax, alpha=0.5, kind='hist', label='filtrated',
legend=True)
    # pyplot.show()
    ax.axvline(data[i].mean(), color='r')
    ax.axvline(DataTable_filtrated[i].mean(), color='b')
    n += 1
plt.subplots_adjust(wspace=0.3, hspace=0.3)
```

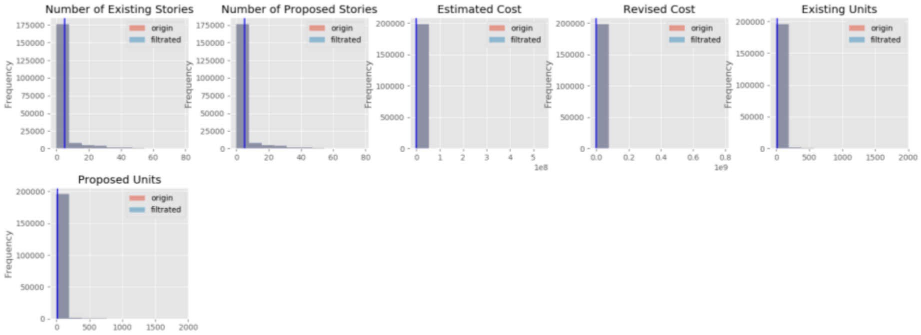


2. 用最高频率值来填补缺失值

```
MostFrequentElement = data[i].value_counts().idxmax();
DataTable_filtrated[i] = DataTable_filtrated[i].fillna(value=MostFrequentElement);    # 众数填补
```

绘制可视化图 ./image1/missing_data_most.png

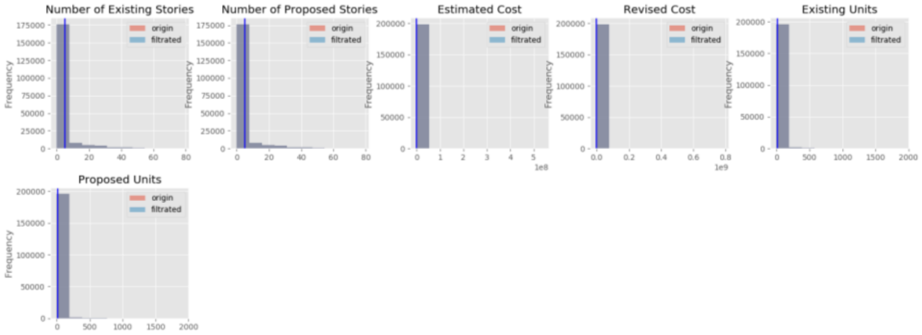
```
for i in frame2:
    ax = fig.add_subplot(4, 5, n)
    MostFrequentElement = data[i].value_counts().idxmax();
    DataTable_filtrated[i] = DataTable_filtrated[i].fillna(value=MostFrequentElement);    # 众数填补缺失值
    ax.set_title(i)
    data[i].plot(ax=ax, alpha=0.5, kind='hist', label='origin', legend=True)
    DataTable_filtrated[i].plot(ax=ax, alpha=0.5, kind='hist', label='filtrated', legend=True)
    # pyplot.show()
    ax.axvline(data[i].mean(), color='r')
    ax.axvline(DataTable_filtrated[i].mean(), color='b')
    n += 1
plt.subplots_adjust(wspace=0.3, hspace=0.3)
```



3. 通过属性的相关关系来填补缺失值，用插值法

绘制可视化图 ./image1/missing_data_corelation.png

```
for i in frame2:
    ax = fig.add_subplot(4, 5, n)
    DataTable_filtrated[i].interpolate(inplace=True)   # 插值
    ax.set_title(i)
    data[i].plot(ax=ax, alpha=0.5, kind='hist', label='origin', legend=True)
    DataTable_filtrated[i].plot(ax=ax, alpha=0.5, kind='hist', label='filtrated', 
legend=True)
    ax.axvline(data[i].mean(), color='r')
    ax.axvline(DataTable_filtrated[i].mean(), color='b')
    n += 1
plt.subplots_adjust(wspace=0.3, hspace=0.3)
```

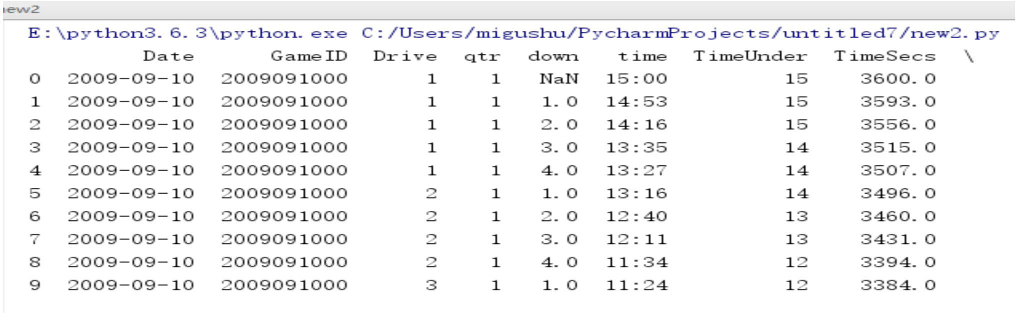


二、对于 NFL Play by Play 2009-2017 (v4)数据集（new2.py）：

Step1 读取数据

```
db='Play.csv'
#读取 csv 文件，生成 data frame
data=pd.read_csv(db,low_memory=False)
```

```
# 定义两类数据：标称型和数值型
frame1=DataFrame(data,columns=['Date','GameID','Drive','qtr','down','time'])
frame2=DataFrame(data,columns=['TimeSecs','PlayTimeDiff','yrdln'])
#查看前十条数据内容
print(data.iloc[:10])
```

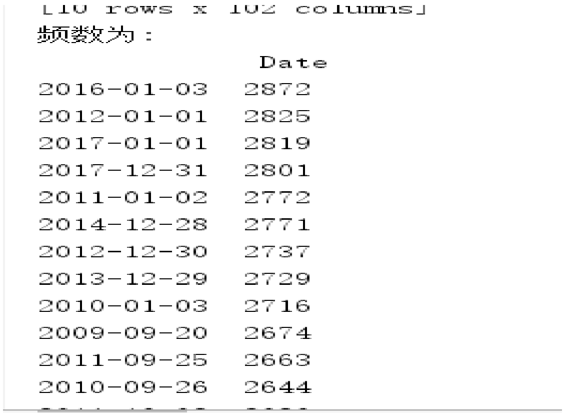


```
iew2
 E:\python3.6.3\python.exe C:/Users/migushu/PycharmProjects/untitled7/new2.py
          Date      GameID  Drive  qtr  down   time  TimeUnder  TimeSecs  \
0  2009-09-10  2009091000      1    1   NaN  15:00         15    3600.0
1  2009-09-10  2009091000      1    1   1.0  14:53         15    3593.0
2  2009-09-10  2009091000      1    1   2.0  14:16         15    3556.0
3  2009-09-10  2009091000      1    1   3.0  13:35         14    3515.0
4  2009-09-10  2009091000      1    1   4.0  13:27         14    3507.0
5  2009-09-10  2009091000      2    1   1.0  13:16         14    3496.0
6  2009-09-10  2009091000      2    1   2.0  12:40         13    3460.0
7  2009-09-10  2009091000      2    1   3.0  12:11         13    3431.0
8  2009-09-10  2009091000      2    1   4.0  11:34         12    3394.0
9  2009-09-10  2009091000      3    1   1.0  11:24         12    3384.0
```

Step 2 数据摘要

对标称属性，给出每个可能取值的频数

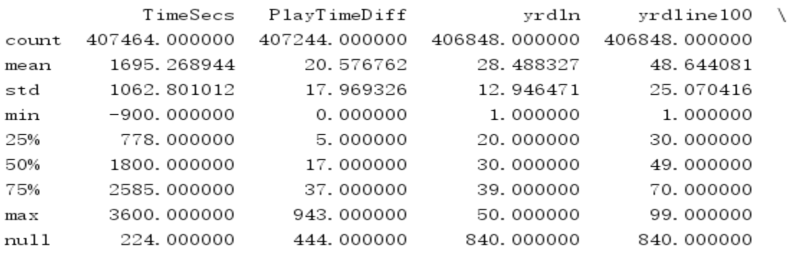
```
for i in range(35):
    print('频数为:\n',frame1.iloc[:,[i]].apply(pd.value_counts),'\n')
```



