Data_processing

Huakun Huang
University of Aizu, Japan
huanghuakun13@gmail.com

Huawei Huang
Sun Yat-Sen University, China
huanghw28@mail.sysu.edu.cn

January 29, 2020

In [1]:
```
import numpy as np
import pandas as pd

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
```

Data Preprocessing

In [2]:
```
# # preprocess 1
# print('df1.columns:','\n', df1.columns) # take a look at columns, find some useless

df = pd.read_csv('./Raw_data.csv')

# # remove date information because of its uselessness
columns_contains_no_useful = [
    column for column in df.columns if 'user' not in column and 'info' not in column]

df = df[columns_contains_no_useful]
print('df.columns:','\n', df.columns) # take a look again
print('df:',df.head(3))
```

```
df.columns:
 Index(['time', 'job ID', 'task index', 'machine ID', 'event type',
       'scheduling class', 'priority', 'CPU request', 'memory request',
       'disk space request', 'different machines restriction'],
      dtype='object')
df:          time      job ID  task index   machine ID  event type  \
0  5611824441  6251812952        1761    1306108.0           4
1  5611824625  6251974185         252   38676295.0           4
2  5612587437  6251995937          33  386450501.0           2

   scheduling class  priority  CPU request  memory request  \
0                 0         2      0.02499         0.07959
1                 0         2      0.02499         0.03339
2                 0         0      0.06873         0.01193

   disk space request  different machines restriction
0            0.000386                               1
```

```
1            0.000386                                1
2            0.000115                                0
```

```
In [3]: print(df['event type'].value_counts())
        # 0: submitted
        # 1: scheduled
        # 2: evict
        # 3: fail
        # 4: finish
        # 5: kill

        df['event type'] = df['event type'].apply(lambda s: 1 if s == 2 or s == 3 or s ==5 else
        # 0 indicate not failure
        # 1 indicate unsuccessful

        # take a look at the label
        print(df['event type'].value_counts())
```

```
1    26231
0    25822
4    14551
5     8171
2     2500
3      501
Name: event type, dtype: int64
0    66604
1    11172
Name: event type, dtype: int64
```

```
In [4]: # To balance the data, by copying multiple items that indicates 'finish'.
        # 1 (i.e yes) is too few, add some new instances by copying old ones

        df = df.append([df[df['event type'] == 1]] * 5) # copy 5 times

        # Take a look at it again
        print(df['event type'].value_counts())
```

```
1    67032
0    66604
Name: event type, dtype: int64
```

```
In [5]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 133636 entries, 0 to 77774
Data columns (total 11 columns):
```

```
time                                133636 non-null int64
job ID                              133636 non-null int64
task index                          133636 non-null int64
machine ID                          77004 non-null float64
event type                          133636 non-null int64
scheduling class                    133636 non-null int64
priority                            133636 non-null int64
CPU request                         133636 non-null float64
memory request                      133636 non-null float64
disk space request                  133636 non-null float64
different machines restriction      133636 non-null int64
dtypes: float64(4), int64(7)
memory usage: 12.2 MB
```

```
In [6]: cleaned = df.dropna()
        # print(cleaned)
        print(cleaned['event type'].value_counts())

        # 'cleaned' is the processed dataset.
        cleaned.to_csv('Cleaned_data_after_processing.csv',encoding='utf-8',index=False)
```

```
0    40782
1    36222
Name: event type, dtype: int64
```

```
In [7]: ## Data splitting for training neural network
        X = cleaned.drop('event type', axis=1) # Remove label
        Y = cleaned['event type'] # Label

        train_x, valid_x, train_y, valid_y = train_test_split(X, Y, test_size=0.25, random_sta

        ss = StandardScaler()
        train_x = ss.fit_transform(train_x)
        valid_x = ss.transform(valid_x)
```

```
In [ ]:
```