Họ và tên: **Lê Khắc Thanh Tùng**

MSSV: **52100943**

My example dataset: https://archive.ics.uci.edu/dataset/2/adult

Study issues in data processing for machine learning:

- **Data cleaning**: Removing missing values, duplicates and outliers, and handling errors and inconsistencies in the data
- Remove duplicate data
    + **Problem**: trong trường hợp tập dữ liệu có các dữ liệu trùng lặp thì ta nên xóa chúng ra để tránh gây nhiễu khi train model
    + **Solution**:

```
1   print(df.drop_duplicates().values)
```

```
[[50 ' Self-emp-not-inc' 83311 ... 13 ' United-States' ' <=50K']
 [38 ' Private' 215646 ... 40 ' United-States' ' <=50K']
 [53 ' Private' 234721 ... 40 ' United-States' ' <=50K']
 ...
 [58 ' Private' 151910 ... 40 ' United-States' ' <=50K']
 [22 ' Private' 201490 ... 20 ' United-States' ' <=50K']
 [52 ' Self-emp-inc' 287927 ... 40 ' United-States' ' >50K']]
```

- Handle missing data
    + **Problem**: trong tập dữ liệu không đảm bảo là observation nào cũng có đầy đủ dữ liệu ở các feature nên trong trường hợp chúng thiếu thì ta nên xóa observation đó để tránh ảnh hưởng khi train model
    + **Solution**: xóa đi observation missing value

```
df = df.replace(" ?", np.NaN)
print(df.shape) # before delete observation with missing data
df.dropna(subset=[' United-States'], axis=0, inplace=True)
print(df.values.shape) # after delete observation with missing data
```

```
(32560, 15)
(31977, 15)
```

- Feature selection (remove irrelevant data)
    + **Problem**: trong 1 số trường hợp ta không muốn lấy hết các feature của dataset vì có có các feature không liên quan
    + **Solution**:

Feature selection (Remove irrelevant data)

```
print(df.drop(columns=['39']).shape) # drop feature age and retain other features
print(df.shape)
```

```
(32560, 14)
(32560, 15)
```

- **Data transformation**: Converting the data into a suitable format for the machine learning algorithm, such as encoding categorical variables, transforming skewed or imbalance data, and create new features
    - **Problem**: Các thuật toán học máy thường hoạt động trên dữ liệu số vì vậy cần encoding categorical variables (chuyển sang dữ liệu dạng số) để thuật toán xử lý tốt hơn
    - **Solution**: Encoding categorical variables

      Đây là kiểu dữ liệu của các feature trước khi encoding categorical variables như ta thấy ngoài các feature mang kiểu dữ liệu số như age,... thì vẫn còn những categorical variable như workclass,.. và nếu những feature đó ta có sử dụng để train model thì ta nên chuyển sang dữ liệu kiểu số

```
39                int64
 State-gov       object
 77516            int64
 Bachelors       object
 13               int64
 Never-married   object
 Adm-clerical    object
 Not-in-family   object
 White           object
 Male            object
 2174             int64
 0                int64
 40               int64
 United-States   object
 <=50K           object
dtype: object
```

```
[38]  1  print(data[0])

    [50 ' Self-emp-not-inc' 83311 ' Bachelors' 13 ' Married-civ-spouse'
     ' Exec-managerial' ' Husband' ' White' ' Male' 0 0 13 ' United-States'
     ' <=50K']
```

Sau khi encoding categorical variables bằng LabelEncoder thì tất cả các feature của tập dữ liệu đã trở thành dữ liệu kiểu số

```python
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
for i in range(len(types)):
    if types[i] == 'object':
        le.fit_transform(df[names[i]])
        df[names[i]] = le.transform(df[names[i]])

print(df.dtypes)
```

```
39                int64
 State-gov        int64
 77516            int64
 Bachelors        int64
 13               int64
 Never-married    int64
 Adm-clerical     int64
 Not-in-family    int64
 White            int64
 Male             int64
 2174             int64
 0                int64
 40               int64
 United-States    int64
 <=50K            int64
dtype: object
```

```python
print(data[0])
```

```
[   50     6 83311     9    13     2     4     0     4     1     0     0
    13    39     0]
```

- **Data normalization**: Scaling the data so that it has the same range of values, allowing the machine learning algorithms to treat all the features equally.
    - **Problem**: Sau khi ta đã encoding categorical variables (như hình trên) thì ta thấy rằng có các feature có miền giá trị rất nhỏ ví dụ như feature age có miền giá trị [17, 90]

```
print(min(df['39']))
print(max(df['39']))
```

```
17
90
```

còn có các feature có miền giá trị rất lớn ví dụ như feature fnlwgt có miền giá trị [12285, 1484705]

```
print(min(df[' 77516']))
print(max(df[' 77516']))
```

```
12285
1484705
```

Điều này sẽ gây ảnh hưởng cho việc khi ta sử dụng một số thuật toán học máy yêu cầu chuẩn hóa dữ liệu trước khi train cho model để đảm bảo tính chính xác thì những feature có miền giá trị quá nhỏ sẽ bị bỏ qua trong quá trình học do các feature có miền giá trị lớn đã chiếm ưu thế như thế model sẽ không còn chính xác

- **Solution**: Scaling tất cả các miền giá trị của các feature về miền giá trị chung [0, 1] để model được train một cách chính xác nhất

```
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
scaler.fit(X)
X_scaled = scaler.transform(X)
print(X_scaled[1])
```

```
[0.28767123 0.5        0.13811345 0.73333333 0.53333333 0.
 0.42857143 0.2        1.         1.         0.         0.
 0.39795918 0.95121951]
```