```
[10 rows x 102 columns]
频数为：
             Date
2016-01-03  2872
2012-01-01  2825
2017-01-01  2819
2017-12-31  2801
2011-01-02  2772
2014-12-28  2771
2012-12-30  2737
2013-12-29  2729
2010-01-03  2716
2009-09-20  2674
2011-09-25  2663
2010-09-26  2644
```

针对数值属性，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，用 describe（）函数获取最大、最小、均值、中位数、四分位数

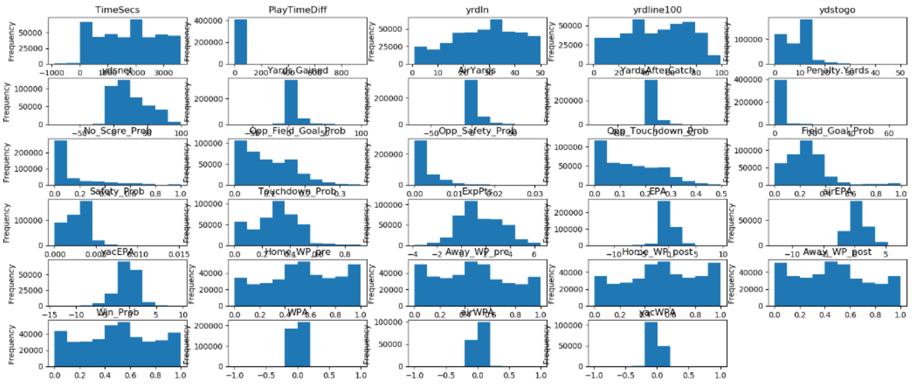
```
statistics=frame2.describe()
statistics.loc['null']=data.shape[0]-statistics.loc['count']
```



```
            TimeSecs   PlayTimeDiff          yrdln     yrdline100  \
count  407464.000000  407244.000000  406848.000000  406848.000000
mean     1695.268944      20.576762      28.488327      48.644081
std      1062.801012      17.969326      12.946471      25.070416
min      -900.000000       0.000000       1.000000       1.000000
25%       778.000000       5.000000      20.000000      30.000000
50%      1800.000000      17.000000      30.000000      49.000000
75%      2585.000000      37.000000      39.000000      70.000000
max      3600.000000     943.000000      50.000000      99.000000
null      224.000000     444.000000     840.000000     840.000000
```

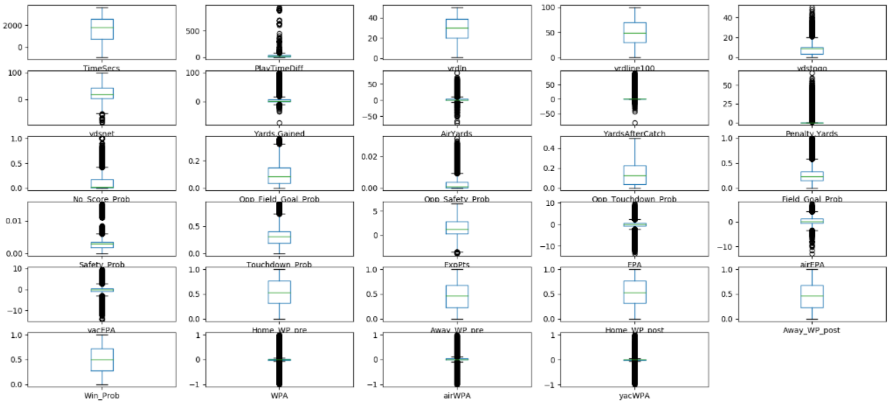
针对数值属性，绘制直方图 ./image2/histogram.png

```
fig = plt.figure(figsize = (20,11))
i = 1
for item in frame2:
    ax = fig.add_subplot(8, 5, i)
    data[item].plot(kind = 'hist', title = item, ax = ax)
    i += 1
plt.subplots_adjust(wspace = 0.3, hspace = 0.3)
```



针对数值属性，绘制盒图，对离群值进行识别 ./image2/boxplot.png

