|  |  |
| --- | --- |
| 실습보고서 | |
| 실습01: 선형회귀 | |
| 학번:2019146037 | 이름:홍석영 |

\*주의사항

- 양식 및 폰트 변경하지 않고 사용할 것

**실습 01**

|  |  |
| --- | --- |
| 코드 (텍스트 형태로 copy할 것, 폰트크기 9pt) | |
| import numpy as np  import matplotlib.pyplot as plt  import pandas as pd  raw\_data = pd.read\_csv('lin\_regression\_data\_01.csv',names = ['x','y'])  age\_data = np.asarray(raw\_data['x'].values.tolist())  height\_data = np.asarray(raw\_data['y'].values.tolist())  plt.plot(age\_data,height\_data,'ro')  plt.legend(['People Data'])  plt.xlabel('age')  plt.ylabel('height')  plt.grid(True)  plt.show() |  |

|  |
| --- |
| 그래프 (이미지 copy할 것) |
|  |

|  |
| --- |
| 설명 (본 실습 과제의 중요 이론 및 결과를 간략히 설명) |
| 엑셀(csv) 파일에 들어있는 첫번째 열 데이터(나이)와 두번째 열 데이터(키)를 ‘np.panda’ 를 사용하여 불러오고, 나이와 키를 ‘np.asarray’를 사용하여 각각 x축 데이터 행렬, y축 데이터 행렬로 지정한다음 ‘plot’을 사용하여 각 데이터의 위치를 빨간 점(People Data) 그래프로 표현하였습니다.  age\_data는 x축의 나이 데이터를 저장한 배열, height\_data는 y축의 키 데이터를 저장한 배열입니다. |

**실습 02**

|  |  |
| --- | --- |
| 코드 (텍스트 형태로 copy할 것, 폰트크기 9pt) | |
| import numpy as np  import matplotlib.pyplot as plt  import pandas as pd  raw\_data = pd.read\_csv('lin\_regression\_data\_01.csv',names = ['x','y'])  age\_data = np.asarray(raw\_data['x'].values.tolist())  height\_data = np.asarray(raw\_data['y'].values.tolist())  x\_average = sum(age\_data)/len(age\_data)  w0\_fraction = sum(height\_data\*(age\_data - x\_average))/len(age\_data)  w0\_denominator = sum((age\_data\*age\_data)-(x\_average\*x\_average))/len(age\_data)  w0 = w0\_fraction/w0\_denominator  w1 = sum(((height\_data) - (w0\*age\_data)))/len(age\_data) | t = np.linspace(min(age\_data),max(age\_data),25)  y\_hat = w0\*t + w1  y\_hat\_m = w0\*age\_data + w1 |

|  |
| --- |
| 최적해 및 설명 (본 실습 과제의 중요 이론 및 결과를 간략히 설명) |
| 최적해란 y\_hat과 y의 평균제곱오차(MSE)의 값을 최소화해주는 해 즉, 와 를 의미합니다.  최적해 와 을 구하는 식은 다음과 같습니다.    최적 선형 회귀 모델 y hat = w0\*x +  위와 같은 식을 사용하여 프로그램을 작성하면 최적 선형회귀를 위한 최적해를 구할 수 있습니다.  = 1.1240030938466818  = 7.223327947748378  x\_average는 age\_data의 평균입니다.  wo\_fraction은 w0의 분자입니다.  wo\_denominator은 w0의 분모입니다. |

**실습 03**

|  |
| --- |
| 그래프 (이미지 copy할 것) |
|  |

|  |
| --- |
| 설명 (본 실습 과제의 중요 이론 및 결과를 간략히 설명) |
| ‘plot’을 사용하여 훈련 데이터(age\_data,height\_data)를 빨간 점(People Data)로 표현하고, age\_data의 최솟값과 최댓값 사이 같은 간격을 갖는 원소 25개를 추출한 것(t)을 x로 잡고(t = np.linspace(min(age\_data),max(age\_data),25)를 사용) 해석해로 구한 선형 모델(y\_hat)을 y로 잡아서 파란색 선(Regression)으로 설정하여 그래프로 구현하였습니다. |

