머신러닝

- 기계가 자동으로 데이터를 학습하여 모델을 만드는 것.

분류1) 데이터셋의 구성

- 지도학습 : x와 y를 모두 사용하는 학습
  - k-최근접 이웃
  - 선형회귀
  - 로지스틱회귀
  - SVM
  - 결정 트리와 랜덤 포레스트
  - 신경망
- 비지도학습 : 훈련 데이터에 레이블이 없는 것
  - 군집 : k-means, DBSCAN, 계층 군집 분석, 이상치 감지와 특이치 탐지, 원-클래스, 아이솔레이션 포레스트
    - 이상치 탐지 : 훈련 세트에 있는 데이터셋에서 이상한 값을 감지 후 이를 이상치로 분류, 자동으로 제거 -> 새로운 샘플을 보고 정상데이터인지 이상치인지 판단
    - 특이치 탐지 : 훈련세트에 잇는 모든 샘플과 달라 보이는 새로운 샘플을 탐지하는 것이 목적(매우 깨끗한 훈련 세트 필요)
  - 시각화와 차원축소 : 주성분 분석(PCA), kernal PCA, 지역적 선형 임베딩, t-SNE
  - 연관 규칙 학습 : 어프라이러리, 이클렛
- 준지도학습 : 일부만 레이블이 있는 데이터, 지도학습과 비지도 학습의 조합으로 이루어져있음
  - ex) 심층 신뢰 신경망(DBN) - RBM 비지도학습에 기초 후, 지도 학습 방식으로 세밀하게 조정
- 강화학습 : 에이전트(학습 시스템)가 환경을 관찰해서 행동하고, 보상을 받음 -> 가장 큰 보상을 부르는 정책(최상의 전략)을 스스로 학습.

분류 2) 점진적인가?

- 배치 학습 : 가용한 데이터를 모두 사용해 훈련 후, 제품 시스템에 적용(더 이상의 학습 없이 실행)하는 오프라인 학습
  - 새로운 데이터를 학습하기 위해서는 전체 데이터(이전 데이터 + 새로운 데이터)를 사용하여 처음부터 다시 훈련시켜야 함
  - 많은 시간 많은 컴퓨팅 자원이 필요
  - 빠른 변화에 적응해야 하는 시스템에 적합하지 않음.
- 미니 배치 학습 : 데이터를 묶음 단위로 주입하여 시스템을 훈련
  - 적은 시간과 컴퓨팅 자원, 데이터가 도착하는 대로 즉시 학습 가능
  - 나쁜 데이터가 주입되었을 때 시스템 성능이 점진적으로 감소
  - 학습률 : 새로운 데이터에 얼마나 영향을 받을 것인지를 결정
    - 낮은 학습률 : 시스템의 관성이 커져 더 느리게 학습, 잡음이나 대표성 없는 데이터 포인트에 덜 민감
    - 높은 학습률 : 빠르게 적응하지만 예전 데이터를 금방 잊어버림.

분류 3) 일반화가 되는가?

- 사례기반학습 : 테스트셋을 단순히 기억하여 유사도 측정을 통해 예측
- 모델기반학습 : 모델을 만들어 예측에 사용

머신러닝의 주요 도전 과제

나쁜 데이터

- 충분하지 않은 데이터 : 데이터가 충분히 많다면 간단한 모델 또한 훌륭하게 예측해냄
  - 하지만 데이터가 충분하기 쉽지 않기 때문에 알고리즘에 노력 또한 기울여야 함
- 대표성 없는 훈련 데이터
  - 1) 여론조사용에 특정 명부를 사용한다면 특정 성향을 가진 계층으로 샘플링이 편향될 것
  - 2) 우편물 수신자 중 25% 미만의 사람이 응답 -> 정치에 관심없는 사람, 싫어하는 사람을 제외시킴으로써 표본이 편향 -> 비응답 편향
- 낮은 품질의 데이터 : 에러, 이상치, 결측값
- 관련 없는 특성 : 훈련 데이터 셋에 관련없는 특성이 적고 관련 있는 특성이 충분해야함
  - 특성 선택 : 가지고 있는 특성 중에서 훈련에 가장 유용한 특성 선택
  - 특성 추출 : 특성을 결합하여 더 유용한 특성을 만듦, ex) 차원 축소
  - 새로운 데이터를 수집하여 새 특성을 만듦

나쁜 알고리즘

- 과대적합 알고리즘 : 훈련 데이터에 있는 잡음의 양에 비해 모델이 너무 복잡하여 일반화가 어려운 경우
  - 파라미터 수가 적은 모델 선택(고차원 -> 저차원)
  - 제약을 가하여 단순화 시킴(하이퍼 파라미터를 이용하여 자유도를 줄임)
  - 더 많은 훈련 데이터
  - 훈련데이터의 잡음을 줄임(이상치제거, 오류 제거)
- 과소적합 알고리즘 : 모델이 너무 단순한 경우
  - 파라미터 수가 더 많은 모델 선택
  - 더 좋은 특성을 제공
  - 모델의 제약을 줄임

