In [1]:

```
import pandas as pd
import numpy as np
import seaborn as sns
import sklearn
```

In [2]:

```
training= pd.read_csv('C:/Users/DELL/Desktop/sc/training_xjt.csv')
```

In [3]:

```
# extract the data start with d and h and id info
# train_id=training.filter(regex='id$', axis=1)
# train_d=training.filter(regex='^d1', axis=1)
# train_h=training.filter(regex='^h1',axis=1)
# resultd = pd.concat([train_id, train_d], axis=1, sort=False)
# resulth = pd.concat([train_id, train_h], axis=1, sort=False)
# resultdh = pd.concat([resultd, resulth], axis=1, sort=False)
```

Albumin

In [4]:

```
# variable: albumin
dfalbumin=training.filter(regex='albumin')

# check day values missing while hour values not
print(len(dfalbumin[(dfalbumin.h1_albumin_max.notna()) & (dfalbumin.d1_albumin_max.isna())]),
      len(dfalbumin[(dfalbumin.h1_albumin_min.notna()) & (dfalbumin.d1_albumin_min.isna())]))
```

0 0

In [5]:

```
# Check if missing values are in pairs for max and min
print(len(dfalbumin[(dfalbumin.d1_albumin_max.notna()) & (dfalbumin.d1_albumin_min.isna())]),
      len(dfalbumin[(dfalbumin.d1_albumin_max.isna()) & (dfalbumin.d1_albumin_min.notna())]))

#len(dfalbumin[(dfalbumin.albumin_apache.notna()) & (dfalbumin.d1_albumin_max.isna())])
#len(dfalbumin[dfalbumin.d1_albumin_max.notna()])
```

0 0

In [6]:

```
# replace apache covariate with null values in d1 albumin
dfalbumin.loc[dfalbumin['d1_albumin_max'].isnull(),'d1_albumin_max']=dfalbumin['albumin_apache']
dfalbumin.loc[dfalbumin['d1_albumin_min'].isnull(),'d1_albumin_min']=dfalbumin['albumin_apache']
```

```
C:\Application\Python\Anaconda3\lib\site-packages\pandas\core\indexing.py:189: Set
tingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy
  self._setitem_with_indexer(indexer, value)
C:\Application\Python\Anaconda3\lib\site-packages\ipykernel_launcher.py:2: Setting
WithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy

C:\Application\Python\Anaconda3\lib\site-packages\ipykernel_launcher.py:3: Setting
WithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy
  This is separate from the ipykernel package so we can avoid doing imports until
```

In [7]:

```
# list unique values in d1_max data to check for abnoramal data
print(sorted(dfalbumin.d1_albumin_max.unique()),
      sorted(dfalbumin.d1_albumin_min.unique()))
```

```
[1.2, 1.4, 1.5, 1.6, 1.7, 2.3, nan, 1.3, 1.8, 1.9, 2.0, 2.1, 2.2, 2.4, 2.5, 2.6,
2.7, 2.8, 2.9, 3.0, 3.1, 3.2, 3.3, 3.4, 3.5, 3.6, 3.7, 3.8, 3.9, 4.0, 4.1, 4.2, 4.
3, 4.4, 4.5, 4.6] [1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 2.3, nan, 1.8, 1.9, 2.0, 2.
1, 2.2, 2.4, 2.5, 2.6, 2.7, 2.8, 2.9, 3.0, 3.1, 3.2, 3.3, 3.4, 3.5, 3.6, 3.7, 3.8,
3.9, 4.0, 4.1, 4.2, 4.3, 4.4, 4.5]
```

In [8]:

```
# data cleaning for min > max situation
dfalbumin[dfalbumin['d1_albumin_max']<dfalbumin['d1_albumin_min']]

# where h1_albumin_max is taken as it is reasonable
dfalbumin.loc[(dfalbumin.d1_albumin_max<dfalbumin.d1_albumin_min) & (dfalbumin.h1_albumin_max.notnull()),'d1_albumin_max']=dfalbumin['h1_albumin_max']

# where apache covariate is taken as it is reasonable
dfalbumin.loc[(dfalbumin.d1_albumin_max<dfalbumin.d1_albumin_min) & (dfalbumin.albumin_apache>=dfalbumin.d1_albumin_min),'d1_albumin_max']=dfalbumin['albumin_apache']
```

```
C:\Application\Python\Anaconda3\lib\site-packages\pandas\core\indexing.py:189: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy
  self._setitem_with_indexer(indexer, value)
C:\Application\Python\Anaconda3\lib\site-packages\ipykernel_launcher.py:5: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy
  """
C:\Application\Python\Anaconda3\lib\site-packages\ipykernel_launcher.py:8: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy
```

In [9]:

```
# look into the data to find reasonable imputation method
backup=dfalbumin[dfalbumin['d1_albumin_max']<dfalbumin['d1_albumin_min']]
print(sorted(backup.d1_albumin_max.unique()),
      sorted(backup.d1_albumin_min.unique()))

albuminimputation=dfalbumin[(dfalbumin['d1_albumin_max']!=1.2) &
                            (dfalbumin['d1_albumin_min']<=4.5) &
                            (dfalbumin['d1_albumin_min']>=3)]

albuminimputation.groupby('d1_albumin_min')['d1_albumin_max'].value_counts()
```

[1.2] [3.0, 3.1, 3.6, 3.8, 3.9, 4.2, 4.3, 4.4, 4.5]

Out[9]:

d1_albumin_min	d1_albumin_max	
3.0	3.0	1965
	3.1	102
	3.2	83
	3.3	63
	3.4	54
	3.5	32
	3.6	28
	3.7	22
	3.8	12
	4.0	9
	3.9	8
	4.1	4
	4.5	3
	4.2	1
3.1	3.1	1979
	3.3	93
	3.2	87
	3.4	64
	3.5	61
	3.6	54
	3.7	31
	3.8	12
	3.9	12
	4.0	7
	4.1	3
	4.3	3
	4.5	2
	4.2	1
	4.4	1
3.2	3.2	1992
		...
3.9	4.6	8
	4.4	5
	4.5	5
4.0	4.0	681
	4.2	18
	4.1	14
	4.6	9
	4.4	8
	4.3	7
	4.5	5
4.1	4.1	467
	4.2	11
	4.3	9
	4.4	6
	4.5	4
	4.6	2
4.2	4.2	325
	4.4	11
	4.3	9
	4.6	7
	4.5	2
4.3	4.3	237
	4.5	6
	4.4	4
	4.6	2
4.4	4.4	148
	4.6	4

```
                4.5                   2
4.5             4.6                 125
                4.5                   1
Name: d1_albumin_max, Length: 145, dtype: int64
```

In [10]:

```
# Due to too many values for each d1 min values, impute the mean/median of each group
for i in (sorted(backup.d1_albumin_min.unique())):
    imputation=albuminimputation.loc[albuminimputation['d1_albumin_min'] == i, 'd1_albumin_max']
.mean()
    dfalbumin.loc[(dfalbumin.d1_albumin_max<dfalbumin.d1_albumin_min) &
                  (dfalbumin.d1_albumin_min==i),'d1_albumin_max']=round(imputation,1)

# imputation check
# dfalbumin = dfalbumin[dfalbumin.index.isin(backup.index)]
# dfalbumin

print(len(dfalbumin[dfalbumin['d1_albumin_max']<dfalbumin['d1_albumin_min']]))
```

```
C:\Application\Python\Anaconda3\lib\site-packages\pandas\core\indexing.py:189: Set
tingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy
  self._setitem_with_indexer(indexer, value)
C:\Application\Python\Anaconda3\lib\site-packages\ipykernel_launcher.py:5: Setting
WithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy
  """

0
```

In [11]:

```
# data cleaning for value 0 situation
dfalbumin[dfalbumin.d1_albumin_max==0]
dfalbumin[dfalbumin.d1_albumin_min==0]

# check all null after finishing data cleaning for albumin
dfalbumin[(dfalbumin.d1_albumin_max.isnull())& (dfalbumin.h1_albumin_max.isnull())&
          (dfalbumin.d1_albumin_min.isnull())& (dfalbumin.h1_albumin_min.isnull())&
          (dfalbumin.albumin_apache.isnull())]
```

Out[11]:

	albumin_apache	d1_albumin_max	d1_albumin_min	h1_albumin_max	h1_alb
2	NaN	NaN	NaN	NaN	NaN
3	NaN	NaN	NaN	NaN	NaN
4	NaN	NaN	NaN	NaN	NaN
5	NaN	NaN	NaN	NaN	NaN
6	NaN	NaN	NaN	NaN	NaN
7	NaN	NaN	NaN	NaN	NaN
10	NaN	NaN	NaN	NaN	NaN
11	NaN	NaN	NaN	NaN	NaN
12	NaN	NaN	NaN	NaN	NaN
14	NaN	NaN	NaN	NaN	NaN
15	NaN	NaN	NaN	NaN	NaN
16	NaN	NaN	NaN	NaN	NaN
17	NaN	NaN	NaN	NaN	NaN
18	NaN	NaN	NaN	NaN	NaN
19	NaN	NaN	NaN	NaN	NaN
20	NaN	NaN	NaN	NaN	NaN
21	NaN	NaN	NaN	NaN	NaN
22	NaN	NaN	NaN	NaN	NaN
24	NaN	NaN	NaN	NaN	NaN
26	NaN	NaN	NaN	NaN	NaN
28	NaN	NaN	NaN	NaN	NaN
29	NaN	NaN	NaN	NaN	NaN
30	NaN	NaN	NaN	NaN	NaN
32	NaN	NaN	NaN	NaN	NaN
34	NaN	NaN	NaN	NaN	NaN
36	NaN	NaN	NaN	NaN	NaN
37	NaN	NaN	NaN	NaN	NaN
38	NaN	NaN	NaN	NaN	NaN
39	NaN	NaN	NaN	NaN	NaN
41	NaN	NaN	NaN	NaN	NaN
...	...	...	...	...	...
87401	NaN	NaN	NaN	NaN	NaN
87402	NaN	NaN	NaN	NaN	NaN

	albumin_apache	d1_albumin_max	d1_albumin_min	h1_albumin_max	h1_alb
87406	NaN	NaN	NaN	NaN	NaN
87407	NaN	NaN	NaN	NaN	NaN
87410	NaN	NaN	NaN	NaN	NaN
87417	NaN	NaN	NaN	NaN	NaN
87418	NaN	NaN	NaN	NaN	NaN
87419	NaN	NaN	NaN	NaN	NaN
87420	NaN	NaN	NaN	NaN	NaN
87422	NaN	NaN	NaN	NaN	NaN
87425	NaN	NaN	NaN	NaN	NaN
87427	NaN	NaN	NaN	NaN	NaN
87429	NaN	NaN	NaN	NaN	NaN
87431	NaN	NaN	NaN	NaN	NaN
87436	NaN	NaN	NaN	NaN	NaN
87439	NaN	NaN	NaN	NaN	NaN
87442	NaN	NaN	NaN	NaN	NaN
87443	NaN	NaN	NaN	NaN	NaN
87449	NaN	NaN	NaN	NaN	NaN
87451	NaN	NaN	NaN	NaN	NaN
87452	NaN	NaN	NaN	NaN	NaN
87453	NaN	NaN	NaN	NaN	NaN
87459	NaN	NaN	NaN	NaN	NaN
87460	NaN	NaN	NaN	NaN	NaN
87462	NaN	NaN	NaN	NaN	NaN
87471	NaN	NaN	NaN	NaN	NaN
87474	NaN	NaN	NaN	NaN	NaN
87475	NaN	NaN	NaN	NaN	NaN
87481	NaN	NaN	NaN	NaN	NaN
87482	NaN	NaN	NaN	NaN	NaN

45887 rows × 5 columns

bilirubin

In [12]:

```
# variable: bilirubin
dfbilirubin=training.filter(regex='bilirubin')

# # check day values missing while hour values not
print(len(dfbilirubin[(dfbilirubin.h1_bilirubin_max.notna()) & (dfbilirubin.d1_bilirubin_max.isna())]),
      len(dfbilirubin[(dfbilirubin.h1_bilirubin_min.notna()) & (dfbilirubin.d1_bilirubin_min.isna())]))
```

```
0 0
```

In [13]:

```
# Check if missing values are in pairs for max and min
print(len(dfbilirubin[(dfbilirubin.d1_bilirubin_max.notna()) & (dfbilirubin.d1_bilirubin_min.isna())]),
      len(dfbilirubin[(dfbilirubin.d1_bilirubin_max.isna()) & (dfbilirubin.d1_bilirubin_min.notna())]))

#len(dfbilirubin[(dfbilirubin.bilirubin_apache_apache.notna()) & (dfbilirubin.d1_bilirubin_max.isna())])
#len(dfbilirubin[dfbilirubin.d1_bilirubin_max.notna()])
```

```
0 0
```

In [14]:

```
# replace apache covariate with null values in d1 albumin
dfbilirubin.loc[dfbilirubin['d1_bilirubin_max'].isnull(),'d1_bilirubin_max']=dfbilirubin['bilirubin_apache']
dfbilirubin.loc[dfbilirubin['d1_bilirubin_min'].isnull(),'d1_bilirubin_min']=dfbilirubin['bilirubin_apache']
```

```
C:\Application\Python\Anaconda3\lib\site-packages\pandas\core\indexing.py:189: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy
  self._setitem_with_indexer(indexer, value)
C:\Application\Python\Anaconda3\lib\site-packages\ipykernel_launcher.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy

C:\Application\Python\Anaconda3\lib\site-packages\ipykernel_launcher.py:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy
  This is separate from the ipykernel package so we can avoid doing imports until
```

In [15]:

```
# list unique values in d1_max data to check for abnoramal data
# print(sorted(dfbilirubin.d1_bilirubin_max.unique()),
#       sorted(dfbilirubin.d1_bilirubin_min.unique()))
```

In [16]:

```
# data cleaning for min > max situation
dfbilirubin[dfbilirubin.d1_bilirubin_max<dfbilirubin.d1_bilirubin_min]

# data cleaning for value 0 situation
dfbilirubin[dfbilirubin.d1_bilirubin_max==0]
dfbilirubin[dfbilirubin.d1_bilirubin_min==0]

# check all null after finishing data cleaning for bilirubin
dfbilirubin[(dfbilirubin.d1_bilirubin_max.isnull())& (dfbilirubin.h1_bilirubin_max.isnull())&
          (dfbilirubin.d1_bilirubin_min.isnull())& (dfbilirubin.h1_bilirubin_min.isnull())&
          (dfbilirubin.bilirubin_apache.isnull())]
```

Out[16]:

	bilirubin_apache	d1_bilirubin_max	d1_bilirubin_min	h1_bilirubin_max	h1_bi
2	NaN	NaN	NaN	NaN	NaN
3	NaN	NaN	NaN	NaN	NaN
4	NaN	NaN	NaN	NaN	NaN
5	NaN	NaN	NaN	NaN	NaN
6	NaN	NaN	NaN	NaN	NaN
7	NaN	NaN	NaN	NaN	NaN
10	NaN	NaN	NaN	NaN	NaN
11	NaN	NaN	NaN	NaN	NaN
12	NaN	NaN	NaN	NaN	NaN
14	NaN	NaN	NaN	NaN	NaN
15	NaN	NaN	NaN	NaN	NaN
16	NaN	NaN	NaN	NaN	NaN
17	NaN	NaN	NaN	NaN	NaN
18	NaN	NaN	NaN	NaN	NaN
19	NaN	NaN	NaN	NaN	NaN
20	NaN	NaN	NaN	NaN	NaN
21	NaN	NaN	NaN	NaN	NaN
22	NaN	NaN	NaN	NaN	NaN
24	NaN	NaN	NaN	NaN	NaN
26	NaN	NaN	NaN	NaN	NaN
28	NaN	NaN	NaN	NaN	NaN
29	NaN	NaN	NaN	NaN	NaN
30	NaN	NaN	NaN	NaN	NaN
32	NaN	NaN	NaN	NaN	NaN
34	NaN	NaN	NaN	NaN	NaN
36	NaN	NaN	NaN	NaN	NaN
37	NaN	NaN	NaN	NaN	NaN
38	NaN	NaN	NaN	NaN	NaN
39	NaN	NaN	NaN	NaN	NaN
41	NaN	NaN	NaN	NaN	NaN
...	...	...	...	...	...
87420	NaN	NaN	NaN	NaN	NaN
87422	NaN	NaN	NaN	NaN	NaN

	bilirubin_apache	d1_bilirubin_max	d1_bilirubin_min	h1_bilirubin_max	h1_bi
87425	NaN	NaN	NaN	NaN	NaN
87427	NaN	NaN	NaN	NaN	NaN
87429	NaN	NaN	NaN	NaN	NaN
87431	NaN	NaN	NaN	NaN	NaN
87436	NaN	NaN	NaN	NaN	NaN
87437	NaN	NaN	NaN	NaN	NaN
87439	NaN	NaN	NaN	NaN	NaN
87442	NaN	NaN	NaN	NaN	NaN
87443	NaN	NaN	NaN	NaN	NaN
87444	NaN	NaN	NaN	NaN	NaN
87446	NaN	NaN	NaN	NaN	NaN
87447	NaN	NaN	NaN	NaN	NaN
87449	NaN	NaN	NaN	NaN	NaN
87451	NaN	NaN	NaN	NaN	NaN
87453	NaN	NaN	NaN	NaN	NaN
87459	NaN	NaN	NaN	NaN	NaN
87460	NaN	NaN	NaN	NaN	NaN
87462	NaN	NaN	NaN	NaN	NaN
87464	NaN	NaN	NaN	NaN	NaN
87467	NaN	NaN	NaN	NaN	NaN
87469	NaN	NaN	NaN	NaN	NaN
87471	NaN	NaN	NaN	NaN	NaN
87474	NaN	NaN	NaN	NaN	NaN
87475	NaN	NaN	NaN	NaN	NaN
87479	NaN	NaN	NaN	NaN	NaN
87481	NaN	NaN	NaN	NaN	NaN
87482	NaN	NaN	NaN	NaN	NaN
87483	NaN	NaN	NaN	NaN	NaN

50083 rows × 5 columns

BUN

In [17]:

```
# variable: albumin
dfbun=training.filter(regex='bun')

# check day values missing while hour values not
print(len(dfbun[(dfbun.h1_bun_max.notna()) & (dfbun.d1_bun_max.isna())]),
      len(dfbun[(dfbun.h1_bun_min.notna()) & (dfbun.d1_bun_min.isna())]))
```

0 0

In [18]:

```
# Check if missing values are in pairs for max and min
print(len(dfbun[(dfbun.d1_bun_max.notna()) & (dfbun.d1_bun_min.isna())]),
      len(dfbun[(dfbun.d1_bun_max.isna()) & (dfbun.d1_bun_min.notna())]))

#len(dfbun[(dfbun.bun_apache.notna()) & (dfbun.d1_bun_max.isna())])
#len(dfbun[dfbun.d1_bun_max.notna()])
```

0 0

In [19]:

```
# replace apache covariate with null values in d1 bun
dfbun.loc[dfbun['d1_bun_max'].isnull(),'d1_bun_max']=dfbun['bun_apache']
dfbun.loc[dfbun['d1_bun_min'].isnull(),'d1_bun_min']=dfbun['bun_apache']

# list unique values in d1_max data to check for abnoramal data
print(sorted(dfbun.d1_bun_max.unique()),
      sorted(dfbun.d1_bun_min.unique()))
```

```
C:\Application\Python\Anaconda3\lib\site-packages\pandas\core\indexing.py:189: Set
tingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy
  self._setitem_with_indexer(indexer, value)
```

[4.0, 4.3, 5.0, 5.3, 5.4, 5.6, 5.7, 5.8, 5.9, 6.0, 6.3, 6.6, 6.8, 6.9, 7.0, 7.1,
7.2, 7.6, 8.0, 8.2, 8.3, 8.7, 8.8, 9.0, 9.1, 9.2, 9.6, 9.8, 9.9, 10.0, 10.1, 10.2,
10.3, 10.4, 10.8, 11.0, 11.1, 11.2, 11.4, 11.5, 11.7, 11.9, 12.0, 12.1, 12.2, 12.
4, 12.5, 12.7, 12.8, 13.0, 13.3, 13.4, 13.7, 13.8, 13.9, 14.0, 14.1, 14.6, 14.7, 1
5.0, 15.01, 15.2, 15.4, 15.6, 16.0, 16.2, 16.3, 16.6, 16.7, 16.8, 17.0, 17.1, 17.
5, 17.6, 18.0, 18.5, 18.6, 19.0, 19.2, 19.3, 19.4, 19.6, 19.7, 20.0, 20.6, 20.8, 2
1.0, 21.5, 22.0, 22.1, 22.2, 22.8, 23.0, 23.2, 23.3, 23.4, 23.5, 23.6, 24.0, 24.2,
24.6, 24.9, 25.0, 25.7, 26.1, 26.3, 27.3, 27.4, 29.5, 29.6, 30.8, 30.9, 31.0, nan,
4.1, 4.4, 4.6, 4.7, 4.8, 5.1, 5.5, 6.1, 6.2, 6.4, 6.5, 6.7, 7.3, 7.4, 7.5, 7.7, 7.
8, 7.9, 8.1, 8.4, 8.5, 8.6, 8.9, 9.3, 9.4, 9.5, 9.7, 10.5, 10.6, 10.7, 10.9, 11.3,
11.6, 11.8, 12.3, 12.6, 12.9, 13.1, 13.5, 13.6, 14.2, 14.3, 14.4, 14.5, 14.8, 14.
9, 15.1, 15.3, 15.5, 15.7, 15.8, 15.9, 16.1, 16.4, 16.5, 16.9, 17.2, 17.3, 17.4, 1
7.7, 17.8, 17.9, 18.1, 18.2, 18.3, 18.4, 18.7, 18.8, 18.9, 19.1, 19.5, 19.8, 20.1,
20.2, 20.3, 20.4, 20.5, 20.7, 20.9, 21.2, 21.3, 21.4, 21.7, 21.8, 21.9, 22.3, 22.
4, 22.6, 22.7, 22.9, 23.1, 23.7, 23.9, 24.1, 24.3, 24.4, 24.5, 24.7, 24.8, 25.1, 2
5.2, 25.4, 25.5, 25.6, 25.8, 26.0, 26.2, 26.4, 26.6, 26.7, 26.8, 26.9, 27.0, 27.2,
27.5, 27.8, 28.0, 28.1, 28.2, 28.3, 28.7, 28.8, 28.9, 29.0, 29.1, 29.3, 29.4, 29.
9, 30.0, 30.7, 31.1, 31.3, 31.4, 31.5, 31.6, 31.7, 32.0, 32.2, 32.4, 32.5, 32.6, 3
2.9, 33.0, 33.1, 33.3, 33.4, 33.6, 33.7, 33.8, 33.9, 34.0, 34.1, 34.2, 34.3, 34.4,
34.5, 34.8, 34.9, 35.0, 35.2, 35.3, 35.6, 35.7, 36.0, 36.1, 36.2, 36.3, 36.6, 36.
7, 36.8, 36.9, 37.0, 37.1, 37.2, 38.0, 38.6, 38.7, 39.0, 39.1, 39.2, 40.0, 40.1, 4
0.2, 40.5, 40.7, 41.0, 41.2, 41.3, 41.5, 41.7, 42.0, 42.2, 42.4, 42.5, 42.7, 42.8,
42.9, 43.0, 43.4, 43.7, 43.8, 44.0, 44.1, 44.8, 45.0, 45.5, 45.7, 46.0, 46.9, 47.
0, 47.3, 47.7, 48.0, 48.1, 48.7, 49.0, 49.3, 49.5, 49.7, 50.0, 50.3, 50.8, 50.9, 5
1.0, 51.8, 51.9, 52.0, 52.1, 52.6, 52.7, 53.0, 53.1, 53.2, 53.5, 53.9, 54.0, 54.1,
54.9, 55.0, 55.1, 55.6, 55.7, 55.9, 56.0, 56.2, 56.3, 57.0, 57.1, 57.5, 58.0, 59.
0, 59.3, 59.9, 60.0, 60.5, 60.6, 60.8, 61.0, 61.1, 61.3, 62.0, 62.1, 62.8, 63.0, 6
3.5, 63.9, 64.0, 64.3, 64.8, 65.0, 66.0, 66.8, 67.0, 67.4, 67.7, 68.0, 68.3, 68.4,
69.0, 69.2, 70.0, 70.1, 70.5, 71.0, 72.0, 72.1, 72.4, 72.5, 72.7, 72.8, 73.0, 73.
4, 74.0, 75.0, 75.3, 75.4, 76.0, 76.6, 77.0, 77.3, 78.0, 79.0, 79.3, 80.0, 80.3, 8
1.0, 82.0, 82.5, 82.6, 83.0, 83.9, 84.0, 84.2, 85.0, 85.4, 86.0, 87.0, 88.0, 89.0,
89.1, 90.0, 90.3, 90.6, 91.0, 91.2, 92.0, 92.4, 92.8, 93.0, 93.6, 94.0, 95.0, 95.
7, 96.0, 97.0, 98.0, 98.2, 98.3, 99.0, 99.2, 100.0, 101.0, 102.0, 103.0, 104.0, 10
5.0, 105.8, 106.0, 106.1, 107.0, 107.1, 108.0, 109.0, 110.0, 111.0, 111.5, 112.0,
113.0, 114.0, 115.0, 116.0, 117.0, 118.0, 119.0, 120.0, 121.0, 122.0, 123.0, 124.
0, 125.0, 126.0] [3.0, 3.1, 3.3, 3.4, 3.5, 3.6, 3.7, 3.8, 3.9, 4.0, 4.2, 4.4, 4.5,
4.6, 4.7, 4.8, 4.9, 5.1, 5.2, 5.3, 5.4, 5.5, 5.6, 5.7, 5.8, 5.9, 6.0, 6.1, 6.2, 6.
3, 6.5, 6.6, 6.7, 6.8, 7.0, 7.1, 7.2, 7.3, 7.4, 7.5, 7.6, 7.7, 7.8, 7.9, 8.0, 8.1,
8.2, 8.3, 8.4, 8.5, 8.6, 8.7, 8.8, 8.9, 9.0, 9.1, 9.2, 9.3, 9.4, 9.5, 9.6, 9.7, 9.
8, 9.9, 10.0, 10.1, 10.2, 10.3, 10.4, 10.5, 10.6, 10.7, 10.8, 10.9, 11.0, 11.1, 1
1.2, 11.3, 11.4, 11.5, 11.6, 11.7, 11.8, 11.9, 12.0, 12.1, 12.2, 12.3, 12.4, 12.5,
12.6, 12.7, 12.8, 12.9, 13.0, 13.1, 13.3, 13.4, 13.5, 13.6, 13.7, 13.8, 13.9, 14.
0, 14.1, 14.2, 14.3, 14.4, 14.5, 14.6, 14.7, 14.8, 14.9, 15.0, 15.01, 15.1, 15.2,
15.3, 15.4, 15.5, 15.6, 15.7, 15.8, 15.9, 16.0, 16.1, 16.2, 16.3, 16.4, 16.5, 16.
6, 16.7, 16.8, 16.9, 17.0, 17.1, 17.2, 17.3, 17.4, 17.5, 17.6, 17.7, 17.8, 17.9, 1
8.0, 18.1, 18.2, 18.3, 18.4, 18.5, 18.6, 18.7, 18.8, 18.9, 19.0, 19.1, 19.2, 19.3,
19.4, 19.5, 19.6, 19.7, 20.0, 20.1, 20.2, 20.4, 20.5, 20.6, 20.7, 20.8, 20.9, 21.
0, 21.2, 21.3, 21.4, 21.5, 21.6, 21.7, 21.8, 21.9, 22.1, 22.2, 22.3, 22.4, 22.5, 2
2.6, 22.7, 22.8, 22.9, 23.0, 23.2, 23.3, 23.4, 23.5, 23.6, 23.7, 23.9, 24.0, 24.1,
24.2, 24.3, 24.4, 24.5, 24.6, 24.7, 24.9, 25.0, 25.1, 25.3, 25.4, 25.5, 25.7, 25.
8, 26.1, 26.3, 26.4, 26.6, 26.7, 26.8, 26.9, 27.0, 27.2, 27.3, 27.4, 27.5, 28.0, 2
8.1, 28.2, 28.3, 28.5, 28.7, 28.8, 29.1, 29.2, 29.3, 29.4, 29.5, 29.9, 30.0, nan,
5.0, 22.0, 26.0, 29.0, 30.4, 30.5, 30.7, 30.8, 30.9, 31.0, 31.1, 31.3, 31.4, 31.5,
31.6, 31.7, 32.0, 32.1, 32.2, 32.3, 32.4, 32.5, 32.6, 32.9, 33.0, 33.1, 33.4, 33.
6, 33.7, 33.8, 33.9, 34.0, 34.1, 34.2, 34.3, 34.4, 34.5, 34.7, 34.8, 34.9, 35.0, 3
5.3, 35.6, 35.9, 36.0, 36.1, 36.2, 36.6, 36.8, 36.9, 37.0, 37.2, 37.5, 38.0, 38.6,
38.7, 38.8, 39.0, 39.1, 39.2, 39.3, 40.0, 40.2, 40.5, 40.7, 41.0, 41.2, 41.3, 41.
4, 41.5, 42.0, 42.5, 42.7, 42.9, 43.0, 43.4, 43.9, 44.0, 44.8, 45.0, 45.1, 45.7, 4
6.0, 46.8, 47.0, 47.3, 47.4, 48.0, 48.1, 48.2, 48.3, 48.9, 49.0, 49.3, 49.7, 50.0,
50.4, 50.6, 50.9, 51.0, 52.0, 52.1, 52.3, 52.7, 53.0, 53.1, 53.2, 53.5, 53.9, 54.
0, 54.9, 55.0, 55.6, 55.7, 55.9, 56.0, 56.3, 56.5, 57.0, 57.1, 58.0, 58.5, 58.7, 5

```
9.0, 60.0, 60.1, 60.2, 60.3, 60.6, 60.8, 61.0, 61.1, 61.3, 61.6, 61.9, 62.0, 62.1,
62.4, 62.7, 63.0, 63.5, 64.0, 64.2, 64.8, 65.0, 66.0, 66.7, 67.0, 67.7, 68.0, 68.
4, 69.0, 69.1, 69.2, 69.4, 69.9, 70.0, 71.0, 72.0, 72.4, 72.7, 73.0, 74.0, 74.9, 7
5.0, 76.0, 77.0, 77.1, 77.3, 78.0, 78.5, 79.0, 79.1, 80.0, 80.1, 80.3, 81.0, 81.6,
82.0, 82.5, 83.0, 83.4, 83.9, 84.0, 84.5, 85.0, 86.0, 87.0, 88.0, 88.8, 89.0, 90.
0, 90.9, 91.0, 92.0, 92.8, 93.0, 93.6, 94.0, 95.0, 96.0, 97.0, 98.0, 99.0, 99.2, 1
00.0, 101.0, 102.0, 103.0, 104.0, 105.0, 105.8, 106.0, 107.0, 107.1, 108.0, 109.0,
110.0, 111.0, 112.0, 113.0, 113.09]

C:\Application\Python\Anaconda3\lib\site-packages\ipykernel_launcher.py:2: Setting
WithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy

C:\Application\Python\Anaconda3\lib\site-packages\ipykernel_launcher.py:3: Setting
WithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy
  This is separate from the ipykernel package so we can avoid doing imports until
```

In [20]:

```
# data cleaning for min > max situation
dfbun[dfbun['d1_bun_max']<dfbun['d1_bun_min']]

# taking the maximum value of h1 and apache is reasonable
dfbun.loc[(dfbun.d1_bun_max<dfbun.d1_bun_min),'d1_bun_max']=dfbun[['bun_apache','h1_bun_max']].m
ax(axis=1)

print(len(dfbun[dfbun['d1_bun_max']<dfbun['d1_bun_min']]))
```

```
0

C:\Application\Python\Anaconda3\lib\site-packages\pandas\core\indexing.py:189: Set
tingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy
  self._setitem_with_indexer(indexer, value)
C:\Application\Python\Anaconda3\lib\site-packages\ipykernel_launcher.py:5: Setting
WithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy
  """
```

In [21]:

```
# data cleaning for value 0 situation
dfbun[dfbun.d1_bun_min==0]
dfbun[dfbun.d1_bun_min==0]

# check all null after finishing data cleaning for bun
dfbun[(dfbun.d1_bun_max.isnull())& (dfbun.h1_bun_max.isnull())&
          (dfbun.d1_bun_min.isnull())& (dfbun.h1_bun_min.isnull())&
          (dfbun.bun_apache.isnull())]
```

Out[21]:

	bun_apache	d1_bun_max	d1_bun_min	h1_bun_max	h1_bun_min
2	NaN	NaN	NaN	NaN	NaN
3	NaN	NaN	NaN	NaN	NaN
4	NaN	NaN	NaN	NaN	NaN
12	NaN	NaN	NaN	NaN	NaN
18	NaN	NaN	NaN	NaN	NaN
19	NaN	NaN	NaN	NaN	NaN
20	NaN	NaN	NaN	NaN	NaN
32	NaN	NaN	NaN	NaN	NaN
49	NaN	NaN	NaN	NaN	NaN
50	NaN	NaN	NaN	NaN	NaN
51	NaN	NaN	NaN	NaN	NaN
55	NaN	NaN	NaN	NaN	NaN
58	NaN	NaN	NaN	NaN	NaN
64	NaN	NaN	NaN	NaN	NaN
66	NaN	NaN	NaN	NaN	NaN
68	NaN	NaN	NaN	NaN	NaN
74	NaN	NaN	NaN	NaN	NaN
78	NaN	NaN	NaN	NaN	NaN
86	NaN	NaN	NaN	NaN	NaN
89	NaN	NaN	NaN	NaN	NaN
100	NaN	NaN	NaN	NaN	NaN
115	NaN	NaN	NaN	NaN	NaN
131	NaN	NaN	NaN	NaN	NaN
139	NaN	NaN	NaN	NaN	NaN
141	NaN	NaN	NaN	NaN	NaN
143	NaN	NaN	NaN	NaN	NaN
148	NaN	NaN	NaN	NaN	NaN
160	NaN	NaN	NaN	NaN	NaN
170	NaN	NaN	NaN	NaN	NaN
191	NaN	NaN	NaN	NaN	NaN
...	...	...	...	...	...
87088	NaN	NaN	NaN	NaN	NaN
87089	NaN	NaN	NaN	NaN	NaN

	bun_apache	d1_bun_max	d1_bun_min	h1_bun_max	h1_bun_min
87097	NaN	NaN	NaN	NaN	NaN
87111	NaN	NaN	NaN	NaN	NaN
87131	NaN	NaN	NaN	NaN	NaN
87145	NaN	NaN	NaN	NaN	NaN
87155	NaN	NaN	NaN	NaN	NaN
87162	NaN	NaN	NaN	NaN	NaN
87173	NaN	NaN	NaN	NaN	NaN
87175	NaN	NaN	NaN	NaN	NaN
87207	NaN	NaN	NaN	NaN	NaN
87214	NaN	NaN	NaN	NaN	NaN
87236	NaN	NaN	NaN	NaN	NaN
87255	NaN	NaN	NaN	NaN	NaN
87257	NaN	NaN	NaN	NaN	NaN
87274	NaN	NaN	NaN	NaN	NaN
87279	NaN	NaN	NaN	NaN	NaN
87314	NaN	NaN	NaN	NaN	NaN
87315	NaN	NaN	NaN	NaN	NaN
87382	NaN	NaN	NaN	NaN	NaN
87385	NaN	NaN	NaN	NaN	NaN
87387	NaN	NaN	NaN	NaN	NaN
87392	NaN	NaN	NaN	NaN	NaN
87393	NaN	NaN	NaN	NaN	NaN
87394	NaN	NaN	NaN	NaN	NaN
87395	NaN	NaN	NaN	NaN	NaN
87422	NaN	NaN	NaN	NaN	NaN
87429	NaN	NaN	NaN	NaN	NaN
87459	NaN	NaN	NaN	NaN	NaN
87460	NaN	NaN	NaN	NaN	NaN

9230 rows × 5 columns

creatinine

In [22]:

```
# variable: creatinine
dfcreatinine=training.filter(regex='creatinine')

# check day values missing while hour values not
print(len(dfcreatinine[(dfcreatinine.h1_creatinine_max.notna()) & (dfcreatinine.d1_creatinine_ma
x.isna())]),
      len(dfcreatinine[(dfcreatinine.h1_creatinine_min.notna()) & (dfcreatinine.d1_creatinine_mi
n.isna())]))
```

0 0

In [23]:

```
# Check if missing values are in pairs for max and min
print(len(dfcreatinine[(dfcreatinine.d1_creatinine_max.notna()) & (dfcreatinine.d1_creatinine_mi
n.isna())]),
      len(dfcreatinine[(dfcreatinine.d1_creatinine_max.isna()) & (dfcreatinine.d1_creatinine_min
.notna())]),
      len(dfcreatinine[(dfcreatinine.creatinine_apache.notna()) & (dfcreatinine.d1_creatinine_ma
x.isna())]),
      len(dfcreatinine[dfcreatinine.d1_creatinine_max.notna()]))
```

0 0 730 77877

In [24]:

```
# replace apache covariate with null values in d1 creatinine
dfcreatinine.loc[dfcreatinine['d1_creatinine_max'].isnull(),'d1_creatinine_max']=dfcreatinine['c
reatinine_apache']
dfcreatinine.loc[dfcreatinine['d1_creatinine_min'].isnull(),'d1_creatinine_min']=dfcreatinine['c
reatinine_apache']
```

```
C:\Application\Python\Anaconda3\lib\site-packages\pandas\core\indexing.py:189: Set
tingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy
  self._setitem_with_indexer(indexer, value)
C:\Application\Python\Anaconda3\lib\site-packages\ipykernel_launcher.py:2: Setting
WithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy

C:\Application\Python\Anaconda3\lib\site-packages\ipykernel_launcher.py:3: Setting
WithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy
  This is separate from the ipykernel package so we can avoid doing imports until
```

In [25]:

```
# data cleaning for min > max situation
dfcreatinine[dfcreatinine['d1_creatinine_max']<dfcreatinine['d1_creatinine_min']]

# where apache covariate is taken as it is reasonable
dfcreatinine.loc[(dfcreatinine.d1_creatinine_max<dfcreatinine.d1_creatinine_min),'d1_creatinine_
max']=dfcreatinine['creatinine_apache']

len(dfcreatinine[dfcreatinine['d1_creatinine_max']<dfcreatinine['d1_creatinine_min']])
```

```
C:\Application\Python\Anaconda3\lib\site-packages\pandas\core\indexing.py:189: Set
tingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy
  self._setitem_with_indexer(indexer, value)
C:\Application\Python\Anaconda3\lib\site-packages\ipykernel_launcher.py:5: Setting
WithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy
  """
```

Out[25]:

```
0
```

In [26]:

```
# data cleaning for value 0 situation
dfcreatinine[dfcreatinine.d1_creatinine_max==0]
dfcreatinine[dfcreatinine.d1_creatinine_min==0]

# check all null after finishing data cleaning for creatinine
dfcreatinine[(dfcreatinine.d1_creatinine_max.isnull())& (dfcreatinine.h1_creatinine_max.isnull
())&
          (dfcreatinine.d1_creatinine_min.isnull())& (dfcreatinine.h1_creatinine_min.isnull())&
          (dfcreatinine.creatinine_apache.isnull())]
```

Out[26]:

	creatinine_apache	d1_creatinine_max	d1_creatinine_min	h1_creatinine_max
2	NaN	NaN	NaN	NaN
3	NaN	NaN	NaN	NaN
4	NaN	NaN	NaN	NaN
12	NaN	NaN	NaN	NaN
18	NaN	NaN	NaN	NaN
19	NaN	NaN	NaN	NaN
20	NaN	NaN	NaN	NaN
32	NaN	NaN	NaN	NaN
49	NaN	NaN	NaN	NaN
50	NaN	NaN	NaN	NaN
51	NaN	NaN	NaN	NaN
55	NaN	NaN	NaN	NaN
58	NaN	NaN	NaN	NaN
64	NaN	NaN	NaN	NaN
66	NaN	NaN	NaN	NaN
68	NaN	NaN	NaN	NaN
74	NaN	NaN	NaN	NaN
78	NaN	NaN	NaN	NaN
86	NaN	NaN	NaN	NaN
89	NaN	NaN	NaN	NaN
100	NaN	NaN	NaN	NaN
115	NaN	NaN	NaN	NaN
131	NaN	NaN	NaN	NaN
139	NaN	NaN	NaN	NaN
141	NaN	NaN	NaN	NaN
143	NaN	NaN	NaN	NaN
148	NaN	NaN	NaN	NaN
160	NaN	NaN	NaN	NaN
170	NaN	NaN	NaN	NaN
191	NaN	NaN	NaN	NaN
...	...	...	...	...
87088	NaN	NaN	NaN	NaN
87089	NaN	NaN	NaN	NaN

	creatinine_apache	d1_creatinine_max	d1_creatinine_min	h1_creatinine_max
87097	NaN	NaN	NaN	NaN
87111	NaN	NaN	NaN	NaN
87131	NaN	NaN	NaN	NaN
87145	NaN	NaN	NaN	NaN
87155	NaN	NaN	NaN	NaN
87162	NaN	NaN	NaN	NaN
87173	NaN	NaN	NaN	NaN
87175	NaN	NaN	NaN	NaN
87207	NaN	NaN	NaN	NaN
87214	NaN	NaN	NaN	NaN
87236	NaN	NaN	NaN	NaN
87255	NaN	NaN	NaN	NaN
87257	NaN	NaN	NaN	NaN
87274	NaN	NaN	NaN	NaN
87279	NaN	NaN	NaN	NaN
87314	NaN	NaN	NaN	NaN
87315	NaN	NaN	NaN	NaN
87382	NaN	NaN	NaN	NaN
87385	NaN	NaN	NaN	NaN
87387	NaN	NaN	NaN	NaN
87392	NaN	NaN	NaN	NaN
87393	NaN	NaN	NaN	NaN
87394	NaN	NaN	NaN	NaN
87395	NaN	NaN	NaN	NaN
87422	NaN	NaN	NaN	NaN
87429	NaN	NaN	NaN	NaN
87459	NaN	NaN	NaN	NaN
87460	NaN	NaN	NaN	NaN

8878 rows × 5 columns

glucose

In [27]:

```
# variable: glucose
dfglucose=training.filter(regex='glucose')

# check day values missing while hour values not
print(len(dfglucose[(dfglucose.h1_glucose_max.notna()) & (dfglucose.d1_glucose_max.isna())]),
      len(dfglucose[(dfglucose.h1_glucose_min.notna()) & (dfglucose.d1_glucose_min.isna())]))
```

0 0

In [28]:

```
# Check if missing values are in pairs for max and min
print(len(dfglucose[(dfglucose.d1_glucose_max.notna()) & (dfglucose.d1_glucose_min.isna())]),
      len(dfglucose[(dfglucose.d1_glucose_max.isna()) & (dfglucose.d1_glucose_min.notna())]),
      len(dfglucose[(dfglucose.glucose_apache.notna()) & (dfglucose.d1_glucose_max.isna())]),
      len(dfglucose[dfglucose.d1_glucose_max.notna()]))
```

0 0 397 81978

In [29]:

```
# replace apache covariate with null values in d1 glucose
dfglucose.loc[dfglucose['d1_glucose_max'].isnull(),'d1_glucose_max']=dfglucose['glucose_apache']
dfglucose.loc[dfglucose['d1_glucose_min'].isnull(),'d1_glucose_min']=dfglucose['glucose_apache']

# list unique values in d1_max data to check for abnoramal data
# print(sorted(dfglucose.d1_glucose_max.unique()),
#       sorted(dfglucose.d1_glucose_min.unique()))
```

```
C:\Application\Python\Anaconda3\lib\site-packages\pandas\core\indexing.py:189: Set
tingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy
  self._setitem_with_indexer(indexer, value)
C:\Application\Python\Anaconda3\lib\site-packages\ipykernel_launcher.py:2: Setting
WithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy

C:\Application\Python\Anaconda3\lib\site-packages\ipykernel_launcher.py:3: Setting
WithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy
  This is separate from the ipykernel package so we can avoid doing imports until
```

In [30]:

```
# data cleaning for min > max situation
dfglucose[dfglucose['d1_glucose_max']<dfglucose['d1_glucose_min']]

# replace d1 with h1
dfglucose.loc[(dfglucose.d1_glucose_max<dfglucose.d1_glucose_min),'d1_glucose_max']=dfglucose['h
1_glucose_max']

print(len(dfglucose[dfglucose['d1_glucose_max']<dfglucose['d1_glucose_min']]))
```

0

```
C:\Application\Python\Anaconda3\lib\site-packages\pandas\core\indexing.py:189: Set
tingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy
  self._setitem_with_indexer(indexer, value)
C:\Application\Python\Anaconda3\lib\site-packages\ipykernel_launcher.py:5: Setting
WithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy
  """
```

In [31]:

```
# data cleaning for value 0 situation
dfglucose[dfglucose.d1_glucose_max==0]
dfglucose[dfglucose.d1_glucose_min==0]

# check all null after finishing data cleaning for glucose
len(dfglucose[(dfglucose.d1_glucose_max.isnull())& (dfglucose.h1_glucose_max.isnull())&
          (dfglucose.d1_glucose_min.isnull())& (dfglucose.h1_glucose_min.isnull())&
          (dfglucose.glucose_apache.isnull())])
```

Out[31]:

5110

hco3

In [32]:

```
# variable: hco3
dfhco3=training.filter(regex='hco3')

# check day values missing while hour values not
print(len(dfhco3[(dfhco3.h1_hco3_max.notna()) & (dfhco3.d1_hco3_max.isna())]),
      len(dfhco3[(dfhco3.h1_hco3_min.notna()) & (dfhco3.d1_hco3_min.isna())]))

# Check if missing values are in pairs for max and min
print(len(dfhco3[(dfhco3.d1_hco3_max.notna()) & (dfhco3.d1_hco3_min.isna())]),
      len(dfhco3[(dfhco3.d1_hco3_max.isna()) & (dfhco3.d1_hco3_min.notna())]),
      len(dfhco3[dfhco3.d1_hco3_max.notna()]))
```

```
0 0
0 0 73260
```

In [33]:

```
# data cleaning for min > max situation
dfhco3[dfhco3['d1_hco3_max']<dfhco3['d1_hco3_min']]

# replace d1 max with h1 max
dfhco3.loc[(dfhco3['d1_hco3_max']<dfhco3['d1_hco3_min']) & (dfhco3.h1_hco3_max.notnull()),'d1_hc
o3_max']=dfhco3['h1_hco3_max']
```

```
C:\Application\Python\Anaconda3\lib\site-packages\pandas\core\indexing.py:189: Set
tingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy
  self._setitem_with_indexer(indexer, value)
C:\Application\Python\Anaconda3\lib\site-packages\ipykernel_launcher.py:5: Setting
WithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy
  """
```

In [34]:

```
# look into the data to find reasonable imputation method
backup=dfhco3[dfhco3['d1_hco3_max']<dfhco3['d1_hco3_min']]
print(sorted(backup.d1_hco3_max.unique()),
      sorted(backup.d1_hco3_min.unique()))

# Check the frequency for ever d1 min value
hco3impute=dfhco3[(dfhco3['d1_hco3_min']>=31) & (dfhco3['d1_hco3_min']<=39)]
freq= hco3impute.groupby('d1_hco3_min')['d1_hco3_max'].value_counts()
freq
```

[12.0] [31.0, 35.0, 36.0, 37.0, 38.0, 38.2, 39.0]

Out[34]:

dl_hco3_min	dl_hco3_max	
31.0	31.0	862
	32.0	72
	33.0	52
	34.0	27
	35.0	18
	36.0	9
	39.0	7
	38.0	6
	37.0	2
	12.0	1
	32.3	1
	40.0	1
31.1	31.1	4
31.2	31.2	1
31.3	31.3	1
31.4	31.4	2
31.5	31.5	1
31.6	31.6	2
31.7	31.7	4
31.8	31.8	2
	33.8	1
31.9	31.9	1
32.0	32.0	668
	33.0	47
	34.0	33
	35.0	27
	36.0	19
	37.0	6
	40.0	5
	39.0	4
		...
36.0	36.0	192
	37.0	18
	38.0	12
	40.0	10
	39.0	7
	12.0	2
	36.6	1
36.2	36.2	1
36.3	36.3	1
36.7	36.7	1
36.9	38.4	1
37.0	37.0	167
	40.0	10
	12.0	9
	38.0	6
	39.0	6
37.1	37.1	3
37.6	38.0	1
37.7	37.7	1
37.8	37.8	1
38.0	38.0	133
	39.0	14
	40.0	7
	12.0	2
38.1	39.3	1
38.2	12.0	1
38.7	38.7	1

```
39.0          40.0           253
              12.0            83
              39.0             3
Name: d1_hco3_max, Length: 105, dtype: int64
```

In [35]:

```
# replace d1 max with following strategies:
# for all 39, impute 40
# for others, which are blow 39,impute themselves
dfhco3.loc[(dfhco3['d1_hco3_max']<dfhco3['d1_hco3_min']) & (dfhco3.d1_hco3_min==39),'d1_hco3_max']=40
dfhco3.loc[(dfhco3['d1_hco3_max']<dfhco3['d1_hco3_min']) & (dfhco3.d1_hco3_min<39),'d1_hco3_max']=dfhco3['d1_hco3_min']

# data cleaning for min > max situation
dfhco3[dfhco3['d1_hco3_max']<dfhco3['d1_hco3_min']]
```

```
C:\Application\Python\Anaconda3\lib\site-packages\pandas\core\indexing.py:189: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy
  self._setitem_with_indexer(indexer, value)
C:\Application\Python\Anaconda3\lib\site-packages\ipykernel_launcher.py:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy
  after removing the cwd from sys.path.
C:\Application\Python\Anaconda3\lib\site-packages\ipykernel_launcher.py:5: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy
  """
```

Out[35]:

	d1_hco3_max	d1_hco3_min	h1_hco3_max	h1_hco3_min

In [36]:

```
# check all null after finishing data cleaning for hco3
dfhco3[(dfhco3.d1_hco3_max.isnull())& (dfhco3.h1_hco3_max.isnull())&
          (dfhco3.d1_hco3_min.isnull())& (dfhco3.h1_hco3_min.isnull())]
```

Out[36]:

	d1_hco3_max	d1_hco3_min	h1_hco3_max	h1_hco3_min
2	NaN	NaN	NaN	NaN
3	NaN	NaN	NaN	NaN
4	NaN	NaN	NaN	NaN
12	NaN	NaN	NaN	NaN
18	NaN	NaN	NaN	NaN
19	NaN	NaN	NaN	NaN
20	NaN	NaN	NaN	NaN
32	NaN	NaN	NaN	NaN
49	NaN	NaN	NaN	NaN
50	NaN	NaN	NaN	NaN
51	NaN	NaN	NaN	NaN
53	NaN	NaN	NaN	NaN
55	NaN	NaN	NaN	NaN
58	NaN	NaN	NaN	NaN
64	NaN	NaN	NaN	NaN
66	NaN	NaN	NaN	NaN
68	NaN	NaN	NaN	NaN
74	NaN	NaN	NaN	NaN
78	NaN	NaN	NaN	NaN
86	NaN	NaN	NaN	NaN
89	NaN	NaN	NaN	NaN
100	NaN	NaN	NaN	NaN
115	NaN	NaN	NaN	NaN
131	NaN	NaN	NaN	NaN
134	NaN	NaN	NaN	NaN
139	NaN	NaN	NaN	NaN
141	NaN	NaN	NaN	NaN
143	NaN	NaN	NaN	NaN
148	NaN	NaN	NaN	NaN
160	NaN	NaN	NaN	NaN
...	...	...	...	...
87197	NaN	NaN	NaN	NaN
87207	NaN	NaN	NaN	NaN

	d1_hco3_max	d1_hco3_min	h1_hco3_max	h1_hco3_min
87214	NaN	NaN	NaN	NaN
87236	NaN	NaN	NaN	NaN
87255	NaN	NaN	NaN	NaN
87257	NaN	NaN	NaN	NaN
87270	NaN	NaN	NaN	NaN
87274	NaN	NaN	NaN	NaN
87279	NaN	NaN	NaN	NaN
87285	NaN	NaN	NaN	NaN
87301	NaN	NaN	NaN	NaN
87314	NaN	NaN	NaN	NaN
87315	NaN	NaN	NaN	NaN
87345	NaN	NaN	NaN	NaN
87382	NaN	NaN	NaN	NaN
87385	NaN	NaN	NaN	NaN
87387	NaN	NaN	NaN	NaN
87392	NaN	NaN	NaN	NaN
87393	NaN	NaN	NaN	NaN
87394	NaN	NaN	NaN	NaN
87395	NaN	NaN	NaN	NaN
87408	NaN	NaN	NaN	NaN
87409	NaN	NaN	NaN	NaN
87414	NaN	NaN	NaN	NaN
87415	NaN	NaN	NaN	NaN
87422	NaN	NaN	NaN	NaN
87429	NaN	NaN	NaN	NaN
87459	NaN	NaN	NaN	NaN
87460	NaN	NaN	NaN	NaN
87474	NaN	NaN	NaN	NaN

14225 rows × 4 columns

In [37]:

```
dfhco3.loc[(dfhco3['d1_hco3_max']<dfhco3['d1_hco3_min']) & (dfhco3.h1_hco3_min==39),'d1_hco3_ma
x']
```

Out[37]:

```
Series([], Name: d1_hco3_max, dtype: float64)
```

hemoglobin

In [38]:

```
# variable: hemoglobin
dfhemaglobin=training.filter(regex='hemaglobin')

# check day values missing while hour values not
print(len(dfhemaglobin[(dfhemaglobin.h1_hemaglobin_max.notna()) & (dfhemaglobin.d1_hemaglobin_ma
x.isna())]),
      len(dfhemaglobin[(dfhemaglobin.h1_hemaglobin_min.notna()) & (dfhemaglobin.d1_hemaglobin_mi
n.isna())]))
```

```
0 0
```

In [39]:

```
# Check if missing values are in pairs for max and min
print(len(dfhemaglobin[(dfhemaglobin.d1_hemaglobin_max.notna()) & (dfhemaglobin.d1_hemaglobin_mi
n.isna())]),
      len(dfhemaglobin[(dfhemaglobin.d1_hemaglobin_max.isna()) & (dfhemaglobin.d1_hemaglobin_min
.notna())]),
      len(dfhemaglobin[dfhemaglobin.d1_hemaglobin_max.notna()]))
```

```
0 0 75966
```

In [40]:

```
# data cleaning for min > max situation
dfhemaglobin[dfhemaglobin['d1_hemaglobin_max']<dfhemaglobin['d1_hemaglobin_min']]
```

Out[40]:

	d1_hemaglobin_max	d1_hemaglobin_min	h1_hemaglobin_max	h1_hemaglob
2851	6.8	16.7	NaN	NaN
5726	6.8	16.2	17.2	17.2
8501	6.8	16.7	17.2	17.2
9651	6.8	16.7	NaN	NaN
14596	6.8	16.7	NaN	NaN
14828	6.8	16.7	NaN	NaN
14838	6.8	16.7	NaN	NaN
15480	6.8	16.7	NaN	NaN
18552	6.8	16.7	17.2	17.2
18559	6.8	16.7	NaN	NaN
20161	6.8	16.2	NaN	NaN
21253	6.8	16.7	NaN	NaN
21975	6.8	16.7	NaN	NaN
24603	6.8	16.7	NaN	NaN
26076	6.8	16.7	NaN	NaN
27716	6.8	16.7	NaN	NaN
29655	6.8	16.7	17.2	17.2
30087	6.8	16.7	NaN	NaN
37133	6.8	13.9	NaN	NaN
37388	6.8	16.1	NaN	NaN
40943	6.8	16.7	17.2	17.2
42152	6.8	12.8	17.2	17.0
42934	6.8	11.2	17.2	17.2
43189	6.8	13.3	NaN	NaN
44638	6.8	15.2	17.2	17.2
46496	6.8	16.7	NaN	NaN
50264	6.8	16.7	NaN	NaN
51725	6.8	16.7	NaN	NaN
52853	6.8	15.8	17.2	17.2
54369	6.8	12.8	17.2	17.2
55001	6.8	13.4	13.4	13.4
57969	6.8	16.7	NaN	NaN
59092	6.8	16.3	NaN	NaN

	d1_hemaglobin_max	d1_hemaglobin_min	h1_hemaglobin_max	h1_hemaglob
60683	6.8	16.7	NaN	NaN
61414	6.8	16.7	NaN	NaN
61696	6.8	13.7	14.6	14.6
62590	6.8	12.2	NaN	NaN
63155	6.8	12.7	15.1	15.1
65389	6.8	16.7	NaN	NaN
67061	6.8	15.9	17.2	17.2
70111	6.8	14.3	17.2	17.2
70575	6.8	14.7	NaN	NaN
75948	6.8	15.0	17.2	17.2
76601	6.8	15.1	17.2	17.2
78519	6.8	16.0	17.2	17.2
78948	6.8	15.5	17.2	17.2
79566	6.8	16.7	NaN	NaN
80357	6.8	15.2	NaN	NaN
80766	6.8	16.7	NaN	NaN
83614	6.8	16.7	NaN	NaN
86005	6.8	16.7	NaN	NaN
86590	6.8	16.7	NaN	NaN

In [41]:

```
# data cleaning for min > max situation
dfhemaglobin[dfhemaglobin['d1_hemaglobin_max']<dfhemaglobin['d1_hemaglobin_min']]

# where h1_hemaglobin_max is taken as it is reasonable
dfhemaglobin.loc[(dfhemaglobin.d1_hemaglobin_max<dfhemaglobin.d1_hemaglobin_min) & (dfhemaglobin
.h1_hemaglobin_max>dfhemaglobin.d1_hemaglobin_max),'d1_hemaglobin_max']=dfhemaglobin['h1_hemaglo
bin_max']

len(dfhemaglobin[dfhemaglobin['d1_hemaglobin_max']<dfhemaglobin['d1_hemaglobin_min']])
```

```
C:\Application\Python\Anaconda3\lib\site-packages\pandas\core\indexing.py:189: Set
tingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy
  self._setitem_with_indexer(indexer, value)
C:\Application\Python\Anaconda3\lib\site-packages\ipykernel_launcher.py:5: Setting
WithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy
  """
```

Out[41]:

```
33
```

In [42]:

```
# look into the data to find reasonable imputation method
backup=dfhemaglobin[dfhemaglobin['d1_hemaglobin_max']<dfhemaglobin['d1_hemaglobin_min']]
print(sorted(backup.d1_hemaglobin_max.unique()),
      sorted(backup.d1_hemaglobin_min.unique()))

# Check the frequency for ever d1 min value
hemaglobinimpute=dfhemaglobin[(dfhemaglobin['d1_hemaglobin_min']>=12.2) &
                              (dfhemaglobin['d1_hemaglobin_min']<=16.7) &
                              (dfhemaglobin['d1_hemaglobin_max']!=6.8)]
freq= hemaglobinimpute.groupby('d1_hemaglobin_min')['d1_hemaglobin_max'].value_counts()
freq
```

[6.8] [12.2, 13.3, 13.9, 14.7, 15.2, 16.1, 16.2, 16.3, 16.7]

Out[42]:

d1_hemaglobin_min	d1_hemaglobin_max	
12.2	12.2	792
	12.6	28
	12.8	27
	13.2	26
	12.3	22
	12.7	22
	13.6	21
	13.1	20
	12.5	18
	12.4	17
	12.9	17
	13.0	16
	13.3	16
	13.5	15
	13.7	13
	13.9	11
	13.8	8
	14.4	7
	13.4	6
	14.5	6
	14.3	5
	14.6	5
	14.2	4
	14.9	4
	15.0	4
	15.1	4
	14.0	3
	14.1	3
	15.2	3
	14.8	2
		...
16.3	16.3	64
	16.7	5
	17.2	5
	16.5	3
	16.6	2
	16.9	2
	17.0	1
	17.1	1
16.4	16.4	63
	16.8	4
	17.2	3
	16.7	2
	16.6	1
	17.0	1
16.5	16.5	47
	17.2	3
	16.8	1
	16.9	1
	17.0	1
	17.1	1
16.6	16.6	40
	16.7	6
	16.9	2
	17.2	2
	16.8	1
16.7	17.2	238
	16.8	37

```
                    16.9                   31
                    17.1                   29
                    17.0                   25
Name: d1_hemaglobin_max, Length: 1048, dtype: int64
```

In [43]:

```
# Due to too many values for each d1 min values, impute the average of each group
hemaglobinmean=hemaglobinimpute.groupby('d1_hemaglobin_min')['d1_hemaglobin_max'].mean()
# print(hemaglobinmean)
# check1=dfhemaglobin[dfhemaglobin['d1_hemaglobin_max']<dfhemaglobin['d1_hemaglobin_min']]

for i in (sorted(backup.d1_hemaglobin_min.unique())):
    imputation=hemaglobinimpute.loc[hemaglobinimpute['d1_hemaglobin_min'] == i, 'd1_hemaglobin_max'].mean()
    dfhemaglobin.loc[(dfhemaglobin.d1_hemaglobin_max<dfhemaglobin.d1_hemaglobin_min) &
                     (dfhemaglobin.d1_hemaglobin_min==i),'d1_hemaglobin_max']=round(imputation,1)

# imputation check
# dfhemaglobin = dfhemaglobin[dfhemaglobin.index.isin(check1.index)]
# dfhemaglobin
```

```
C:\Application\Python\Anaconda3\lib\site-packages\pandas\core\indexing.py:189: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy
  self._setitem_with_indexer(indexer, value)
C:\Application\Python\Anaconda3\lib\site-packages\ipykernel_launcher.py:9: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy
  if __name__ == '__main__':
```

In [44]:

```
# data cleaning for value 0 situation
dfhemaglobin[dfhemaglobin.d1_hemaglobin_max==0]
dfhemaglobin[dfhemaglobin.d1_hemaglobin_min==0]

# check all null after finishing data cleaning for hemaglobin
dfhemaglobin[(dfhemaglobin.d1_hemaglobin_max.isnull())& (dfhemaglobin.h1_hemaglobin_max.isnull
())&
          (dfhemaglobin.d1_hemaglobin_min.isnull())& (dfhemaglobin.h1_hemaglobin_min.isnull())]
```

Out[44]:

	d1_hemaglobin_max	d1_hemaglobin_min	h1_hemaglobin_max	h1_hemaglob
2	NaN	NaN	NaN	NaN
4	NaN	NaN	NaN	NaN
11	NaN	NaN	NaN	NaN
20	NaN	NaN	NaN	NaN
53	NaN	NaN	NaN	NaN
58	NaN	NaN	NaN	NaN
66	NaN	NaN	NaN	NaN
86	NaN	NaN	NaN	NaN
87	NaN	NaN	NaN	NaN
91	NaN	NaN	NaN	NaN
99	NaN	NaN	NaN	NaN
103	NaN	NaN	NaN	NaN
107	NaN	NaN	NaN	NaN
110	NaN	NaN	NaN	NaN
123	NaN	NaN	NaN	NaN
131	NaN	NaN	NaN	NaN
139	NaN	NaN	NaN	NaN
168	NaN	NaN	NaN	NaN
180	NaN	NaN	NaN	NaN
183	NaN	NaN	NaN	NaN
185	NaN	NaN	NaN	NaN
204	NaN	NaN	NaN	NaN
212	NaN	NaN	NaN	NaN
214	NaN	NaN	NaN	NaN
226	NaN	NaN	NaN	NaN
236	NaN	NaN	NaN	NaN
237	NaN	NaN	NaN	NaN
244	NaN	NaN	NaN	NaN
247	NaN	NaN	NaN	NaN
252	NaN	NaN	NaN	NaN
...	...	...	...	...
87270	NaN	NaN	NaN	NaN
87274	NaN	NaN	NaN	NaN

	d1_hemaglobin_max	d1_hemaglobin_min	h1_hemaglobin_max	h1_hemaglob
87279	NaN	NaN	NaN	NaN
87311	NaN	NaN	NaN	NaN
87314	NaN	NaN	NaN	NaN
87315	NaN	NaN	NaN	NaN
87329	NaN	NaN	NaN	NaN
87342	NaN	NaN	NaN	NaN
87366	NaN	NaN	NaN	NaN
87374	NaN	NaN	NaN	NaN
87378	NaN	NaN	NaN	NaN
87382	NaN	NaN	NaN	NaN
87385	NaN	NaN	NaN	NaN
87387	NaN	NaN	NaN	NaN
87393	NaN	NaN	NaN	NaN
87394	NaN	NaN	NaN	NaN
87395	NaN	NaN	NaN	NaN
87408	NaN	NaN	NaN	NaN
87415	NaN	NaN	NaN	NaN
87419	NaN	NaN	NaN	NaN
87422	NaN	NaN	NaN	NaN
87443	NaN	NaN	NaN	NaN
87449	NaN	NaN	NaN	NaN
87452	NaN	NaN	NaN	NaN
87458	NaN	NaN	NaN	NaN
87459	NaN	NaN	NaN	NaN
87463	NaN	NaN	NaN	NaN
87465	NaN	NaN	NaN	NaN
87468	NaN	NaN	NaN	NaN
87479	NaN	NaN	NaN	NaN

11519 rows × 4 columns

hematocrit

In [45]:

```
# variable: hematocrit
dfhematocrit=training.filter(regex='hematocrit')

# check day values missing while hour values not
print(len(dfhematocrit[(dfhematocrit.h1_hematocrit_max.notna()) & (dfhematocrit.d1_hematocrit_ma
x.isna())]),
      len(dfhematocrit[(dfhematocrit.h1_hematocrit_min.notna()) & (dfhematocrit.d1_hematocrit_mi
n.isna())]))
```

```
0 0
```

In [46]:

```
# Check if missing values are in pairs for max and min
print(len(dfhematocrit[(dfhematocrit.d1_hematocrit_max.notna()) & (dfhematocrit.d1_hematocrit_mi
n.isna())]),
      len(dfhematocrit[(dfhematocrit.d1_hematocrit_max.isna()) & (dfhematocrit.d1_hematocrit_min
.notna())]),
      len(dfhematocrit[(dfhematocrit.hematocrit_apache.notna()) & (dfhematocrit.d1_hematocrit_ma
x.isna())]),
      len(dfhematocrit[dfhematocrit.d1_hematocrit_max.notna()]))
```

```
0 0 865 76453
```

In [47]:

```
# replace apache covariate with null values in d1 hematocrit
dfhematocrit.loc[dfhematocrit['d1_hematocrit_max'].isnull(),'d1_hematocrit_max']=dfhematocrit['h
ematocrit_apache']
dfhematocrit.loc[dfhematocrit['d1_hematocrit_min'].isnull(),'d1_hematocrit_min']=dfhematocrit['h
ematocrit_apache']

# list unique values in d1_max data to check for abnoramal data
print(sorted(dfhematocrit.d1_hematocrit_max.unique()),
      sorted(dfhematocrit.d1_hematocrit_min.unique()))
```

[20.6, 22.3, 24.1, 25.5, 25.7, 26.6, 27.0, 27.4, 28.9, 29.7, 30.3, 30.8, 32.0, 32.
2, 32.6, 33.1, 33.3, 36.9, nan, 16.2, 19.3, 19.9, 20.2, 20.3, 20.4, 20.5, 20.7, 2
0.8, 20.9, 21.0, 21.1, 21.2, 21.3, 21.4, 21.5, 21.6, 21.7, 21.8, 21.9, 22.0, 22.1,
22.2, 22.4, 22.5, 22.6, 22.7, 22.8, 22.9, 23.0, 23.1, 23.2, 23.3, 23.4, 23.5, 23.
6, 23.7, 23.8, 23.9, 24.0, 24.2, 24.3, 24.4, 24.5, 24.6, 24.7, 24.8, 24.9, 25.0, 2
5.1, 25.2, 25.3, 25.4, 25.6, 25.8, 25.9, 26.0, 26.1, 26.2, 26.3, 26.4, 26.5, 26.7,
26.8, 26.9, 27.1, 27.2, 27.3, 27.5, 27.6, 27.7, 27.8, 27.9, 28.0, 28.1, 28.2, 28.
3, 28.4, 28.5, 28.6, 28.7, 28.8, 29.0, 29.1, 29.2, 29.3, 29.4, 29.5, 29.6, 29.8, 2
9.9, 30.0, 30.1, 30.2, 30.4, 30.5, 30.6, 30.7, 30.9, 31.0, 31.1, 31.2, 31.3, 31.4,
31.5, 31.6, 31.7, 31.8, 31.9, 32.1, 32.3, 32.4, 32.5, 32.7, 32.8, 32.9, 33.0, 33.
2, 33.4, 33.5, 33.6, 33.7, 33.8, 33.9, 34.0, 34.1, 34.2, 34.3, 34.4, 34.5, 34.6, 3
4.7, 34.8, 34.9, 35.0, 35.1, 35.2, 35.3, 35.4, 35.5, 35.6, 35.7, 35.8, 35.9, 36.0,
36.1, 36.2, 36.3, 36.4, 36.5, 36.6, 36.7, 36.8, 37.0, 37.1, 37.2, 37.3, 37.4, 37.
5, 37.6, 37.7, 37.8, 37.9, 38.0, 38.1, 38.2, 38.3, 38.4, 38.5, 38.6, 38.7, 38.8, 3
8.9, 39.0, 39.1, 39.2, 39.3, 39.4, 39.5, 39.6, 39.7, 39.8, 39.9, 40.0, 40.1, 40.2,
40.3, 40.4, 40.5, 40.6, 40.7, 40.8, 40.9, 41.0, 41.1, 41.2, 41.3, 41.4, 41.5, 41.
6, 41.7, 41.8, 41.9, 42.0, 42.1, 42.2, 42.3, 42.4, 42.5, 42.6, 42.7, 42.8, 42.9, 4
3.0, 43.1, 43.2, 43.3, 43.4, 43.5, 43.6, 43.7, 43.8, 43.9, 44.0, 44.1, 44.2, 44.3,
44.4, 44.5, 44.6, 44.7, 44.8, 44.9, 45.0, 45.1, 45.2, 45.3, 45.4, 45.5, 45.6, 45.
7, 45.8, 45.9, 46.0, 46.1, 46.2, 46.3, 46.4, 46.5, 46.6, 46.7, 46.8, 46.9, 47.0, 4
7.1, 47.2, 47.3, 47.4, 47.5, 47.6, 47.7, 47.8, 47.9, 48.0, 48.1, 48.2, 48.3, 48.4,
48.5, 48.6, 48.7, 48.8, 48.9, 49.0, 49.1, 49.2, 49.3, 49.4, 49.5, 49.6, 49.7, 49.
8, 49.9, 50.0, 50.1, 50.2, 50.3, 50.4, 50.5, 50.6, 50.7, 50.8, 50.9, 51.0, 51.1, 5
1.2, 51.3, 51.4, 51.5] [19.8, 19.9, 20.0, 20.5, 21.8, 21.9, 24.1, 24.4, 24.5, 24.
8, 25.5, 25.7, 27.0, 27.2, 27.3, 27.4, 28.0, 28.3, 28.5, 28.7, 28.9, 29.0, 29.7, 3
0.1, 30.3, 30.4, 30.5, 30.7, 30.8, 31.2, 31.8, 32.0, 32.2, 33.1, 33.4, 33.8, 34.4,
34.7, 34.9, 35.1, 35.6, 35.7, 35.9, 36.0, 36.1, nan, 16.1, 16.2, 16.3, 16.4, 16.5,
16.6, 16.7, 16.8, 16.9, 17.0, 17.1, 17.2, 17.3, 17.4, 17.5, 17.6, 17.7, 17.8, 17.
9, 18.0, 18.1, 18.2, 18.3, 18.4, 18.5, 18.6, 18.7, 18.8, 18.9, 19.0, 19.1, 19.2, 1
9.3, 19.4, 19.5, 19.6, 19.7, 20.1, 20.2, 20.3, 20.4, 20.6, 20.7, 20.8, 20.9, 21.0,
21.1, 21.2, 21.3, 21.4, 21.5, 21.6, 21.7, 22.0, 22.1, 22.2, 22.3, 22.4, 22.5, 22.
6, 22.7, 22.8, 22.9, 23.0, 23.1, 23.2, 23.3, 23.4, 23.5, 23.6, 23.7, 23.8, 23.9, 2
4.0, 24.2, 24.3, 24.6, 24.7, 24.9, 25.0, 25.1, 25.2, 25.3, 25.4, 25.6, 25.8, 25.9,
26.0, 26.1, 26.2, 26.3, 26.4, 26.5, 26.6, 26.7, 26.8, 26.9, 27.1, 27.5, 27.6, 27.
7, 27.8, 27.9, 28.1, 28.2, 28.4, 28.6, 28.8, 29.1, 29.2, 29.3, 29.4, 29.5, 29.6, 2
9.8, 29.9, 30.0, 30.2, 30.6, 30.9, 31.0, 31.1, 31.3, 31.4, 31.5, 31.6, 31.7, 31.9,
32.1, 32.3, 32.4, 32.5, 32.6, 32.7, 32.8, 32.9, 33.0, 33.2, 33.3, 33.5, 33.6, 33.
7, 33.9, 34.0, 34.1, 34.2, 34.3, 34.5, 34.6, 34.8, 35.0, 35.2, 35.3, 35.4, 35.5, 3
5.8, 36.2, 36.3, 36.4, 36.5, 36.6, 36.7, 36.8, 36.9, 37.0, 37.1, 37.2, 37.3, 37.4,
37.5, 37.6, 37.7, 37.8, 37.9, 38.0, 38.1, 38.2, 38.3, 38.4, 38.5, 38.6, 38.7, 38.
8, 38.9, 39.0, 39.1, 39.2, 39.3, 39.4, 39.5, 39.6, 39.7, 39.8, 39.9, 40.0, 40.1, 4
0.2, 40.3, 40.4, 40.5, 40.6, 40.7, 40.8, 40.9, 41.0, 41.1, 41.2, 41.3, 41.4, 41.5,
41.6, 41.7, 41.8, 41.9, 42.0, 42.1, 42.2, 42.3, 42.4, 42.5, 42.6, 42.7, 42.8, 42.
9, 43.0, 43.1, 43.2, 43.3, 43.4, 43.5, 43.6, 43.7, 43.8, 43.9, 44.0, 44.1, 44.2, 4
4.3, 44.4, 44.5, 44.6, 44.7, 44.8, 44.9, 45.0, 45.1, 45.2, 45.3, 45.4, 45.5, 45.6,
45.7, 45.8, 45.9, 46.0, 46.1, 46.2, 46.3, 46.4, 46.5, 46.6, 46.7, 46.8, 46.9, 47.
0, 47.1, 47.2, 47.3, 47.4, 47.5, 47.6, 47.7, 47.8, 47.9, 48.0, 48.1, 48.2, 48.3, 4
8.4, 48.5, 48.6, 48.7, 48.8, 48.9, 49.0, 49.1, 49.2, 49.3, 49.4, 49.5, 49.6, 49.7,
49.8, 49.9, 50.0, 50.2, 51.4]

```
C:\Application\Python\Anaconda3\lib\site-packages\pandas\core\indexing.py:189: Set
tingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy
  self._setitem_with_indexer(indexer, value)
C:\Application\Python\Anaconda3\lib\site-packages\ipykernel_launcher.py:2: Setting
WithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy

C:\Application\Python\Anaconda3\lib\site-packages\ipykernel_launcher.py:3: Setting
WithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy
  This is separate from the ipykernel package so we can avoid doing imports until
```

In [48]:

```
# data cleaning for min > max situation
dfhematocrit[dfhematocrit['d1_hematocrit_max']<dfhematocrit['d1_hematocrit_min']]

# taking the maximum value of h1 and apache is reasonable
dfhematocrit.loc[(dfhematocrit.d1_hematocrit_max<dfhematocrit.d1_hematocrit_min) &
                 (dfhematocrit.hematocrit_apache.notnull()),'d1_hematocrit_max']=dfhematocrit[[
'hematocrit_apache','h1_hematocrit_max']].max(axis=1)
```

```
C:\Application\Python\Anaconda3\lib\site-packages\pandas\core\indexing.py:189: Set
tingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy
  self._setitem_with_indexer(indexer, value)
C:\Application\Python\Anaconda3\lib\site-packages\ipykernel_launcher.py:6: Setting
WithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy
```

In [49]:

```
dfhematocrit[dfhematocrit['d1_hematocrit_max']<dfhematocrit['d1_hematocrit_min']]
```

Out[49]:

	hematocrit_apache	d1_hematocrit_max	d1_hematocrit_min	h1_hematocrit_m
28935	37.1	37.1	43.2	NaN
65541	NaN	20.4	50.0	NaN

In [50]:

```
# look into the data to find reasonable imputation method
# Check the frequency for ever d1 min value
backup=dfhematocrit[(dfhematocrit['d1_hematocrit_min']==43.2) | (dfhematocrit['d1_hematocrit_mi
n']==50)]
backup=backup[backup['d1_hematocrit_max']>=43.2]

freq= backup.groupby('d1_hematocrit_min')['d1_hematocrit_max'].value_counts()
freq

# Due to too many values for each d1 min values, impute the average of each group
hemaglobinmean=hemaglobinimpute.groupby('d1_hemaglobin_min')['d1_hemaglobin_max'].mean()
# print(hemaglobinmean)
# check1=dfhemaglobin[dfhemaglobin['d1_hemaglobin_max']<dfhemaglobin['d1_hemaglobin_min']]

for i in ([43.2, 50.0]):
    imputation=backup.loc[backup['d1_hematocrit_min'] == i, 'd1_hematocrit_max'].mean()
    dfhematocrit.loc[(dfhematocrit.d1_hematocrit_max<dfhematocrit.d1_hematocrit_min) &
                     (dfhematocrit.d1_hematocrit_min==i),'d1_hematocrit_max']=round(imputation,1
)

# imputation check
# dfhematocrit[dfhematocrit.index.isin([28935,65541])]
```

```
C:\Application\Python\Anaconda3\lib\site-packages\pandas\core\indexing.py:189: Set
tingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy
  self._setitem_with_indexer(indexer, value)
C:\Application\Python\Anaconda3\lib\site-packages\ipykernel_launcher.py:17: Settin
gWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy
```

In [51]:

```
dfhematocrit[dfhematocrit.index.isin([28935,65541])]
```

Out[51]:

	hematocrit_apache	d1_hematocrit_max	d1_hematocrit_min	h1_hematocrit_m
28935	37.1	43.7	43.2	NaN
65541	NaN	51.2	50.0	NaN

In [52]:

```
# data cleaning for value 0 situation
dfhematocrit[dfhematocrit.d1_hematocrit_max==0]
dfhematocrit[dfhematocrit.d1_hematocrit_min==0]

# check all null after finishing data cleaning for hematocrit
dfhematocrit[(dfhematocrit.d1_hematocrit_max.isnull())& (dfhematocrit.h1_hematocrit_max.isnull
())&
          (dfhematocrit.d1_hematocrit_min.isnull())& (dfhematocrit.h1_hematocrit_min.isnull())&
          (dfhematocrit.hematocrit_apache.isnull())]
```

Out[52]:

	hematocrit_apache	d1_hematocrit_max	d1_hematocrit_min	h1_hematocrit_m
2	NaN	NaN	NaN	NaN
4	NaN	NaN	NaN	NaN
11	NaN	NaN	NaN	NaN
20	NaN	NaN	NaN	NaN
58	NaN	NaN	NaN	NaN
66	NaN	NaN	NaN	NaN
86	NaN	NaN	NaN	NaN
87	NaN	NaN	NaN	NaN
91	NaN	NaN	NaN	NaN
99	NaN	NaN	NaN	NaN
107	NaN	NaN	NaN	NaN
123	NaN	NaN	NaN	NaN
131	NaN	NaN	NaN	NaN
139	NaN	NaN	NaN	NaN
180	NaN	NaN	NaN	NaN
183	NaN	NaN	NaN	NaN
185	NaN	NaN	NaN	NaN
204	NaN	NaN	NaN	NaN
212	NaN	NaN	NaN	NaN
214	NaN	NaN	NaN	NaN
226	NaN	NaN	NaN	NaN
236	NaN	NaN	NaN	NaN
237	NaN	NaN	NaN	NaN
244	NaN	NaN	NaN	NaN
247	NaN	NaN	NaN	NaN
252	NaN	NaN	NaN	NaN
271	NaN	NaN	NaN	NaN
281	NaN	NaN	NaN	NaN
300	NaN	NaN	NaN	NaN
314	NaN	NaN	NaN	NaN
...	...	...	...	...
87249	NaN	NaN	NaN	NaN
87255	NaN	NaN	NaN	NaN

	hematocrit_apache	d1_hematocrit_max	d1_hematocrit_min	h1_hematocrit_m
87257	NaN	NaN	NaN	NaN
87267	NaN	NaN	NaN	NaN
87274	NaN	NaN	NaN	NaN
87279	NaN	NaN	NaN	NaN
87311	NaN	NaN	NaN	NaN
87314	NaN	NaN	NaN	NaN
87315	NaN	NaN	NaN	NaN
87329	NaN	NaN	NaN	NaN
87342	NaN	NaN	NaN	NaN
87374	NaN	NaN	NaN	NaN
87378	NaN	NaN	NaN	NaN
87382	NaN	NaN	NaN	NaN
87385	NaN	NaN	NaN	NaN
87387	NaN	NaN	NaN	NaN
87393	NaN	NaN	NaN	NaN
87394	NaN	NaN	NaN	NaN
87395	NaN	NaN	NaN	NaN
87407	NaN	NaN	NaN	NaN
87419	NaN	NaN	NaN	NaN
87422	NaN	NaN	NaN	NaN
87443	NaN	NaN	NaN	NaN
87449	NaN	NaN	NaN	NaN
87452	NaN	NaN	NaN	NaN
87459	NaN	NaN	NaN	NaN
87463	NaN	NaN	NaN	NaN
87465	NaN	NaN	NaN	NaN
87468	NaN	NaN	NaN	NaN
87479	NaN	NaN	NaN	NaN

10167 rows × 5 columns

inr

In [53]:

```
# variable: inr
dfinr=training.filter(regex='inr')

# check day values missing while hour values not
print(len(dfinr[(dfinr.h1_inr_max.notna()) & (dfinr.d1_inr_max.isna())]),
      len(dfinr[(dfinr.h1_inr_min.notna()) & (dfinr.d1_inr_min.isna())]))
```

0 0

In [54]:

```
# Check if missing values are in pairs for max and min
print(len(dfinr[(dfinr.d1_inr_max.notna()) & (dfinr.d1_inr_min.isna())]),
      len(dfinr[(dfinr.d1_inr_max.isna()) & (dfinr.d1_inr_min.notna())]),
      len(dfinr[dfinr.d1_inr_max.notna()]))
```

0 0 32398

In [55]:

```
# data cleaning for min > max situation
dfinr[dfinr['d1_inr_max']<dfinr['d1_inr_min']]

# data cleaning for value 0 situation
dfinr[dfinr.d1_inr_max==0]
dfinr[dfinr.d1_inr_min==0]
```

Out[55]:

	d1_inr_max	d1_inr_min	h1_inr_max	h1_inr_min

In [56]:

```
# check all null after finishing data cleaning for inr
len(dfinr[(dfinr.d1_inr_max.isnull())& (dfinr.h1_inr_max.isnull())&
          (dfinr.d1_inr_min.isnull())& (dfinr.h1_inr_min.isnull())])
```

Out[56]:

55087

calcium

In [57]:

```
# variable: calcium
dfcalcium=training.filter(regex='calcium')

# check day values missing while hour values not
print(len(dfcalcium[(dfcalcium.h1_calcium_max.notna()) & (dfcalcium.d1_calcium_max.isna())]),
      len(dfcalcium[(dfcalcium.h1_calcium_min.notna()) & (dfcalcium.d1_calcium_min.isna())]))

# Check if missing values are in pairs for max and min
print(len(dfcalcium[(dfcalcium.d1_calcium_max.notna()) & (dfcalcium.d1_calcium_min.isna())]),
      len(dfcalcium[(dfcalcium.d1_calcium_max.isna()) & (dfcalcium.d1_calcium_min.notna())]),
      len(dfcalcium[dfcalcium.d1_calcium_max.notna()]))

# list unique values in d1_max data to check for abnoramal data
print(sorted(dfcalcium.d1_calcium_max.unique()),
      sorted(dfcalcium.d1_calcium_min.unique()))
```

```
0 0
0 0 75091
[6.2, 6.3, 6.4, 6.5, 6.6, 6.7, 6.8, 6.9, 7.0, 7.1, 7.2, 7.3, 7.4, 7.5, 7.6, 7.7,
7.9, 8.5, 8.6, nan, 7.8, 8.0, 8.1, 8.2, 8.3, 8.4, 8.7, 8.8, 8.9, 9.0, 9.1, 9.2, 9.
3, 9.4, 9.5, 9.6, 9.7, 9.8, 9.9, 10.0, 10.1, 10.2, 10.3, 10.4, 10.5, 10.6, 10.7, 1
0.8] [5.5, 5.6, 5.7, 5.8, 5.9, 6.0, 6.1, 6.2, 6.3, 6.4, 6.5, 6.6, 6.7, 6.8, 6.9,
7.0, 7.1, 7.2, 7.3, 7.4, 7.5, 7.6, 7.7, 8.0, nan, 7.8, 7.9, 8.1, 8.2, 8.3, 8.4, 8.
5, 8.6, 8.7, 8.8, 8.9, 9.0, 9.1, 9.2, 9.3, 9.4, 9.5, 9.6, 9.7, 9.8, 9.9, 10.0, 10.
1, 10.2, 10.3]
```

In [58]:

```
# data cleaning for min > max situation
dfcalcium[dfcalcium['d1_calcium_max']<dfcalcium['d1_calcium_min']]

# where h1_calcium_max is taken as it is reasonable
dfcalcium.loc[(dfcalcium.d1_calcium_max<dfcalcium.d1_calcium_min) & (dfcalcium.h1_calcium_max.no
tnull()),'d1_calcium_max']=dfcalcium['h1_calcium_max']
```

```
C:\Application\Python\Anaconda3\lib\site-packages\pandas\core\indexing.py:189: Set
tingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy
  self._setitem_with_indexer(indexer, value)
C:\Application\Python\Anaconda3\lib\site-packages\ipykernel_launcher.py:5: Setting
WithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy
  """
```

In [59]:

```
# look into the data to find reasonable imputation method
backup=dfcalcium[dfcalcium['d1_calcium_max']<dfcalcium['d1_calcium_min']]
print(sorted(backup.d1_calcium_max.unique()),
      sorted(backup.d1_calcium_min.unique()))

calciumimpute=dfcalcium[(dfcalcium['d1_calcium_min']>=7.2) &
                        (dfcalcium['d1_calcium_min']<=10.3) &
                        (dfcalcium['d1_calcium_max']!=6.2)]
freq= calciumimpute.groupby('d1_calcium_min')['d1_calcium_max'].value_counts()

#calciumimpute.groupby('d1_calcium_min')['d1_calcium_max'].mean()
#check2=dfcalcium[dfcalcium['d1_calcium_max']<dfcalcium['d1_calcium_min']]

for i in (sorted(backup.d1_calcium_min.unique())):
    imputation=calciumimpute.loc[calciumimpute['d1_calcium_min'] == i, 'd1_calcium_max'].mean()
    dfcalcium.loc[(dfcalcium.d1_calcium_max<dfcalcium.d1_calcium_min) &
                  (dfcalcium.d1_calcium_min==i),'d1_calcium_max']=round(imputation,1)
```

```
[6.2] [7.2, 8.4, 8.5, 8.7, 8.9, 9.4, 9.6, 10.2, 10.3]

C:\Application\Python\Anaconda3\lib\site-packages\pandas\core\indexing.py:189: Set
tingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy
  self._setitem_with_indexer(indexer, value)
C:\Application\Python\Anaconda3\lib\site-packages\ipykernel_launcher.py:17: Settin
gWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy
```

In [60]:

```
# data cleaning for value 0 situation
dfcalcium[dfcalcium.d1_calcium_max==0]
dfcalcium[dfcalcium.d1_calcium_min==0]
```

Out[60]:

	d1_calcium_max	d1_calcium_min	h1_calcium_max	h1_calcium_min

In [61]:

```
# check all null after finishing data cleaning for calcium
len(dfcalcium[(dfcalcium.d1_calcium_max.isnull())& (dfcalcium.h1_calcium_max.isnull())&
         (dfcalcium.d1_calcium_min.isnull())& (dfcalcium.h1_calcium_min.isnull())])
```

Out[61]:

```
12394
```

PH

In [62]:

```
training = training.rename(columns={'ph_apache': 'arterial_ph_apache'})
arterial_ph=training.filter(regex='arterial_ph')
#impute h1 data once d1 data is null
arterial_ph.loc[arterial_ph['d1_arterial_ph_max'].isnull(),'d1_arterial_ph_max']=arterial_ph['h1
_arterial_ph_max']
arterial_ph.loc[arterial_ph['d1_arterial_ph_min'].isnull(),'d1_arterial_ph_min']=arterial_ph['h1
_arterial_ph_min']
# impute with apache covariate
arterial_ph.loc[arterial_ph['d1_arterial_ph_max'].isnull(),'d1_arterial_ph_max']=arterial_ph['ar
terial_ph_apache']
arterial_ph.loc[arterial_ph['d1_arterial_ph_min'].isnull(),'d1_arterial_ph_min']=arterial_ph['ar
terial_ph_apache']
```

```
C:\Application\Python\Anaconda3\lib\site-packages\pandas\core\indexing.py:189: Set
tingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy
  self._setitem_with_indexer(indexer, value)
C:\Application\Python\Anaconda3\lib\site-packages\ipykernel_launcher.py:4: Setting
WithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy
  after removing the cwd from sys.path.
C:\Application\Python\Anaconda3\lib\site-packages\ipykernel_launcher.py:5: Setting
WithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy
  """
C:\Application\Python\Anaconda3\lib\site-packages\ipykernel_launcher.py:7: Setting
WithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy
  import sys
C:\Application\Python\Anaconda3\lib\site-packages\ipykernel_launcher.py:8: Setting
WithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy
```

In [63]:

```
#check the values with min>max
arterial_ph[arterial_ph.d1_arterial_ph_max<arterial_ph.d1_arterial_ph_min].head(10)
#print(len(arterial_ph[arterial_ph.d1_arterial_ph_max<arterial_ph.d1_arterial_ph_min]))
##mpute the min>max data by h1 max since the max values are more likely to be wrong
arterial_ph.loc[(arterial_ph.d1_arterial_ph_max<arterial_ph.d1_arterial_ph_min) & (arterial_ph.h
1_arterial_ph_max.notnull()),'d1_arterial_ph_max']=arterial_ph['h1_arterial_ph_max']
#impute by apache data
arterial_ph.loc[(arterial_ph.d1_arterial_ph_max<arterial_ph.d1_arterial_ph_min) & (arterial_ph.a
rterial_ph_apache.notnull())&(arterial_ph.h1_arterial_ph_max.isnull()),'d1_arterial_ph_max']=art
erial_ph['arterial_ph_apache']
```

```
C:\Application\Python\Anaconda3\lib\site-packages\pandas\core\indexing.py:189: Set
tingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy
  self._setitem_with_indexer(indexer, value)
C:\Application\Python\Anaconda3\lib\site-packages\ipykernel_launcher.py:5: Setting
WithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy
  """
C:\Application\Python\Anaconda3\lib\site-packages\ipykernel_launcher.py:7: Setting
WithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy
  import sys
```

In [64]:

```
arterial_ph[arterial_ph['d1_arterial_ph_max']<arterial_ph['d1_arterial_ph_min']]
```

Out[64]:

	arterial_ph_apache	d1_arterial_ph_max	d1_arterial_ph_min	h1_arterial_ph_m
30921	NaN	7.05428	7.420	NaN
40773	NaN	7.05428	7.537	NaN
65476	NaN	7.05428	7.400	NaN

In [65]:

```
# look into the data to find reasonable imputation method
backup=arterial_ph[arterial_ph['d1_arterial_ph_max']<arterial_ph['d1_arterial_ph_min']]
print(sorted(backup.d1_arterial_ph_max.unique()),
      sorted(backup.d1_arterial_ph_min.unique()))

phimpute=arterial_ph[(arterial_ph['d1_arterial_ph_min']>=7.4) &
                        (arterial_ph['d1_arterial_ph_min']<=7.537) &
                        (arterial_ph['d1_arterial_ph_max']!=7.05428)]
freq= phimpute.groupby('d1_arterial_ph_min')['d1_arterial_ph_max'].value_counts()

# for i in (sorted(backup.d1_calcium_min.unique())):
#     imputation=calciumimpute.loc[calciumimpute['d1_calcium_min'] == i, 'd1_calcium_max'].mean
()
#     dfcalcium.loc[(dfcalcium.d1_calcium_max<dfcalcium.d1_calcium_min) &
#                   (dfcalcium.d1_calcium_min==i),'d1_calcium_max']=round(imputation,1)
```

```
[7.05428] [7.4, 7.42, 7.537000000000001]
```

In [66]:

```
freq
```

Out[66]:

d1_arterial_ph_min	d1_arterial_ph_max	
7.400	7.400	485
	7.440	39
	7.420	37
	7.430	37
	7.410	32
	7.450	31
	7.460	28
	7.470	17
	7.480	14
	7.490	12
	7.500	9
	7.540	6
	7.510	3
	7.520	3
	7.620	3
	7.515	2
	7.560	2
	7.600	2
	7.401	1
	7.407	1
	7.419	1
	7.426	1
	7.429	1
	7.432	1
	7.434	1
	7.446	1
	7.453	1
	7.455	1
	7.459	1
	7.472	1
		...
7.525	7.525	4
	7.527	1
7.526	7.526	9
	7.537	1
	7.541	1
	7.565	1
7.527	7.527	5
	7.538	1
	7.597	1
7.528	7.528	4
	7.532	1
7.529	7.529	3
	7.558	1
7.530	7.530	64
	7.620	5
	7.550	3
	7.560	2
	7.540	1
	7.543	1
	7.553	1
	7.600	1
7.531	7.531	2
7.532	7.532	2
7.533	7.533	3
	7.543	1
7.534	7.534	5
	7.602	1

```
7.535                7.535                    3
                     7.582                    1
7.536                7.536                    1
Name: d1_arterial_ph_max, Length: 1113, dtype: int64
```

In [67]:

```
training.head()
```

Out[67]:

	encounter_id	patient_id	hospital_id	hospital_death	age	bmi	elective_surgery
0	66154	25312	118	0	68	22.73	0
1	114252	59342	81	0	77	27.42	0
2	119783	50777	118	0	25	31.95	0
3	79267	46918	118	0	81	22.64	1
4	92056	34377	33	0	19	NaN	0

5 rows × 187 columns

In [68]:

```
# extract the data start with d and demographicinfo
train_id=training.filter(regex='id$', axis=1)
age=training.filter(regex='age', axis=1)
gender=training.filter(regex='gender', axis=1)
# train_d=training.filter(regex='^d1', axis=1)
# train_h=training.filter(regex='^h1',axis=1)
# resultd = pd.concat([train_id, train_d], axis=1, sort=False)
# resulth = pd.concat([train_id, train_h], axis=1, sort=False)
# resultdh = pd.concat([resultd, resulth], axis=1, sort=False)
```

In [69]:

```
dataclean1=pd.concat([age,gender,dfalbumin,
                      dfbilirubin,dfbun,dfcalcium,
                      dfcreatinine,dfglucose,dfhco3,
                      dfhemaglobin,dfhematocrit,dfinr],axis=1, sort=False)

dataclean1 = dataclean1[dataclean1.columns.drop(list(dataclean1.filter(regex='^h1', axis=1)))]
dataclean1 = dataclean1[dataclean1.columns.drop(list(dataclean1.filter(regex='apache', axis=1
)))]

dataclean1.filter(regex='^d1', axis=1).describe()
```

Out[69]:

	d1_albumin_max	d1_albumin_min	d1_bilirubin_max	d1_bilirubin_min	d1_b
count	41598.000000	41598.000000	37402.000000	37402.000000	78255
mean	2.974751	2.900731	1.135617	1.067007	25.346
std	0.669813	0.673202	2.143682	2.032086	20.378
min	1.200000	1.100000	0.100000	0.100000	4.000
25%	2.500000	2.400000	0.400000	0.400000	12.000
50%	3.000000	2.900000	0.600000	0.600000	19.000
75%	3.500000	3.400000	1.100000	1.000000	31.000
max	4.600000	4.500000	51.000000	51.000000	126.00

In [70]:

```
missing_clean=dataclean1.isnull().sum()/len(dataclean1)
missing_clean
```

Out[70]:

```
age                  0.000000
gender               0.000149
d1_albumin_max       0.524513
d1_albumin_min       0.524513
d1_bilirubin_max     0.572475
d1_bilirubin_min     0.572475
d1_bun_max           0.105504
d1_bun_min           0.105504
d1_calcium_max       0.141670
d1_calcium_min       0.141670
d1_creatinine_max    0.101480
d1_creatinine_min    0.101480
d1_glucose_max       0.058410
d1_glucose_min       0.058410
d1_hco3_max          0.162599
d1_hco3_min          0.162599
d1_hemaglobin_max    0.131668
d1_hemaglobin_min    0.131668
d1_hematocrit_max    0.116214
d1_hematocrit_min    0.116214
d1_inr_max           0.629674
d1_inr_min           0.629674
dtype: float64
```

Data bining

In [71]:

```
# albumin
def albumin(x):
    # CF: 10. CF= conversion factor.
    x = x * 10
    if x <20:
        return 11
    if x<25:
        return 6
    if x<45:
        return 0
    if x>=45:
        return 4
    return 'Unknown'

def albumin1(x):
    # CF: 10. CF= conversion factor.
    x = x * 10
    if x <20:
        return 0
    if x<25:
        return 1
    if x<45:
        return 2
    if x>=45:
        return 3
    return 'Unknown'


# bilirubin
def bilirubin(x):
    # To convert results from mg/dL to µmol/L, multiply mg/dL by 17.1
    x = x * 17.1
    if x<2:
        return 0
    if x<3:
        return 5
    if x<5:
        return 6
    if x<8:
        return 8
    if x>=8:
        return 16
    return 'Unknown'

def bilirubin1(x):
    # To convert results from mg/dL to µmol/L, multiply mg/dL by 17.1
    x = x * 17.1
    if x<2:
        return 0
    if x<3:
        return 1
    if x<5:
        return 2
    if x<8:
        return 3
    if x>=8:
        return 4
    return 'Unknown'
```

```
# bun
def bun(x):
    if x<=17:
        return 0
    if x<20:
        return 2
    if x<40:
        return 7
    if x<80:
        return 11
    if x>=80:
        return 12
    return 'Unknown'

def bun1(x):
    if x<=17:
        return 0
    if x<20:
        return 1
    if x<40:
        return 2
    if x<80:
        return 3
    if x>=80:
        return 4
    return 'Unknown'


# calcium: Chapter 143Serum Calcium
# https://www.ncbi.nlm.nih.gov/books/NBK250/
# def calcium(x):
#     return 'Unknown'


# creatinine
def creatinine(x):
    if x<0.5:
        return 3
    if x<1.5:
        return 0
    if x<1.95:
        return 4
    if x>=1.95:
        return 7
    return 'Unknown'

# creatinine
def creatinine1(x):
    if x<0.5:
        return 0
    if x<1.5:
        return 1
    if x<1.95:
        return 2
    if x>=1.95:
        return 3
    return 'Unknown'


# apache 3
```

```
# glucose
# Conversion Formula for Converting mmol/L to mg/dl: Multiply mmol/L by the number 18.
def glucose(x):
    x=x/18.
    if x<=2.1:
        return 8
    if x<=3.3:
        return 5
    if x<=11.1:
        return 0
    if x<19.4:
        return 3
    if x>=19.4:
        return 5
    return 'Unknown'

def glucose1(x):
    x=x/18.
    if x<=2.1:
        return 4
    if x<=3.3:
        return 3
    if x<=11.1:
        return 2
    if x<19.4:
        return 1
    if x>=19.4:
        return 0
    return 'Unknown'


# hco3
# risk scale not found (apache 2)
def hco3(x):
    if x>=22 and x<=31.9:
        return 0
    if x>31.9 and x<=40.9:
        return 1
    if x>=18 and x<22:
        return 2
    if x>=15 and x<=52:
        return 3
    if x<15 or x>=52:
        return 4
    return 'Unknown'

def hco31(x):
    if x>=52:
        return 0
    if x>=41:
        return 1
    if x>=32:
        return 2
    if x>=22:
        return 3
    if x>=18:
        return 4
    if x>=15:
        return 5
    if x<15:
        return 6
```

```
    return 'Unknown'


# dfhemoglobin
# reference: Chapter 151Hemoglobin and Hematocrit
# https://www.ncbi.nlm.nih.gov/books/NBK259/
# def hemaglobin(x):
#     if x>=12 and x<=18:
#         return 0
#     if x<12 or x>18:
#         return 1
#     return 'Unknown'


# hematocrit
def hematocrit(x):
    if x>=41 and x<50:
        return 0
    if x<41 or x>=50:
        return 3
    return 'Unknown'

# hematocrit
def hematocrit1(x):
    if x<41:
        return 0
    if x<50:
        return 1
    if x>=50:
        return 2
    return 'Unknown'

# # dfinr
# # https://www.vaughns-1-pagers.com/medicine/blood-INR-range-chart.htm
# def inr(x):
#     if x<=1.2:   # min is 0.9, normal range 0.8-1.2
#         return 0
#     if x<=2:
#         return 1
#     if x<=3:
#         return 2
#     if x<=3.5:
#         return 3
#     if x<=4.5:
#         return 4
#     if x>4.5:
#         return 5
#     return 'Unknown'
```

In [72]:

```
# create new columns bin respectively
dataclean1['alb_max_bin']=dataclean1.d1_albumin_max.apply(lambda x: albumin1(x))
dataclean1['alb_min_bin']=dataclean1.d1_albumin_min.apply(lambda x: albumin1(x))

dataclean1['bil_max_bin']=dataclean1.d1_bilirubin_max.apply(lambda x: bilirubin1(x))
dataclean1['bil_min_bin']=dataclean1.d1_bilirubin_min.apply(lambda x: bilirubin1(x))

dataclean1['bun_max_bin']=dataclean1.d1_bun_max.apply(lambda x: bun1(x))
dataclean1['bun_min_bin']=dataclean1.d1_bun_min.apply(lambda x: bun1(x))

dataclean1['cre_max_bin']=dataclean1.d1_creatinine_max.apply(lambda x: creatinine1(x))
dataclean1['cre_min_bin']=dataclean1.d1_creatinine_min.apply(lambda x: creatinine1(x))

dataclean1['glu_max_bin']=dataclean1.d1_glucose_max.apply(lambda x: glucose1(x))
dataclean1['glu_min_bin']=dataclean1.d1_glucose_min.apply(lambda x: glucose1(x))

dataclean1['hco3_max_bin']=dataclean1.d1_hco3_max.apply(lambda x: hco31(x))
dataclean1['hco3_min_bin']=dataclean1.d1_hco3_min.apply(lambda x: hco31(x))

dataclean1['hto_max_bin']=dataclean1.d1_hematocrit_max.apply(lambda x: hematocrit1(x))
dataclean1['hto_min_bin']=dataclean1.d1_hematocrit_min.apply(lambda x: hematocrit1(x))

# dataclean1['inr_max_bin']=dataclean1.d1_inr_max.apply(lambda x: inr(x))
# dataclean1['inr_min_bin']=dataclean1.d1_inr_min.apply(lambda x: inr(x))

# create new columns risk respectively
dataclean1['alb_max_risk']=dataclean1.d1_albumin_max.apply(lambda x: albumin(x))
dataclean1['alb_min_risk']=dataclean1.d1_albumin_min.apply(lambda x: albumin(x))

dataclean1['bil_max_risk']=dataclean1.d1_bilirubin_max.apply(lambda x: bilirubin(x))
dataclean1['bil_min_risk']=dataclean1.d1_bilirubin_min.apply(lambda x: bilirubin(x))

dataclean1['bun_max_risk']=dataclean1.d1_bun_max.apply(lambda x: bun(x))
dataclean1['bun_min_risk']=dataclean1.d1_bun_min.apply(lambda x: bun(x))

dataclean1['cre_max_risk']=dataclean1.d1_creatinine_max.apply(lambda x: creatinine(x))
dataclean1['cre_min_risk']=dataclean1.d1_creatinine_min.apply(lambda x: creatinine(x))

dataclean1['glu_max_risk']=dataclean1.d1_glucose_max.apply(lambda x: glucose(x))
dataclean1['glu_min_risk']=dataclean1.d1_glucose_min.apply(lambda x: glucose(x))

dataclean1['hco3_max_risk']=dataclean1.d1_hco3_max.apply(lambda x: hco3(x))
dataclean1['hco3_min_risk']=dataclean1.d1_hco3_min.apply(lambda x: hco3(x))

dataclean1['hto_max_risk']=dataclean1.d1_hematocrit_max.apply(lambda x: hematocrit(x))
dataclean1['hto_min_risk']=dataclean1.d1_hematocrit_min.apply(lambda x: hematocrit(x))

# dataclean1['inr_max_bin']=dataclean1.d1_inr_max.apply(lambda x: inr(x))
# dataclean1['inr_min_bin']=dataclean1.d1_inr_min.apply(lambda x: inr(x))
```

In [73]:

```
# add change columns
def changefunc(x,y):
    if x=='Unknown' and y=='Unknown':
        return 'Unknown'
    if x==y:
        return 0
    if x!=y:
        return 1
    return 'Error'


column_initials=['alb','bil','bun','cre','glu','hco3','hto']
for column_initials in column_initials:
    max = column_initials+'_max'+'_bin'
    min = column_initials+'_min'+'_bin'
    list=[]
    for i in range(0,len(dataclean1)):
        value=changefunc(dataclean1[max][i],dataclean1[min][i])
        list.append(value)
    dataclean1.loc[:,column_initials+'_change']=list


# column_initials=['alb','bil','bun','cre','glu','hco3','hto']
# for i in column_initials:
#     print(change[i+'_change'].unique())
```

In [74]:

```
# add final bin columns
column_initials=['alb','bil','bun','cre','glu','hco3','hto']
for column_initials in column_initials:
    maxbin = column_initials+'_max'+'_bin'
    minbin = column_initials+'_min'+'_bin'
    maxrisk = column_initials+'_max'+'_risk'
    minrisk = column_initials+'_min'+'_risk'
    list=[]
    for i in range(0,len(dataclean1)):
        if dataclean1[maxrisk][i]=='Unknown' and dataclean1[minrisk][i]=='Unknown':
            value='Unknown'
        if dataclean1[maxrisk][i]>=dataclean1[minrisk][i]:
            value=dataclean1[maxbin][i]
        if dataclean1[maxrisk][i]<dataclean1[minrisk][i]:
            value=dataclean1[minbin][i]
        list.append(value)
    print(len(list))
    dataclean1.loc[:,column_initials+'_final']=list
```

```
87485
87485
87485
87485
87485
87485
87485
```

Numeric Data

In [75]:

```
# calcium: Chapter 143Serum Calcium
# https://www.ncbi.nlm.nih.gov/books/NBK250/
# normal average is 9.5

calmaxmean=dataclean1.d1_calcium_max.mean()
calmaxstd=dataclean1.d1_calcium_max.std()
dataclean1['tempo_calcium_max']= abs((dataclean1['d1_calcium_max'] - calmaxmean)/calmaxstd-9.5)

calminmean=dataclean1.d1_calcium_min.mean()
calminstd=dataclean1.d1_calcium_min.std()
dataclean1['tempo_calcium_min']= abs((dataclean1['d1_calcium_min'] - calminmean)/calminstd-9.5)



# dfhemoglobin
# reference: Chapter 151Hemoglobin and Hematocrit
# https://www.ncbi.nlm.nih.gov/books/NBK259/
# normal average is 15

hemoglobinmaxmean=dataclean1.d1_hemaglobin_max.mean()
hemoglobinmaxstd=dataclean1.d1_hemaglobin_max.std()
dataclean1['tempo_hemaglobin_max']= abs((dataclean1['d1_hemaglobin_max'] - hemoglobinmaxmean)/he
moglobinmaxstd-15)

hemoglobinminmean=dataclean1.d1_hemaglobin_min.mean()
hemoglobinminstd=dataclean1.d1_hemaglobin_min.std()
dataclean1['tempo_hemaglobin_min']= abs((dataclean1['d1_hemaglobin_min'] - hemoglobinminmean)/he
moglobinminstd-15)



# # dfinr
# # https://www.vaughns-1-pagers.com/medicine/blood-INR-range-chart.htm
#     if x<=1.2:   # min is 0.9, normal range 0.8-1.2

inrmaxmean=dataclean1.d1_inr_max.mean()
inrmaxstd=dataclean1.d1_inr_max.std()
dataclean1['tempo_inr_max']= abs((dataclean1['d1_inr_max'] - inrmaxmean)/inrmaxstd-1)

inrminmean=dataclean1.d1_inr_min.mean()
inrminstd=dataclean1.d1_inr_min.std()
dataclean1['tempo_inr_min']= abs((dataclean1['d1_inr_min'] - inrminmean)/inrminstd-1)
```

In [91]:

```
# take the larger values with more risk
dataclean1 = dataclean1.fillna('Unknown')

column_names=['calcium','hemaglobin','inr']
for column_names in column_names:
    tempomax = 'tempo_'+column_names+'_max'
    tempomin = 'tempo_'+column_names+'_min'
    finalmax = 'd1_'+column_names+'_max'
    finalmin = 'd1_'+column_names+'_min'
    list=[]
    for i in range(0,len(dataclean1)):
        if dataclean1[tempomax][i]=='Unknown' and dataclean1[tempomin][i]=='Unknown':
            value='Unknown'
        if dataclean1[tempomax][i]>=dataclean1[tempomin][i]:
            value=dataclean1[finalmax][i]
        if dataclean1[tempomax][i]<dataclean1[tempomin][i]:
            value=dataclean1[finalmin][i]
        list.append(value)
    print(len(list))
    dataclean1.loc[:,column_names+'_final']=list
```

```
87485
87485
87485
```

In [95]:

```
# extract the data start with d and h and id info
train_id=training.filter(regex='id$', axis=1)
train_d=dataclean1.filter(regex='^d1', axis=1)
train_bin=dataclean1.filter(regex='_bin',axis=1)
train_risk=dataclean1.filter(regex='risk',axis=1)
train_change=dataclean1.filter(regex='change',axis=1)
train_final=dataclean1.filter(regex='final',axis=1)
```

In [97]:

```
train_xjt = pd.concat([train_id,train_final,train_change,train_d,], axis=1, sort=False)
#train_xjt = pd.concat([train_risk,train_final,train_bin], axis=1, sort=False)
#train_xjt.filter(regex='alb',axis=1)
train_xjt.to_csv(r'C:\Users\DELL\Desktop\sc\train_cleaned_xjt.csv', index = False)
```