**실습 04**

|  |  |
| --- | --- |
| 코드 (텍스트 형태로 copy할 것, 폰트크기 9pt) | |
| import numpy as np  import matplotlib.pyplot as plt  import pandas as pd  raw\_data = pd.read\_csv('lin\_regression\_data\_01.csv',names = ['x','y'])  age\_data = np.asarray(raw\_data['x'].values.tolist())  height\_data = np.asarray(raw\_data['y'].values.tolist())  x\_average = sum(age\_data)/len(age\_data)  w0\_fraction = sum(height\_data\*(age\_data - x\_average))/len(age\_data)  w0\_denominator = sum((age\_data\*age\_data)-(x\_average\*x\_average))/len(age\_data)  w0 = w0\_fraction/w0\_denominator  w1 = sum(((height\_data) - (w0\*age\_data)))/len(age\_data) | y\_hat\_m = w0\*age\_data + w1  MSE = sum((y\_hat\_m-height\_data)\*\*2)/len(height\_data)  print(MSE) |

|  |
| --- |
| 평균제곱오차 및 설명 (본 실습 과제의 중요 이론 및 결과를 간략히 설명) |
| 평균 제곱오차(MSE)란 각각의 **선형모델의 값과 실제 값의 차이를 제곱한 값의 평균**입니다.  평균 제곱오차(MSE) 값이 최소화 일 때, 선형모델이 최적의 선형모델이라고 할 수 있으며 이는 다항 선형회귀 모델의 해석해를 구할 때 사용됩니다.  해석해로 구한 선형 모델의 평균제곱오차(MSE)의 값 = 1.8631967487108971 입니다.  y\_hat\_m은 해석해로 구한 선형모델의 x값을 age\_data로 잡은 선형모델입니다.  MSE는 y\_hat\_m과 height\_data의 차를 제곱한 값들의 평균으로 설정했습니다. |

**실습 05**

|  |  |
| --- | --- |
| 코드 (텍스트 형태로 copy할 것, 폰트크기 9pt) | |
| import numpy as np  import matplotlib.pyplot as plt  import pandas as pd  raw\_data = pd.read\_csv('lin\_regression\_data\_01.csv',names = ['x','y'])  age\_data = np.asarray(raw\_data['x'].values.tolist())  height\_data = np.asarray(raw\_data['y'].values.tolist())  w0\_first = 0  w1\_first = 0  LR = 0.0043 | for i in range(10000):    error = w0\_first\*age\_data + w1\_first - height\_data    w0\_diff = 2/len(age\_data)\*(sum(age\_data\*error))  w1\_diff = 2/len(age\_data)\*(sum(error))    w0\_first = w0\_first - LR\*w0\_diff  w1\_first = w1\_first - LR\*w1\_diff |

**실습 06**

|  |
| --- |
| 결과물 및 설명 (본 실습 과제의 중요 이론 및 결과를 간략히 설명) |
| 학습률(LR) = 0.0043 , 초기값(w0\_first, w1\_first) = 0, 반복 횟수 = 10000 으로 하여 최적 매개변수를 찾아보았습니다.  그 결과 최종 평균제곱오차는 ‘1.863201575408113’  최적 매개변수는 ’ = 1.1246943053177825 = 7.215551203808762 ‘가 나왔습니다.  error는 경사하강법을 이용해 만든 선형모델과(w0\_first\*age\_data + w1\_first) 실제 모델(height\_data)의 차이입니다.  w0\_diff는 w0를 미분하여 구한 기울기 값입니다.  w1\_diff은 w1을 미분하여 구한 기울기 값입니다.  ‘w0\_first = w0\_first - LR\*w0\_diff’은 w0[t+1]을 구하는 과정입니다.  ‘w1\_first = w1\_first - LR\*w1\_diff’은 w1[t+1]을 구하는 과정입니다. |