테스트와 검증

- 검증 세트를 통하여 홀드아웃 검증
- 검증 세트가 작을 경우 교차 검증을 통해 검증(시간이 늘어난다는 단점)
- 데이터 불일치의 경우(웹이미지로 앱이미지)
  - 트레인, 트레인개발, 검증, 테스트를 통하여 정확도를 검증할 수도 있음.
  - 트레인개발 set을 이용하여 데이터의 특성이 다른건지, overfitting 된 것인지를 알 수 있음

실습

In [25]:

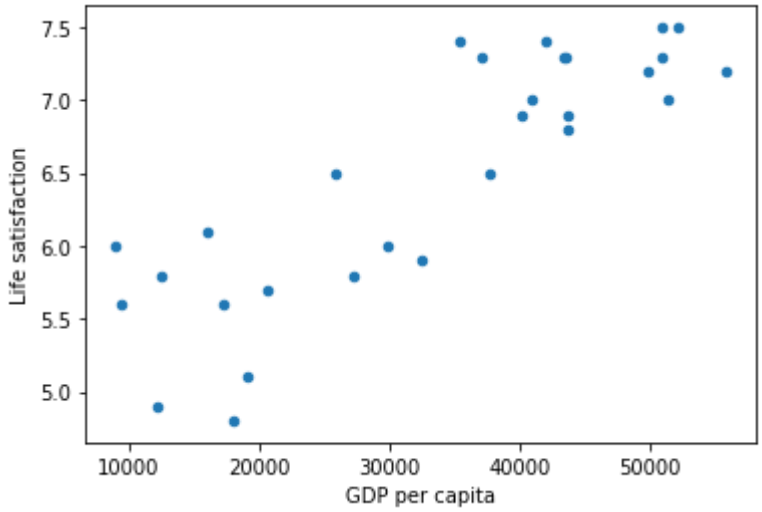
```
# 예제 1-1 사이킷런을 이용한 선형 모델의 훈련과 실행
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import sklearn.linear_model
import os
import seaborn as sns
os.chdir("C:₩₩Users₩₩이혜림₩₩Desktop₩₩핸즈온 머신러닝2₩₩handson-ml2-master₩₩handson-ml2-master₩₩data

# 데이터 적재
oecd_bli = pd.read_csv("./lifesat/oecd_bli_2015.csv", thousands=",")
gdp_per_capita = pd.read_csv("./lifesat/gdp_per_capita.csv", thousands = ",", delimiter="₩t",
                             encoding="latin1", na_values = "n/a")
```

In [26]:

```
def prepare_country_stats(oecd_bli, gdp_per_capita):
    oecd_bli = oecd_bli[oecd_bli["INEQUALITY"]=="TOT"]
    oecd_bli = oecd_bli.pivot(index="Country", columns="Indicator", values="Value")
    gdp_per_capita.rename(columns={"2015": "GDP per capita"}, inplace=True)
    gdp_per_capita.set_index("Country", inplace=True)
    full_country_stats = pd.merge(left=oecd_bli, right=gdp_per_capita,
                                  left_index=True, right_index=True)
    full_country_stats.sort_values(by="GDP per capita", inplace=True)
    remove_indices = [0, 1, 6, 8, 33, 34, 35]
    keep_indices = list(set(range(36)) - set(remove_indices))
    return full_country_stats[["GDP per capita", 'Life satisfaction']].iloc[keep_indices]
```

In [27]:

```
# 데이터 준비
country_stats = prepare_country_stats(oecd_bli, gdp_per_capita) # 두 데이터를 merge

# 2차원을 유지하면서 numpy로 바꿔주기 위하여 np.c_ 사용, 그냥 array 하면 1차원으로 바뀜
# np.c_ -> 1차원 데이터들을 세로로 붙임으로써 2차원을 만듦
# np.r_ -> 1차원 데이터들을 가로로 붙임으로써 1차원을 만듦

X = np.c_[country_stats["GDP per capita"]]
y = np.c_[country_stats["Life satisfaction"]]

# 데이터 시각화
country_stats.plot(kind="scatter",x="GDP per capita", y="Life satisfaction")
plt.show()

# 선형 모델 선택
model = sklearn.linear_model.LinearRegression()

# 모델 훈련
model.fit(X,y)

# 키프로스에 대한 예측 만들기
X_new = [[22587]] # 키프로스 1인당 GDP
print(model.predict(X_new))
```




```
[[5.96242338]]
```