```
fig = plt.figure(figsize = (20,12))
i = 1
for item in frame2:
    ax = fig.add_subplot(3, 5, i)
    data[item].plot(kind = 'box')
    i += 1
```



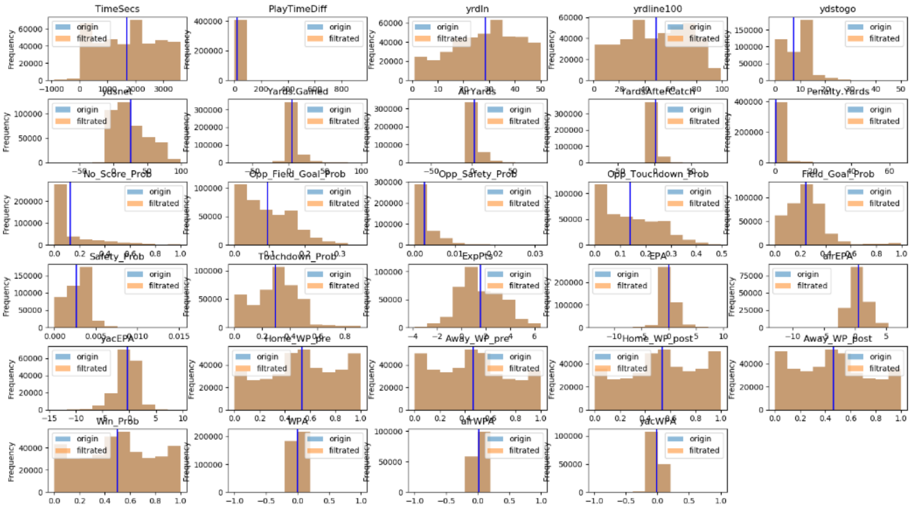
Step 3 处理缺失值

4. 将缺失部分剔除

```
DataTable_filtrated[i] = DataTable_filtrated[i].dropna()
```

绘制可视化图 ./image2/missing_data_delete.png

```
for i in frame2:
    ax = fig.add_subplot(8, 5, n)
    DataTable_filtrated[i] = DataTable_filtrated[i].dropna()   # 删除
    ax.set_title(i)
    data[i].plot(ax=ax, alpha=0.5, kind='hist', label='origin', legend=True)
    DataTable_filtrated[i].plot(ax=ax, alpha=0.5, kind='hist', label='filtrated', legend=True)
    # pyplot.show()
    ax.axvline(data[i].mean(), color='r')
    ax.axvline(DataTable_filtrated[i].mean(), color='b')
    n += 1
plt.subplots_adjust(wspace=0.3, hspace=0.3)
```

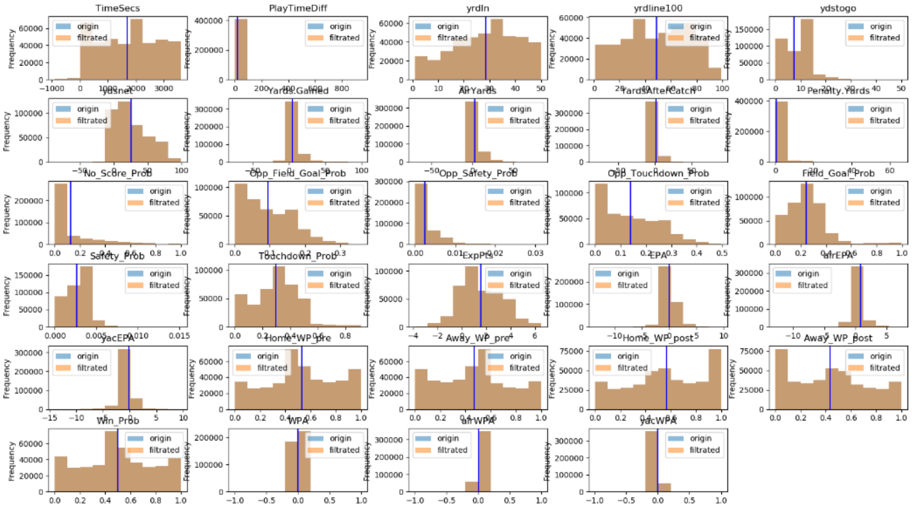


5. 用最高频率值来填补缺失值