**실습 07**

|  |
| --- |
| 그래프 (이미지 copy할 것) |
|  |

|  |
| --- |
| 설명 (본 실습 과제의 중요 이론 및 결과를 간략히 설명) |
| 경사하강법의 초기값, 학습률, 반복 횟수 코드는 [실습 6]과 같은 코드를 사용하여 만들었습니다.  프로그램 실습 결과 반복 회수가 증가할 수록 매개변수 값은 최적 매개변수의 값과 가까워지고, 평균제곱오차(MSE)의 값은 점점 감소하면서 최적해를 이용하여 구한 평균제곱오차와 비슷해짐을 확인할 수 있었습니다.  [실습 6]에서 추가된 코드로는 10000번을 반복하면서 변화되는 매개변수와 MSE 값을 전부 확인하기 위하여  ‘de\_wo = [0]\*10000, de\_w1 = [0]\*10000, de\_MSE = [0]\*10000’의 배열 리스트를 만든다음 for문 안에 한번씩 반복될 때 마다 w0,w1,MSE의 값을 넣어줌으로써 해결하였습니다.  그 다음 ‘plot’을 이용하여 x축은 0 ~ 10000까지 같은 간격을 같는 10000개의 원소 t로 설정하였고(t = np.linspace(0,10000,10000)), y축은 ‘de\_w0’을 빨간 (점+선)(w0), ‘de\_w1’을 파란 (점+선)(w1)으로 figure(1)그래프를 구현하였습니다.  figure(2) 그래프의 x축은 위에서 사용했던 t로 설정하고 y축은 ‘de\_MSE’ 빨간 (점+선)으로 구현하였습니다. |

**실습 08**

|  |  |
| --- | --- |
| 코드 (텍스트 형태로 copy할 것, 폰트크기 9pt) | |
| import numpy as np  import matplotlib.pyplot as plt  import pandas as pd  raw\_data = pd.read\_csv('lin\_regression\_data\_01.csv',names = ['x','y'])  age\_data = np.asarray(raw\_data['x'].values.tolist())  height\_data = np.asarray(raw\_data['y'].values.tolist())  x\_average = sum(age\_data)/len(age\_data)  w0\_fraction = sum(height\_data\*(age\_data - x\_average))/len(age\_data)  w0\_denominator = sum((age\_data\*age\_data)-(x\_average\*x\_average))/len(age\_data)  w0 = w0\_fraction/w0\_denominator  w1 = sum(((height\_data) - (w0\*age\_data)))/len(age\_data)  t = np.linspace(min(age\_data),max(age\_data),25)  y\_hat = w0\*t + w1 | w0\_first = 0  w1\_first = 0  LR = 0.0043  for i in range(10000):    error = w0\_first\*age\_data + w1\_first - height\_data    w0\_diff = 2/len(age\_data)\*(sum(age\_data\*error))  w1\_diff = 2/len(age\_data)\*(sum(error))    w0\_first = w0\_first - LR\*w0\_diff  w1\_first = w1\_first - LR\*w1\_diff  t\_de = np.linspace(min(age\_data),max(age\_data),25)  y\_hat\_de = w0\_first\*t\_de + w1\_first  plt.plot(age\_data,height\_data,'ro',t,y\_hat,'b-',t\_de,y\_hat\_de,'g-')  plt.legend(['People Data','Regression','Descent'])  plt.xlabel('age')  plt.ylabel('height')  plt.grid(True)  plt.show() |

|  |
| --- |
| 그래프 (이미지 copy할 것) |
|  |

|  |
| --- |
| 설명 (본 실습 과제의 중요 이론 및 결과를 간략히 설명) |
| 훈련 데이터(age\_data,height\_data)를 빨간 점(People Data)로 표현하고, 해석해를 이용해 구한 회귀모델(y\_hat)은 파란색 선(Regrsstion)으로 표현하고, 경사하강법을 이용해 구한 회귀모델(y\_hat\_de)은 초록색 선(Descent)로 표현하여 ‘plot’을 통해 그래프로 구현하였습니다.  훈련데이터와 해석해를 이용해 구한 회귀모델은 [실습 3]에서 설정한 방법대로 코딩 하였고 경사하강법을 이용해 구한 회귀모델은 x축을 ‘t\_de’(age\_data의 최솟값과 최댓값 사이 같은 간격을 갖는 원소 25개를 추출한 것)으로 잡고, y축을 ‘y\_hat\_de’(경사하강법을 이용한 선형모델의 x값에 t\_de를 넣은 것)로 설정하였습니다. |

