|  |  |
| --- | --- |
| 교육제목 | 데이터 기반 인공지능 시스템 엔지니어 양성 과정 |
| 교육일시 | 211013 |
| 교육장소 | 자택(디스코드 사용 온라인 학습) |
| **교육내용** | |
| 1. 머신러닝 – 모델링 프로세스  * 지도학습(Supervised Learning): 분류(Classification), 회귀(Regression)으로 나뉨.   분류 : Pass/Fail, Positive/Negative, Good/Bad 같이 데이터를 분류하는 것.  회귀 : 범주형/연속형데이터를 다루는것.   1. 선형회귀  * 지도학습 * 목적변수(반응변수)가 연속형일 경우 * 정규성, 독립성, 등분산성을 만족해야 함 * Feature가 하나인 경우 단순회귀(Simple), Feature가 여러 개일 경우 중회귀(Multiple) * 2차항 이상이 포함된 경우 다항회귀(Polynomial regression)  1. 단순회귀   캡처.JPG   1. 최소제곱 추정량   캡처.JPG  캡처.JPG  임의로 그린 선과 자료간의 거리를 오류값, ‘e(error)’라고 했을 때 오류값이 음수 혹은 0으로 나오지 않게 제곱하여 모든 e를 더해 추정하는 방법   1. 다항회귀   캡처.JPG  캡처.JPG   1. 기계학습의 분류  * Supervised Learning : 학습 데이터에 레이블 포함 * Unsupervised Learning : 학습 데이터 레이블 없음 * 임의로 그룹의 수를 정해주면 컴퓨터가 알아서 나눠준다. * PCA(Principal component analysis) : 2개 이상의 Feature를 하나로 줄일 때 ‘데이터 상관성’에 기반하여 줄이며 손실을 최소화 하는 것. * Semi-Supervised Learning : 레이블이 있는 데이터와 없는 데이터를 조합함. * Reinforcement Learning : 에이전트가 상태의 보상을 최대화 하기 위해 행동함.  1. 모델의 치우침과 분산   캡처.JPG  캡처.JPG     1. Parameter estimation   캡처.JPG  import numpy as np  import pandas as pd  import matplotlib.pyplot as plt  np.random.seed(1013)  x = np.arange(1, 30, 1, dtype = np.int16)  y = 2 \* x + 1  # 노이즈 추가  y\_random = y + np.random.normal(loc = 0, scale = 8, size = len(y))  #plt.plot(x, y, ".", markersize = 20)  plt.plot(x, y\_random, ".", markersize = 20)  plt.show()  Sxy = (x - np.mean(x))\*(y\_random - np.mean(y\_random))  Sxx = (x - np.mean(x))\*\*2  Sxy = Sxy.sum()  Sxx = Sxx.sum()  beta\_1 = Sxy/Sxx  beta\_0 = np.mean(y\_random) - beta\_1\*np.mean(x)  print("beta\_1 = {}".format(beta\_1.round(3)))  print("beta\_0 = {}".format(beta\_0.round(3)))  #기울기와 y절편으로 그래프 그리는 코드  def abline(slope, intercept):  """Plot a line from slope and intercept"""  axes = plt.gca()  x\_vals = np.array(axes.get\_xlim())  y\_vals = intercept + slope \* x\_vals  plt.plot(x\_vals, y\_vals, '--')  ## 추정량  y\_hat = 2.01\*x + 1.031  print(y\_hat)  #plt.plot(x, y, ".", markersize = 20)  plt.plot(x, y\_random, ".", markersize = 20)  plt.plot(x, y\_hat, color = "r")  # 추정량(y\_hat)이 어느정도 노이즈를 통해서 직선이 그어짐  ## sklearn import  from sklearn.linear\_model import LinearRegression  # 3줄 코드로 위에서 했던 계산 다 할수있음  lr = LinearRegression()  x\_2d = x.reshape(-1, 1)  lr.fit(x\_2d, y\_random)  print(lr.coef\_, lr.intercept\_) | |