```
MostFrequentElement = data[i].value_counts().idxmax();
DataTable_filtrated[i] = DataTable_filtrated[i].fillna(value=MostFrequentElement);   # 众数填补
```

绘制可视化图 ./image2/missing_data_most.png

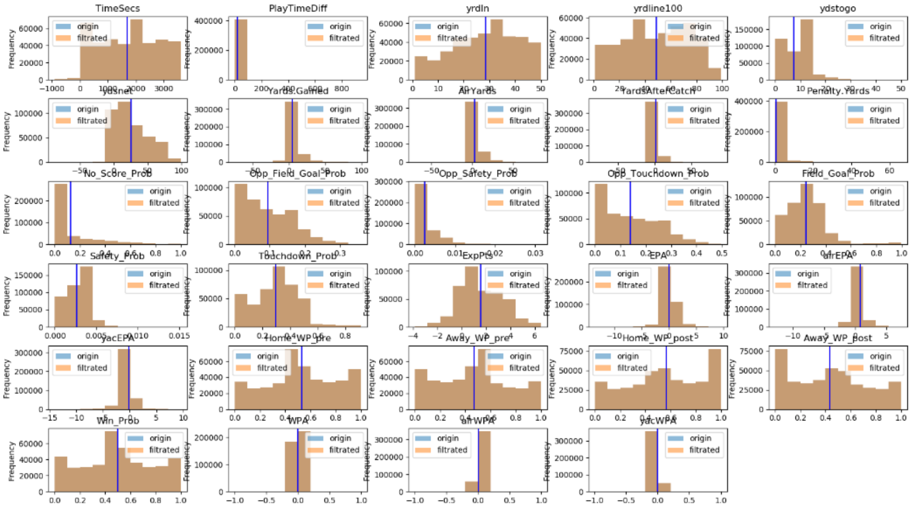
```
for i in frame2:
    ax = fig.add_subplot(8, 5, n)
    MostFrequentElement = data[i].value_counts().idxmax();
    DataTable_filtrated[i] = DataTable_filtrated[i].fillna(value=MostFrequentElement);   # 众数填补缺失值
    ax.set_title(i)
    data[i].plot(ax=ax, alpha=0.5, kind='hist', label='origin', legend=True)
    DataTable_filtrated[i].plot(ax=ax, alpha=0.5, kind='hist', label='filtrated', legend=True)
    # pyplot.show()
    ax.axvline(data[i].mean(), color='r')
    ax.axvline(DataTable_filtrated[i].mean(), color='b')
    n += 1
plt.subplots_adjust(wspace=0.3, hspace=0.3)
```



6. 通过属性的相关关系来填补缺失值，用插值法

绘制可视化图 ./image2/missing_data_corelation.png

```
for i in frame2:
    ax = fig.add_subplot(8, 5, n)
    DataTable_filtrated[i].interpolate(inplace=True)   # 插值
    ax.set_title(i)
    data[i].plot(ax=ax, alpha=0.5, kind='hist', label='origin', legend=True)
    DataTable_filtrated[i].plot(ax=ax, alpha=0.5, kind='hist', label='filtrated',
legend=True)
    ax.axvline(data[i].mean(), color='r')
    ax.axvline(DataTable_filtrated[i].mean(), color='b')
    n += 1
plt.subplots_adjust(wspace=0.3, hspace=0.3)
```


TimeSecs
PlayTimeDiff
yrdln
yrdline100
ydstogo
ydsnet
Yards.Gained
AirYards
YardsAfterCatch
Penalty.Yards
No_Score_Prob
Opp_Field_Goal_Prob
Opp_Safety_Prob
Opp_Touchdown_Prob
Field_Goal_Prob
Safety_Prob
Touchdown_Prob
ExpPts
EPA
airEPA
yacEPA
Home_WP_pre
Away_WP_pre
Home_WP_post
Away_WP_post
Win_Prob
WPA
airWPA
yacWPA
origin
filtrated
Frequency