

# Machine Learning with Python

*Life is too short, You need Python*



## 실습 내용

- Happy 데이터를 불러와 살펴봅니다.
- 단순 선형회귀 모델을 만들고 평가합니다.
- 특히 회귀 계수를 명확히 이해합니다.
- 예측 결과를 시각화합니다.

## 1.환경 준비

- 기본 라이브러리와 대상 데이터를 가져와 이후 과정을 준비합니다.

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

warnings.filterwarnings(action='ignore')
%config InlineBackend.figure_format='retina'
```

```
In [2]: # 데이터 읽어오기
path = 'https://raw.githubusercontent.com/Jangrae/csv/master/income_happy.csv'
data = pd.read_csv(path)
```

## 2.데이터 이해

- 분석할 데이터를 충분히 이해할 수 있도록 다양한 탐색 과정을 수행합니다.

```
In [3]: # 상위 몇 개 행 확인
data.head()
```

```
Out[3]:
```

	income	happiness
0	3.862647	2.314489
1	4.979381	3.433490
2	4.923957	4.599373
3	3.214372	2.791114
4	7.196409	5.596398

### 데이터 설명

- income: 수입 (단위: 10,000\$)
- happiness: 행복 정도 (1~ 10)

```
In [4]: # 하위 몇 개 행 확인
data.tail()
```

```
Out[4]:
```

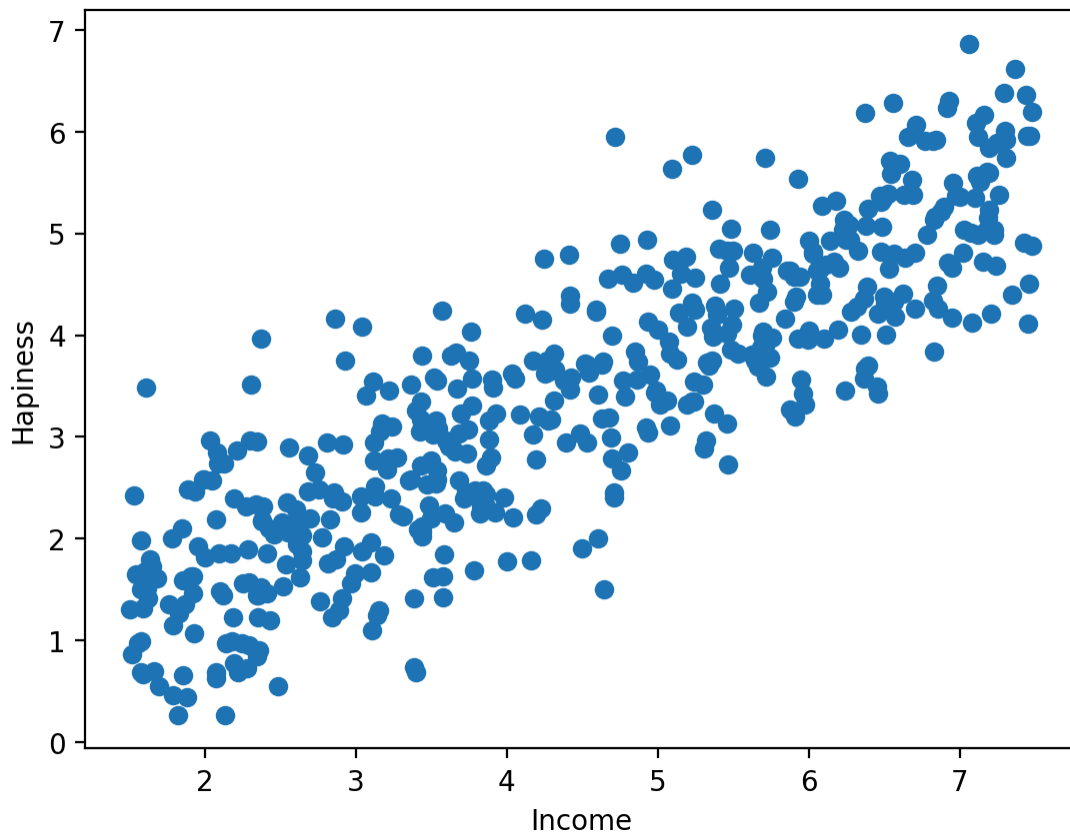
	income	happiness
493	5.249209	4.568705
494	3.471799	2.535002
495	6.087610	4.397451
496	3.440847	2.070664
497	4.530545	3.710193

