智能数据挖掘作业 3

19200300004 黄铭瑞

一、 实验目的

知道数据预处理的原理，掌握数据预处理的方法。重点掌握如插值、去噪、缺失值、离群值处理方法。

二、 实验原理

数据预处理一般包含数据清洗，数据转换，数据描述，特征选择或特征组合，特征抽取这五个步骤。

数据清洗阶段，主要是处理缺失值，重复值，离群值。

缺失值可以通过删除数据或者比较相邻值来填补数据。

重复值一般采用删除法来处理但有些重复值不能删除，例如订单明细数据或交易明细数据等。

对离群值的甄别，一般使用 3σ 准则或者箱型图，如果样本是正态分布或近似正态分布，可以考虑使用 3σ 方法，认为 99%以上的数据集中在均值上下 3 个标准差的范围内；箱型图一般用于统计数据分散情况而做的，超出上下限的值定位离群值。

三、 实验过程

1. 导入数据集

通过 pd.read_csv() 来导入 mini_chart_events.csv 与 mini_label_events.csv 文件，并把他们转为 N 维数组的形式。

	ROW_ID	SUBJECT_ID	HADM_ID	ICUSTAY_ID	ITEMID	CHARTTIME	STORETIME	CGID	VALUE	VALUENUM	VALUEUOM	WARNING	ERROR
0	304858	3379	142774	238507.000	220046	2162-05-27 20:00:00	2162-05-27 20:48:00	17574.000	120.000	120.000	bpm	0.000	0.000
1	304859	3379	142774	238507.000	220047	2162-05-27 20:00:00	2162-05-27 20:48:00	17574.000	60.000	60.000	bpm	0.000	0.000
2	304860	3379	142774	238507.000	223751	2162-05-27 20:00:00	2162-05-27 20:49:00	17574.000	160.000	160.000	mmHg	0.000	0.000
3	304861	3379	142774	238507.000	223752	2162-05-27 20:00:00	2162-05-27 20:49:00	17574.000	90.000	90.000	mmHg	0.000	0.000
4	304862	3379	142774	238507.000	223761	2162-05-27 20:00:00	2162-05-27 20:49:00	17574.000	99.400	99.400	?F	0.000	0.000
...	...	...	...	...	...	...	...	...	...	...	...	...	...
3388374	330605881	99863	100749	216757.000	224370	2142-04-26 21:00:00	2142-04-26 21:17:00	16351.000	White	NaN	NaN	0.000	0.000
3388375	330605882	99863	100749	216757.000	224372	2142-04-26 21:00:00	2142-04-26 21:17:00	16351.000	Suctioned	NaN	NaN	0.000	0.000
3388376	330605883	99863	100749	216757.000	224373	2142-04-26 21:00:00	2142-04-26 21:17:00	16351.000	Scant	NaN	NaN	0.000	0.000
3388377	330605884	99863	100749	216757.000	224642	2142-04-26 21:00:00	2142-04-26 21:43:00	20061.000	Oral	NaN	NaN	0.000	0.000
3388378	330605885	99863	100749	216757.000	224650	2142-04-26 21:00:00	2142-04-26 21:12:00	20061.000	None	NaN	NaN	0.000	0.000

3388379 rows × 15 columns

（df1,mini_chart_events.csv 原始数据）

	ROW_ID	SUBJECT_ID	HADM_ID	ITEMID	CHARTTIME	VALUE	VALUENUM	VALUEUOM	FLAG
0	71572	127	NaN	50861	2183-08-08 13:23:00	14	14.000	IU/L	NaN
1	71573	127	NaN	50862	2183-08-08 13:23:00	3.4	3.400	g/dL	abnormal
2	71574	127	NaN	50863	2183-08-08 13:23:00	91	91.000	IU/L	NaN
3	71575	127	NaN	50878	2183-08-08 13:23:00	24	24.000	IU/L	NaN
4	71576	127	NaN	50893	2183-08-08 13:23:00	9.1	9.100	mg/dL	NaN
...	...	...	...	...	...	...	...	...	...
290660	27777211	99085	147862.000	51200	2130-06-11 05:15:00	0.1	0.100	%	NaN
290661	27777212	99085	147862.000	51221	2130-06-11 05:15:00	37.2	37.200	%	NaN
290662	27777213	99085	147862.000	51222	2130-06-11 05:15:00	12.7	12.700	g/dL	NaN
290663	27777214	99085	147862.000	51237	2130-06-11 05:15:00	1.2	1.200	NaN	abnormal
290664	27777215	99085	147862.000	51244	2130-06-11 05:15:00	16.7	16.700	%	abnormal

290665 rows × 9 columns

（df2,mini_label_events.csv 原始数据）

```
df1_narray
executed in 18ms, finished 22:06:10 2022-05-02
array([[304858, 3379, 142774, ..., 0.0, nan, nan],
       [304859, 3379, 142774, ..., 0.0, nan, nan],
       [304860, 3379, 142774, ..., 0.0, nan, nan],
       ...,
       [330605883, 99863, 100749, ..., 0.0, nan, nan],
       [330605884, 99863, 100749, ..., 0.0, nan, nan],
       [330605885, 99863, 100749, ..., 0.0, nan, nan]], dtype=object)
```

（df1_narray,ndarray 类型的数据）

```
df2_narray
executed in 16ms, finished 22:06:39 2022-05-02
array([[71572, 127, nan, ..., 14.0, 'IU/L', nan],
       [71573, 127, nan, ..., 3.4, 'g/dL', 'abnormal'],
       [71574, 127, nan, ..., 91.0, 'IU/L', nan],
       ...,
       [27777213, 99085, 147862.0, ..., 12.7, 'g/dL', nan],
       [27777214, 99085, 147862.0, ..., 1.2, nan, 'abnormal'],
       [27777215, 99085, 147862.0, ..., 16.7, '%', 'abnormal']],
      dtype=object)
```

（df2_narray,ndarray 类型的 label events）

2. 提取出需要的 p02,pC02 数据。

依次把两个文件里的 ITEMID 为[490, 3785, 3837, 50821]之一，[3784, 3835, 50818]之一的数据提取出来，并放入一个空列表中。

```
[[907, '2155-08-21 19:00:00', '257.20001220703125', ''],
 [946, '2120-05-05 05:00:00', '308.5', ''],
 [946, '2120-05-10 10:59:00', '247.19999694824219', ''],
 [4033, '2159-06-15 01:00:00', '', '58'],
 [4033, '2159-06-15 01:00:00', '50', ''],
 [4033, '2159-06-20 17:00:00', '', nan],
 [4033, '2159-06-20 17:00:00', nan, ''],
 [4367, '2120-09-30 16:00:00', '657', ''],
 [6843, '2142-06-15 03:15:00', '', '40'],
 [6843, '2142-06-14 12:00:00', '', '45'],
 [6843, '2142-06-14 12:00:00', '73', ''],
 [6843, '2142-06-14 11:50:00', '', '45'],
 [6843, '2142-06-14 11:50:00', '73', ''],
 [6843, '2142-06-14 17:00:00', '', '22'],
 [6843, '2142-06-14 17:00:00', '82', ''],
 [6843, '2142-06-14 17:07:00', '', '22'],
 [6843, '2142-06-14 17:07:00', '82', ''],
 [6843, '2142-06-18 14:00:00', '', nan],
 [6843, '2142-06-18 14:00:00', nan, ''],
```

（o2co2data,提取出有用数据的列表）

3. 数据变换，保存数据

把 pO2,pCO2 作为其所含值的列名，构造新的列表。List 转为 dataframe 并以 csv 形式储存到本地。

```
[[907, '2155-08-21 19:00:00', '257.20001220703125', ''],
 [946, '2120-05-05 05:00:00', '308.5', ''],
 [946, '2120-05-10 10:59:00', '247.19999694824219', ''],
 [4033, '2159-06-15 01:00:00', '50', '58'],
 [4033, '2159-06-15 01:00:00', '50', '58'],
 [4033, '2159-06-20 17:00:00', nan, nan],
 [4033, '2159-06-20 17:00:00', nan, nan],
 [4367, '2120-09-30 16:00:00', '657', ''],
 [6843, '2142-06-15 03:15:00', '88', '40'],
 [6843, '2142-06-14 12:00:00', '73', '45'],
 [6843, '2142-06-14 12:00:00', '73', '45'],
 [6843, '2142-06-14 11:50:00', '73', '45'],
 [6843, '2142-06-14 11:50:00', '73', '45'],
 [6843, '2142-06-14 17:00:00', '82', '22'],
 [6843, '2142-06-14 17:00:00', '82', '22'],
 [6843, '2142-06-14 17:07:00', '82', '22'],
 [6843, '2142-06-14 17:07:00', '82', '22'],
 [6843, '2142-06-18 14:00:00', nan, nan],
 [6843, '2142-06-18 14:00:00', nan, nan],
```

（o2co2data，各列表示为 subject_id,charttime,pO2, pCO2）

4. 读取新保存的 csv 文件

	subject_id	charttime	pO2	pCO2
0	907	2155-08-21 19:00:00	257.20001220703125	28
1	946	2120-05-05 05:00:00	308.5	48
2	946	2120-05-10 10:59:00	247.19999694824219	38
3	4033	2159-06-15 01:00:00	50	58
4	4033	2159-06-15 01:00:00	50	58
...	...	...	...	...
9608	99085	2130-12-08 14:19:00	180	42
9609	99085	2130-12-14 20:22:00	69	46
9610	99085	2130-12-14 20:22:00	69	46
9611	99085	2131-02-08 21:04:00	68	47
9612	99085	2131-02-08 21:04:00	68	47

（df,o2co2data.csv 数据文件）

5. 查找并处理重复值

```
df.duplicated(keep='first')
```

executed in 22ms, finished 22:38:02 2022-05-02

```
0       False
1       False
2       False
3       False
4        True
        ...
9608     True
9609    False
9610     True
9611    False
9612     True
Length: 9613, dtype: bool
```

```
df = df.drop_duplicates()
```

executed in 15ms, finished 22:38:04 2022-05-02

6. 查找并处理空缺值

```
df.isnull().sum()
```

executed in 13ms, finished 22:40:07 2022-05-02

```
subject_id     0
charttime      0
pO2           15
pCO2          15
dtype: int64
```

```
df = df.dropna()
```

executed in 9ms, finished 20:55:01 2022-05-02

7. 数据类型转换

把 pO2,pCO2 数据转为 float 类型

```
df.info()
```

executed in 25ms, finished 22:42:22 2022-05-02

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 4730 entries, 0 to 9611
Data columns (total 4 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   subject_id  4730 non-null   int64 
 1   charttime   4730 non-null   object
 2   pO2         4715 non-null   object
 3   pCO2        4715 non-null   object
dtypes: int64(1), object(3)
memory usage: 184.8+ KB
```

（df 原始数据类型）

```
df["pO2"] = pd.to_numeric(df["pO2"],errors='coerce')
df["pCO2"] = pd.to_numeric(df["pCO2"],errors='coerce')
df.info()
```

executed in 34ms, finished 22:42:32 2022-05-02

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 4730 entries, 0 to 9611
Data columns (total 4 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   subject_id  4730 non-null   int64
 1   charttime   4730 non-null   object
 2   pO2         4714 non-null   float64
 3   pCO2        4714 non-null   float64
dtypes: float64(2), int64(1), object(1)
memory usage: 184.8+ KB
```

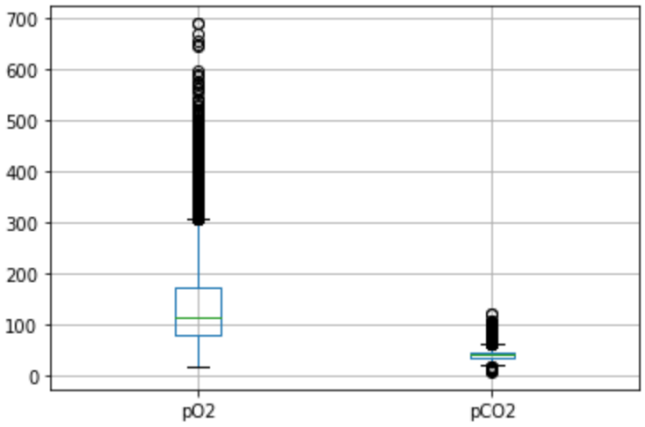
（转换后的 df 数据类型）

8. 查找离群值

df.describe()查看最大最小值是否是异常值。

	subject_id	pO2	pCO2
count	4712.000	4711.000	4711.000
mean	27821.661	146.186	42.435
std	25809.511	102.330	10.513
min	127.000	19.000	8.000
25%	9978.000	81.000	36.000
50%	20280.000	114.000	41.000
75%	32476.000	171.500	47.000
max	99863.000	689.000	121.000

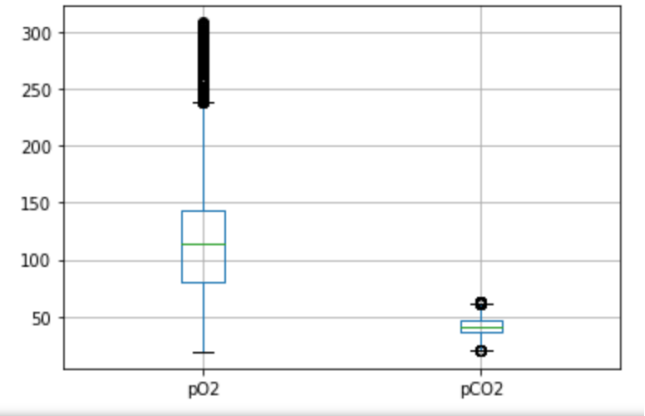
绘制 pO2,pCO2 两列数据的箱型图，列出离群数据。



（pO2 和 pCO2 的箱型图）

9. 离群值的处理

用平均值代替离群值，做出新的箱型图



（平均值代替离群值后的箱型图）

df.describe(),看出离群值平滑了很多

	subject_id	pO2	pCO2
count	4712.000	4711.000	4711.000
mean	27821.661	118.803	41.148
std	25809.511	55.685	7.695
min	127.000	19.000	20.000
25%	9978.000	81.000	36.000
50%	20280.000	114.000	41.000
75%	32476.000	144.000	46.000
max	99863.000	307.000	63.000

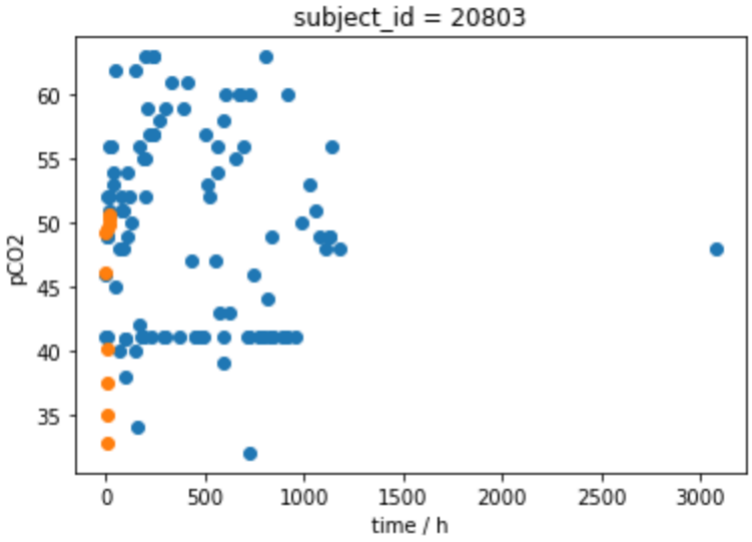
10.时间换算

把 charttime 换成小时的形式，并以小时作为时间轴，第一次采集时间为 0 小时。

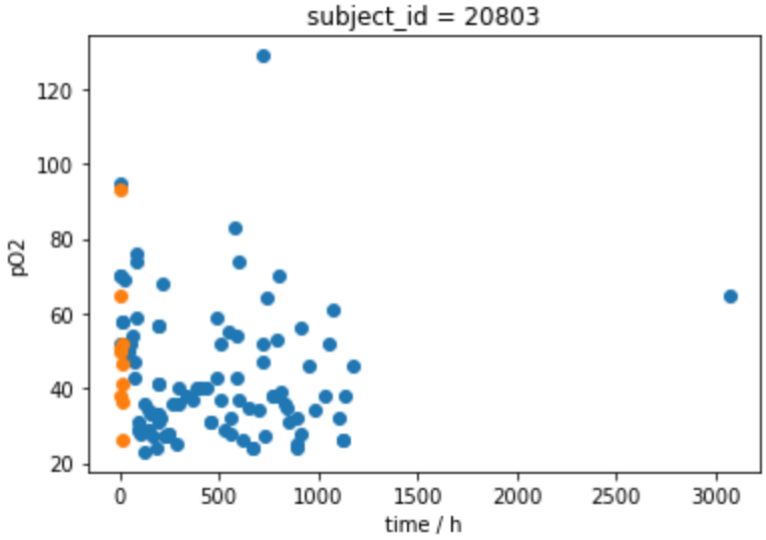
11.插值计算

选取三个不同的病人，分别对他们的 pO2,pCO2 进行拉格朗日插值计算，计算前 10 个插值。

病人 1：

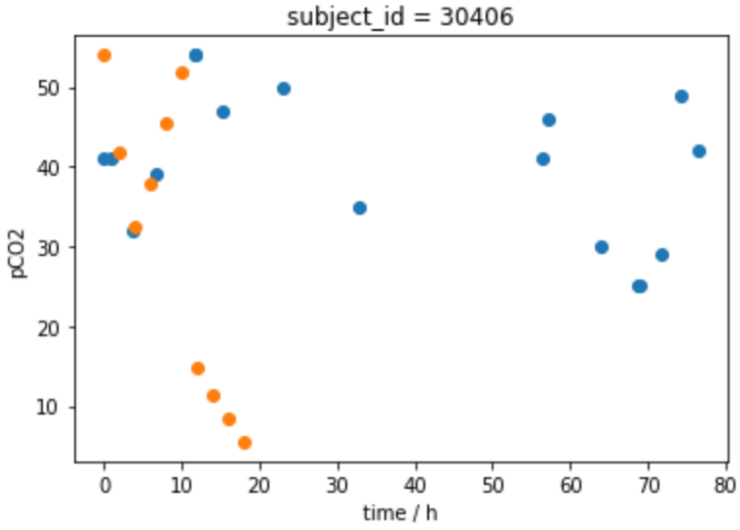


（病人 1 的 pCO2 插值）

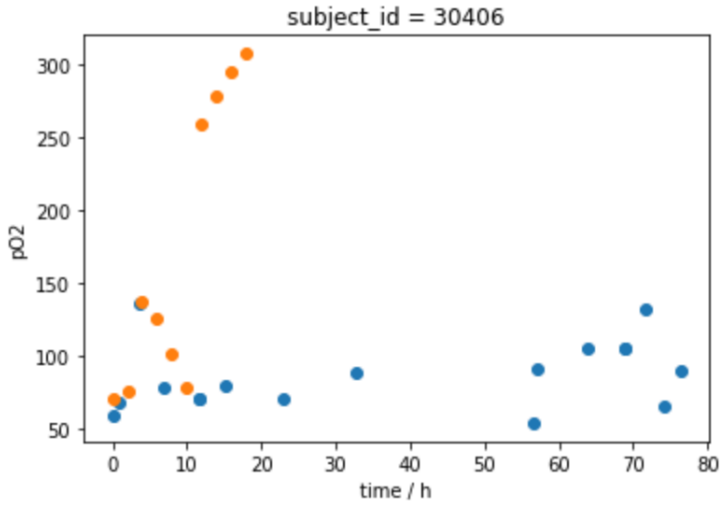


（病人 1 的 pO2 插值）

病人 2：

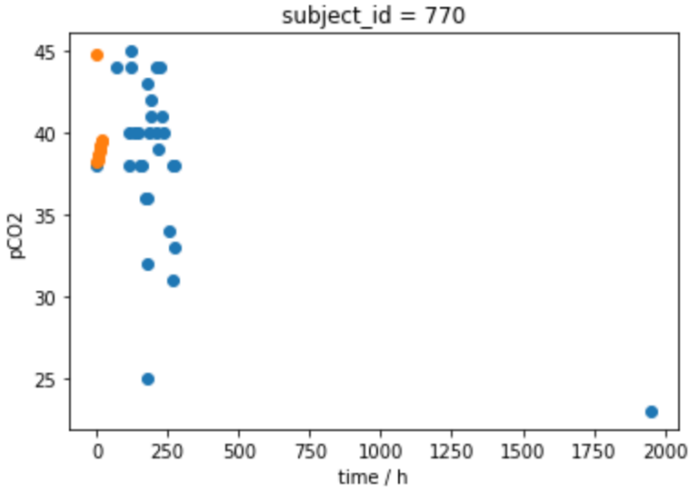


（病人 2 的 pCO2 插值）

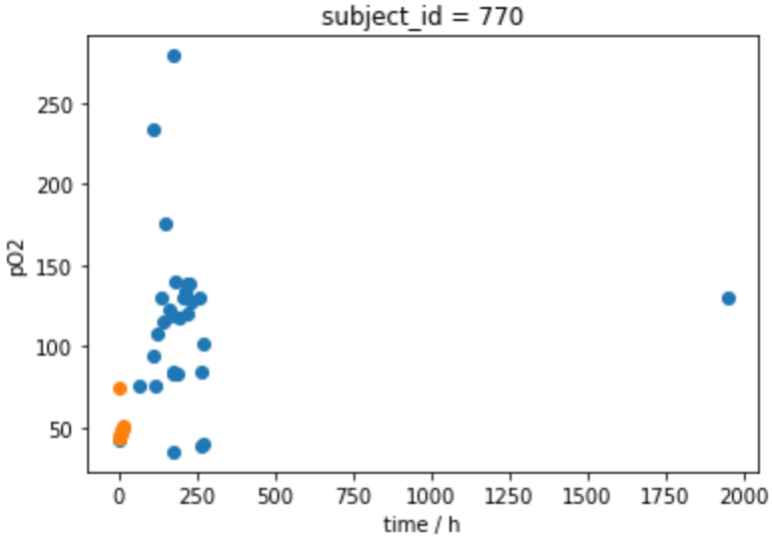


（病人 2 的 pO2 插值）

病人 3：



（病人 3 的 pCO2 插值）



（病人 3 的 pO2 插值）

四、 实验结果与分析

在对 pCO2 和 pO2 进行数据提取的时候，其中一个有值另一个必为空，在对他们两进行合并的时候就会出现大量的重复值，所以采用删除的方式进行去重。

在去重，去缺失值之后，数据中还存在大量的离群值，在对 pO2 和 pCO2 统一数据类型后，对他们的离群值平滑处理，采用均值替代的方式，大大的缩小了极差。

五、 附录

MIMIC 数据预处理.html