**실습 09**

|  |  |
| --- | --- |
| 코드 (텍스트 형태로 copy할 것, 폰트크기 9pt) | |
| import numpy as np  import matplotlib.pyplot as plt  import pandas as pd  raw\_data = pd.read\_csv('lin\_regression\_data\_01.csv',names = ['x','y'])  age\_data = np.asarray(raw\_data['x'].values.tolist())  height\_data = np.asarray(raw\_data['y'].values.tolist())  t = np.linspace(min(age\_data),max(age\_data),25)  y\_data = np.zeros(25)  for i in range(25):  y\_data[i] = height\_data[i]  Fake\_GK = np.ones((25,1), int) | K = [3,5,8]  total\_G\_Par = [0]\*len(K)  for i in range(len(K)):  G\_Par = np.zeros(K[i])  Mk = np.zeros(K[i])  Guass = np.zeros((K[i],25))  Guass1 = np.zeros((K[i],25))  Final\_Guass\_y\_hat = np.zeros(25)  dis = (max(age\_data) - min(age\_data))/(K[i]-1)  for k in range(K[i]):  Mk[k] = min(age\_data) + dis\*k  Guass[k] = np.exp(-0.5\*((age\_data-Mk[k])/(dis))\*\*2)  Guass1[k] = np.exp(-0.5\*((t-Mk[k])/(dis))\*\*2)  RGuass = np.concatenate([Guass.transpose(), Fake\_GK],1)  G\_Par = (np.linalg.inv(RGuass.transpose().dot(RGuass))).dot(RGuass.transpose()).dot(y\_data.reshape(25,1))  total\_G\_Par[i] = G\_Par  print(total\_G\_Par) |

|  |
| --- |
| 매개변수 및 설명 (본 실습 과제의 중요 이론 및 결과를 간략히 설명) |
| K= 3일 때 w0 = 0.99557866, w1 = 5.90764664, w2 = 13.14040901, w3 = 5.34004445  K = 5일 때 w0 = -25.85546567, w1 = 5.27827628, w2 = -18.94373778, w3 = 6.19309345, w4 = -16.13153749, w5 = 37.26402946  K = 8일 때 w0 = -1.13831218, w1 = 1.56672026, w2 = 2.8861904, w3 = 2.49417908, w4 = 2.37521706, w5 = 9.31513781, w6 = -2.26629064, w7 = 11.36272625, w8 = 11.17435832 가 나왔습니다.  y\_data[i]는 행이 25, 열이 1인 행렬에 height\_data를 넣어 출력 데이터를 만들어 주었습니다.  Fake\_GK는 Wm을 표현하기 위해 필요한 더미 값(항상 1인 값)들을 나타내는 값입니다.  G\_Par은 매개변수의 값들을 나타냅니다. 매개변수는 (RGuass의 전치행렬과 RGuass값의 곱)을 역행렬 해준다음, 그 식을 RGuass 전치행렬과 곱해주고 그 다음 25행 1열인 y\_data의 배열과 곱해주었습니다.  Mk는 k번째 가우스 함수의 평균을 나타냅니다.  Guass는 x를 age\_data로 설정하여 표현한 기저 함수입니다.  Guass1은 x를 [x값의 최소 ~ x값의 최대를 균일하게 25개로 나눈 x값]으로 설정한 기저 함수입니다.  RGuass는 기저 함수(Guass)와 더미 변수(Fake\_GK)를 합친 행렬로 매개변수를 구하기 위한 최종 기저함수 행렬을 의미합니다.  total\_G\_Par은 K가 3일 때, 5일 때, 8일 때의 모든 매개변수들을 저장하는 공간을 의미합니다. |

**실습 10**

|  |
| --- |
| 그래프 (이미지 copy할 것) |
|  |

|  |
| --- |
| 설명 (본 실습 과제의 중요 이론 및 결과를 간략히 설명) |
| 첫번째 사진(왼쪽 상단)은 K=3일 때, 두번째 사진(오른쪽 상단)은 K=5일 때, 세번째 사진(왼쪽 하단)은 K=8일 때, 네번째 사진(오른쪽 하단)은 K=10일 때의 그래프 모습입니다.  훈련 데이터(age\_data,height\_data)를 빨간 점(People Data)로 표현하고, 선형 기저함수 회귀 모델(sum(Guass\_y\_hat))을 파란 선(K=3,5,8,10)로 표현하여 ‘plot’을 통해 그래프로 구현하였습니다.  K 값이 커질수록 정확한 선형 기저함수 회귀 모델이 됨을 알 수 있습니다. |

**실습 11**

|  |  |
| --- | --- |
| 코드 (텍스트 형태로 copy할 것, 폰트크기 9pt) | |
| import numpy as np  import matplotlib.pyplot as plt  import pandas as pd  raw\_data = pd.read\_csv('lin\_regression\_data\_01.csv',names = ['x','y'])  age\_data = np.asarray(raw\_data['x'].values.tolist())  height\_data = np.asarray(raw\_data['y'].values.tolist())  t = np.linspace(min(age\_data),max(age\_data),25)  y\_data = np.zeros(25)  for i in range(25):  y\_data[i] = height\_data[i]  Fake\_GK = np.ones((25,1), int) | K = [3,5,8,10]  MSE = np.zeros(4)  for b in range(4):  G\_Par = np.zeros(K[b])  Mk = np.zeros(K[b])  Guass = np.zeros((K[b],25))  Guass1 = np.zeros((K[b],25))  dis = (max(age\_data) - min(age\_data))/(K[b]-1)  for k in range(K[b]):  Mk[k] = min(age\_data) + dis\*k  Guass[k] = np.exp(-0.5\*((age\_data-Mk[k])/(dis))\*\*2)  Guass1[k] = np.exp(-0.5\*((t-Mk[k])/(dis))\*\*2)  RGuass = np.concatenate([Guass.transpose(), Fake\_GK],1)  G\_Par = (np.linalg.inv(RGuass.transpose().dot(RGuass))).dot(RGuass.transpose()).dot(y\_data.reshape(25,1))  Guass\_y\_hat = (G\_Par)\*(np.concatenate([Guass,Fake\_GK.transpose()]))  MSE[b] = sum((sum(Guass\_y\_hat) - height\_data)\*\*2)/len(height\_data)  plt.plot(K,MSE,'ro')  plt.legend(['Data'])  plt.xlabel('K')  plt.ylabel('MSE')  plt.grid(True)  plt.show() |

|  |
| --- |
| 그래프 (이미지 copy할 것) o |
|  |

|  |
| --- |
| 설명 (본 실습 과제의 중요 이론 및 결과를 간략히 설명) |
| [실습 9 ~11]에서 설명한 코드내 변수 설명은 생략하겠습니다.  MSE는 각 K에서 나온 MSE값을 넣기 위하여 1행 4열의 배열을 만들어 주었습니다.  K값이 총 4개(3,5,8,10) 이므로 ‘for문’을 사용하여 총 4번을 반복해주고 for문 내에서 선형 기저함수 회귀 모델에서 필요한 매개변수를 구하는 과정은 먼저 K의 원소의 값만큼 반복해서 기저함수를 구해야 하므로 ‘for’문을 사용하여 총 K의 원소의 값만큼 반복해서 기저함수를 구하였습니다.  그 다음 매개변수를 구하기 위해 실습[9]번에서 설명한 G\_par 에다 매개변수 구하는 식을 적어서 매개변수를 구하였습니다.  MSE는 선형 기저함수 모델의 출력값인(sum(y\_hat))과 실제 값(height\_data)의 차를 제곱한 값들의 평균을 구하여 나타냈습니다.  그래프는 ‘plot’을 사용하여 x값은 K, y값은 MSE, 빨간 점(Data)로 그래프를 구현하였습니다. |