```
In [5]: # 기술통계 확인
data.describe()
```

Out[5]:

	income	happiness
<b>count</b>	498.000000	498.000000
<b>mean</b>	4.466902	3.392859
<b>std</b>	1.737527	1.432813
<b>min</b>	1.506275	0.266044
<b>25%</b>	3.006256	2.265864
<b>50%</b>	4.423710	3.472536
<b>75%</b>	5.991913	4.502621
<b>max</b>	7.481521	6.863388

```
In [6]: # Income --> Happiness 관계
plt.scatter(x='income', y='happiness', data=data)
plt.xlabel('Income')
plt.ylabel('Happiness')
plt.show()
```



### 3.데이터 준비

- 전처리 과정을 통해 머신러닝 알고리즘에 사용할 수 있는 형태의 데이터를 준비합니다.

#### 1) x, y 분리

```
In [7]: # target 확인
target = 'happiness'

# 데이터 분리
x = data.drop(target, axis=1)
y = data.loc[:, target]
```

## 2) 학습용, 평가용 데이터 분리

```
In [8]: # 모듈 불러오기
from sklearn.model_selection import train_test_split

# 7:3으로 분리
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.3, random_state=1)
```

# 4.모델링

- 본격적으로 모델을 선언하고 학습하고 평가하는 과정을 진행합니다.

```
In [9]: # 1단계: 불러오기
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, r2_score
```

```
In [10]: # 2단계: 선언하기
model = LinearRegression()
```

```
In [11]: # 3단계: 학습하기
model.fit(x_train, y_train)
```

```
Out[11]: ▼ LinearRegression
LinearRegression()
```

```
In [12]: # 4단계: 예측하기
y_pred = model.predict(x_test)
```

```
In [13]: # 5단계: 평가하기
print('MAE:', mean_absolute_error(y_test, y_pred))
print('R2:', r2_score(y_test, y_pred))
```

```
MAE: 0.5588237111283767
R2: 0.7545536758581621
```

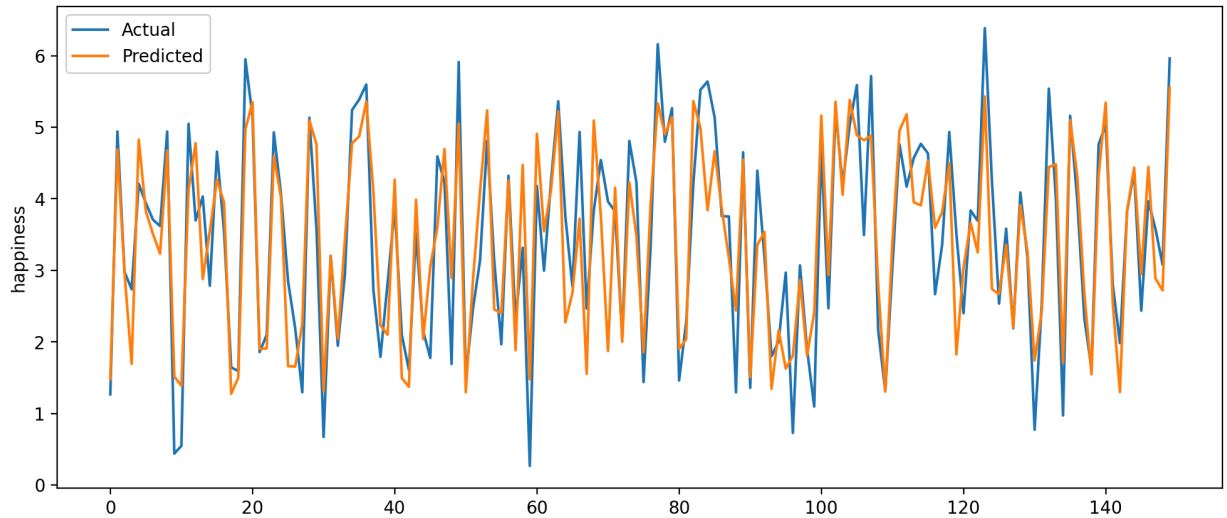
```
In [14]: # 회귀계수 확인
print(model.coef_)
print(model.intercept_)
```

