Chapter 4 - Ex 3: Titanic

Cho dữ liệu titanic3.csv chứa thông tin về các hành khách trên con tàu Titanic

- Một trong những thông tin quan trọng để dự đoán một hành khách còn sống hay đã chết là 'age', 'fare'. Kiểm tra xem dữ liệu trên 2 cột này có null hay không, nếu có hãy xóa bỏ các dòng null. Phân tích thông tin sơ bộ về dữ liệu trên hai thuộc tính này. Trực quan hóa dữ liệu.
- Để việc dự đoán tốt hơn cần phải kiểm tra và chuẩn hóa dữ liệu. Hãy chọn một phương pháp để chuẩn hóa dữ liệu dựa trên thông tin nêu trên.

Gợi ý

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```
# Đọc dữ liệu. Tìm hiểu thông tin sơ bộ về dữ liệu
data = pd.read_csv("titanic3.csv")
data.head()
```

Out[2]:

	pclass	survived	name	sex	age	sibsp	parch	ticket	fare	cabin	embarked	boat	body	hor
0	1	1	Allen, Miss. Elisabeth Walton	female	29.0000	0	0	24160	211.3375	B5	S	2	NaN	S
1	1	1	Allison, Master. Hudson Trevor	male	0.9167	1	2	113781	151.5500	C22 C26	S	11	NaN	M Ches
2	1	0	Allison, Miss. Helen Loraine	female	2.0000	1	2	113781	151.5500	C22 C26	S	NaN	NaN	M Ches
3	1	0	Allison, Mr. Hudson Joshua Creighton	male	30.0000	1	2	113781	151.5500	C22 C26	S	NaN	135.0	M Ches
4	1	0	Allison, Mrs. Hudson J C (Bessie Waldo Daniels)	female	25.0000	1	2	113781	151.5500	C22 C26	S	NaN	NaN	M Ches

In [3]:
```
data = data[['survived', 'pclass', 'sex', 'age', 'sibsp', 'parch', 'fare', 'embarked']]
```

```
data.head()
```

Out[3]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked
0	1	1	female	29.0000	0	0	211.3375	S
1	1	1	male	0.9167	1	2	151.5500	S
2	0	1	female	2.0000	1	2	151.5500	S
3	0	1	male	30.0000	1	2	151.5500	S
4	0	1	female	25.0000	1	2	151.5500	S

In [4]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1309 entries, 0 to 1308
Data columns (total 8 columns):
survived    1309 non-null int64
pclass      1309 non-null int64
sex         1309 non-null object
age         1046 non-null float64
sibsp       1309 non-null int64
parch       1309 non-null int64
fare        1308 non-null float64
embarked    1307 non-null object
dtypes: float64(2), int64(4), object(2)
memory usage: 81.9+ KB
```

In [5]:
```
# Kiểm tra dữ liệu null
print(data.isnull().sum())
# => Age có 263 dữ liệu null, fare có 1 dữ liệu null
```

```
survived      0
pclass        0
sex           0
age         263
sibsp         0
parch         0
fare          1
embarked      2
dtype: int64
```

In [6]:
```
# Xóa dữ liệu null
data = data.dropna()
```

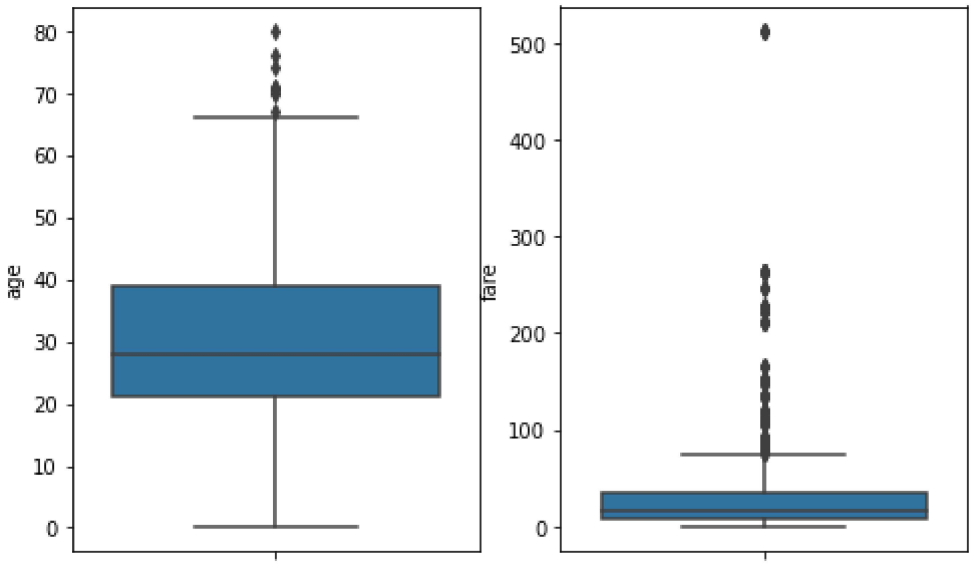
In [7]:
```
data.describe()
```

Out[7]:

	survived	pclass	age	sibsp	parch	fare
count	1043.000000	1043.000000	1043.000000	1043.000000	1043.000000	1043.000000
mean	0.407478	2.209012	29.813199	0.504314	0.421860	36.603024
std	0.491601	0.840685	14.366261	0.913080	0.840655	55.753648
min	0.000000	1.000000	0.166700	0.000000	0.000000	0.000000
25%	0.000000	1.000000	21.000000	0.000000	0.000000	8.050000

	survived	pclass	age	sibsp	parch	fare
50%	0.000000	2.000000	28.000000	0.000000	0.000000	15.750000
75%	1.000000	3.000000	39.000000	1.000000	1.000000	35.077100
max	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

In [8]:
```
# Phân tích đơn biến: trực quan hóa, kiểm tra dữ liệu outlier
# Trực quan hóa dữ liệu cho từng biến liên tục
plt.figure(figsize=(8,5))
plt.subplot(1,2,1)
sns.boxplot(data.age, orient="v")
plt.subplot(1,2,2)
sns.boxplot(data.fare, orient="v")
plt.show()
# => Cả hai biến liên tục fare và age đều có outlier
```



In [9]:
```
np.ptp(data.age)
```

```
c:\program files\python36\lib\site-packages\numpy\core\fromnumeric.py:2542: FutureWarning: Method
.ptp is deprecated and will be removed in a future version. Use numpy.ptp instead.
  return ptp(axis=axis, out=out, **kwargs)
```

Out[9]: 79.8333

In [10]:
```
np.ptp(data.fare)
# Có khoảng cách lớn giữa min và max
# 2 thang đo cho 2 cột khác nhau
```

Out[10]: 512.3292

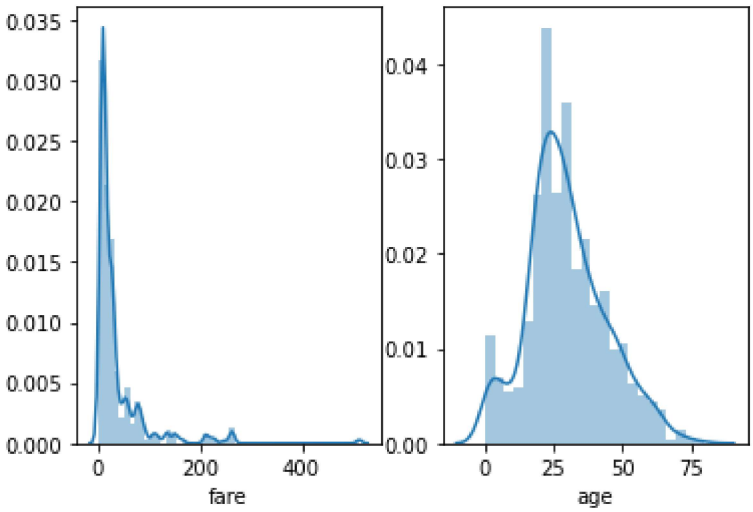
In [11]:
```
data.age.skew()
```

Out[11]: 0.40688028266803467

In [12]:
```
data.fare.skew()
```

Out[12]: 4.122508729348891

In [13]:
```
plt.subplot(1,2,1)
sns.distplot(data.fare)
plt.subplot(1,2,2)
sns.distplot(data.age)
plt.show()
```



Nhận xét:

- Nhìn biểu đồ trên và giá trị skew ta thấy age hơi lệch phải, còn fare lệch phải nhiều
- Dữ liệu có outlier #### => Nếu muốn giữ outier => Chọn Robust Scaler. Nếu loại bỏ outlier => Chọn MinMaxScaler

In [14]:
```
from sklearn import preprocessing
```

In [15]:
```
age_fare = data[['age', 'fare']].astype('float64')
age_fare.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1043 entries, 0 to 1308
Data columns (total 2 columns):
age     1043 non-null float64
fare    1043 non-null float64
dtypes: float64(2)
memory usage: 24.4 KB
```

Robust Scaler

In [16]:
```
scaler = preprocessing.RobustScaler()
robust_scaler = scaler.fit_transform(age_fare)
df = pd.DataFrame(robust_scaler, columns=['age_scaler', 'fare_scaler'])
df.head()
```

Out[16]:

	age_scaler	fare_scaler
0	0.055556	7.236718
1	-1.504628	5.024586

	age_scaler	fare_scaler
2	-1.444444	5.024586
3	0.111111	5.024586
4	-0.166667	5.024586

In [17]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1043 entries, 0 to 1042
Data columns (total 2 columns):
age_scaler     1043 non-null float64
fare_scaler    1043 non-null float64
dtypes: float64(2)
memory usage: 16.4 KB
```

In [18]:
```
data['age_scaler'] = df.age_scaler.values
data['fare_scaler'] = df.fare_scaler.values
data.head()
```

Out[18]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	age_scaler	fare_scaler
0	1	1	female	29.0000	0	0	211.3375	S	0.055556	7.236718
1	1	1	male	0.9167	1	2	151.5500	S	-1.504628	5.024586
2	0	1	female	2.0000	1	2	151.5500	S	-1.444444	5.024586
3	0	1	male	30.0000	1	2	151.5500	S	0.111111	5.024586
4	0	1	female	25.0000	1	2	151.5500	S	-0.166667	5.024586

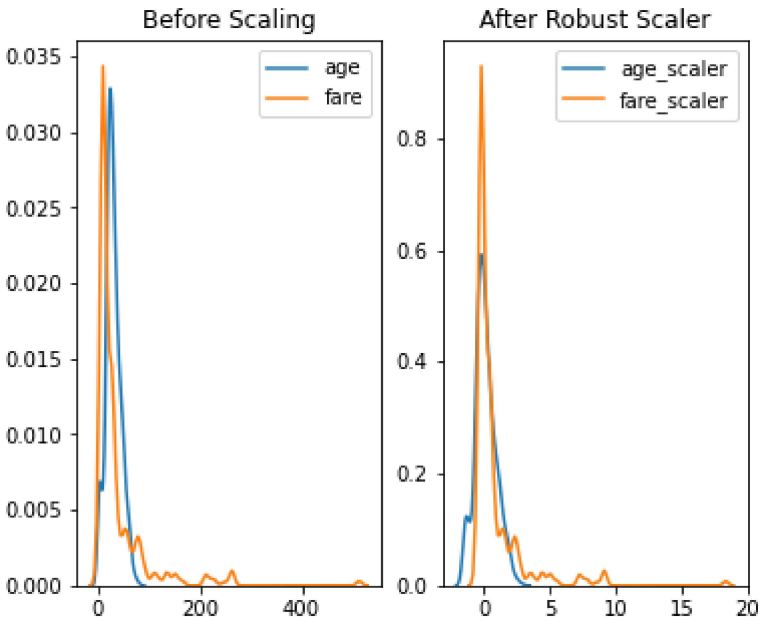
In [19]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1043 entries, 0 to 1308
Data columns (total 10 columns):
survived       1043 non-null int64
pclass         1043 non-null int64
sex            1043 non-null object
age            1043 non-null float64
sibsp          1043 non-null int64
parch          1043 non-null int64
fare           1043 non-null float64
embarked       1043 non-null object
age_scaler     1043 non-null float64
fare_scaler    1043 non-null float64
dtypes: float64(4), int64(4), object(2)
memory usage: 89.6+ KB
```

In [20]:
```
fig, (ax1, ax2) = plt.subplots(ncols=2, figsize=(6, 5))
ax1.set_title('Before Scaling')
sns.kdeplot(data['age'], ax=ax1)
sns.kdeplot(data['fare'], ax=ax1)

ax2.set_title('After Robust Scaler')
sns.kdeplot(data['age_scaler'], ax=ax2)
sns.kdeplot(data['fare_scaler'], ax=ax2)
```

```
plt.show()
```



In [21]:
```
X = pd.get_dummies(data)
X.head()
```

Out[21]:

	survived	pclass	age	sibsp	parch	fare	age_scaler	fare_scaler	sex_female	sex_male	embarked_C	em
0	1	1	29.0000	0	0	211.3375	0.055556	7.236718	1	0	0	
1	1	1	0.9167	1	2	151.5500	-1.504628	5.024586	0	1	0	
2	0	1	2.0000	1	2	151.5500	-1.444444	5.024586	1	0	0	
3	0	1	30.0000	1	2	151.5500	0.111111	5.024586	0	1	0	
4	0	1	25.0000	1	2	151.5500	-0.166667	5.024586	1	0	0	

In [22]:
```
y=data['survived']  # Labels
```

Dữ liệu gốc

In [23]:
```
X_original = X[['pclass', 'age', 'sibsp', 'parch', 'fare', 'sex_female',
                'sex_male', 'embarked_C', 'embarked_Q', 'embarked_S']]
```

In [24]:
```
from sklearn.linear_model import LogisticRegression
```

In [25]:
```
model = LogisticRegression(solver='liblinear')
```

In [26]:
```
model.fit(X_original, y)
```

Out[26]:
```
LogisticRegression(solver='liblinear')
```

In [27]:
```
from sklearn.metrics import accuracy_score
```

In [28]:
```
print("Acc:", round(accuracy_score(model.predict(X_original), y),3))
```

```
Acc: 0.789
```

Dữ liệu có scale

In [29]:
```
X_scale = X[['pclass', 'age_scaler', 'sibsp', 'parch', 'fare_scaler',
             'sex_female', 'sex_male', 'embarked_C', 'embarked_Q', 'embarked_S']]
```

In [30]:
```
model_s = LogisticRegression(solver='liblinear')
```

In [31]:
```
model_s.fit(X_scale, y)
```

Out[31]:
```
LogisticRegression(solver='liblinear')
```

In [32]:
```
print("Acc:", round(accuracy_score(model_s.predict(X_scale), y),3))
```

```
Acc: 0.791
```

Kết luận:

- Với kết quả nhận được nói trên ta thấy việc scale hay không scale dữ liệu của 2 cột 'age và 'fare' cho độ chính xác gần như là như nhau
- Lý do: có thể vai trò của cột age và/hoặc fare không nhiều trong việc dự đoán