```
[0.72244152]
0.15849441169520562
```

# 5.기타

- 기타 필요한 내용이 있으면 진행합니다.

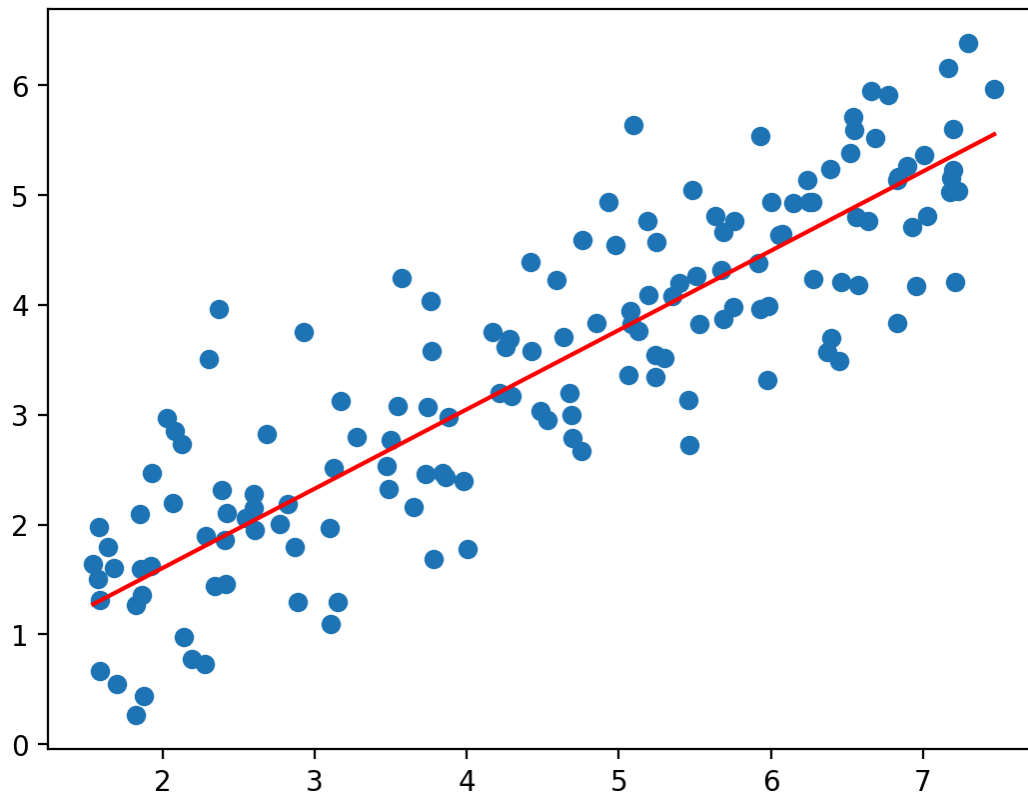
```
In [22]: # 시각화
plt.figure(figsize=(12, 5))
plt.plot(y_test.values, label='Actual')
plt.plot(y_pred, label='Predicted')
plt.legend()
plt.ylabel('happiness')
plt.show()
```



```
In [15]: # 예측값, 실제값 시각화
#  $y = a * x + b$ 
a = model.coef_
b = model.intercept_
x_ = np.linspace(x_test.min(), x_test.max(), 2)
happy = a * x_ + b
```

```
In [19]: plt.scatter(x_test, y_test) # 평가 데이터
# plt.scatter(x_train, y_train, color='orange') # 학습 데이터
plt.plot(x_, happy, color='r') # 학습 데이터 회귀선

plt.show()
```



In [ ]: