|  |  |
| --- | --- |
| 교육제목 | 데이터 기반 인공지능 시스템 엔지니어 양성 과정 |
| 교육일시 | 210930 |
| 교육장소 | 자택(디스코드 사용 온라인 학습) |
| **교육내용** | |
| 오전 | 1. 데이터 살펴보기   import pandas as pd  df = pd.read\_csv("./auto-mpg.csv", header=None) # 자료에 컬럼명이없다  df.columns = ['mpg','cylinders','displacement','horsepower',  'weight','acceleration','model\_year','origin','name']# 컬럼명추가  print(df.head(3),'\n')  print(df.tail())  print(df.shape) # 데이터프레임의 차원을 보여줌 / 결과물 : 튜플  # 데이터 프레임의 행의 갯수와 열의 갯수를 출력  print("rows : {}, columns : {} ".format(df.shape[0],df.shape[1]))  df.info() # 데이터의 정보 : 컬럼의 갯수, 각 컬럼의 row갯수, 데이터 타입  print(df.describe()) # df.describe():return data type = DataFrame  df\_desc = df.describe().loc[['count','std'],['mpg','weight']]  df\_desc  print(df.count()) # 전체 컬럼들의 갯수  df['name'].value\_counts() # 같은 값들의 갯수  #df.value\_counts()  # 평균값, df.mean() : 전체 컬럼들의 평균  # df["컬럼명"] : 해당 컬럼의 평균  # mean, median, min, max, std, corr(열 이름 리스트)을 각각 구해보기 std:표준편차,corr:상관계수  df.mean() df["mpg"].mean()  df.std() df["mpg"].std  df.[["mpg","weight"]].corr   * 판다스 내장 그래프 활용   df = pd.read\_excel("./남북한발전전력량.xlsx")  df\_ns = df.iloc[[0,5],3:] # 0(남한).1~4(NaN),5(북한)  df\_ns.index = ['South','North'] # 0을 남한, 5를 북한으로 바꾸기  #df.columns # column이 문자니까 숫자로 바꿔주기  df\_ns.columns = df\_ns.columns.map(int) # 컬럼명을 int로 변경  # 데이터 전처리 : 컬럼의 데이터 타입 변경, 원하는 정보만 추출  # 데이터 가공, NaN 데이터 처리 ....  # df\_ns.plot()  df\_ns\_t = df\_ns.T #DataFrame의 행과 열을 바꿈  df\_ns.T.plot(kind='line') # kind = line(Default),bar,barh,hist,box,ked,area,pie,scatter,hexbin  df\_ns.T.plot(kind='bar')  df\_ns.T.plot(kind='hist')  df = pd.read\_csv("./auto-mpg.csv", header=None)  df.columns = ['mpg','cylinders','displacement','horsepower',  'weight','acceleration','model\_year','origin','name']  df[['mpg','cylinders']].plot(kind='box')  df.plot(x='weight', y='mpg',kind='scatter')   1. 시각화 도구   Matplotlib : 기본 그래프 도구  import pandas as pd  import matplotlib.pyplot as plt  plt.rcParams['font.family'] = 'Malgun Gothic' # 한글 폰트 설정  plt.rcParams['axes.unicode\_minus']=False  # -- 데이터 전처리 작업 --  df = pd.read\_excel('./시도별 전출입 인구수.xlsx', header=0)  df = df.fillna(method='ffill') # 전출지 누락된 데이터를 앞 데이터로 채움  df.head()  #서울에서 다른 지역으로 이동한 데이터만 추출해서 정리  mask = (df['전출지별'] == '서울특별시') & (df['전입지별'] != '서울특별시')  #파이썬은 and, 판다스는 &  # True/False 체크  df\_seoul = df[mask] # mask를 이용하여 True인 데이터만 추출  df\_seoul = df\_seoul.drop(['전출지별'],axis=1) # 전출지별 컬럼 삭제  df\_seoul.rename({'전입지별':'전입지'}, axis=1, inplace=True) # 전입지별을 전입지로 변환(컬럼명 변경)  df\_seoul.set\_index('전입지',inplace=True) # 전입지 컬럼을 인덱스로 설정  df\_seoul.plot()  df\_1 = df\_seoul.loc['경기도'] # x축과 y축으로 구성됨  plt.plot(df\_1.index , df\_1.values, marker='o') # plt.plot(x축,y축)  plt.style.use('ggplot')  plt.title("서울에서 경기도로 전출한 인구수", size = 20) # 제목 설정  plt.xlabel("연도", size=20) # x축 라벨 설정  plt.ylabel("인구수", size=20) # y축 라벨 설정  plt.xticks(rotation='120') # x라벨의 기울기  plt.legend(labels=["서울->경기"], loc='center') # 타이틀 주기 leg end  plt.show()  # 부산에서 다른지역으로 전출한 데이터만 추출해서 df\_busan으로 저장  # 불필요한 컬럼 제거, 인덱스는 전입지별을 전입지로 변경해서 설정.  mask = (df['전출지별']=='부산광역시')&(df['전입지별']!='부산광역시')  df\_busan = df[mask]  df\_busan = df\_busan.drop(['전출지별'],axis=1) #불필요한 컬럼 제거  df\_busan.rename({'전입지별':'전입지'},axis=1, inplace=True)  df\_busan.set\_index('전입지', inplace=True)  df\_busan.plot() |
| 오후 | < excel파일로 그래프 만들기>  # %load graph.py << 미리 만들어둔 graph.py 로드하기  #!/usr/bin/env python  # In[ ]:  import pandas as pd  import matplotlib.pyplot as plt  plt.rcParams['font.family'] = 'Malgun Gothic' # 한글 폰트 설정  plt.rcParams['axes.unicode\_minus']=False  # Excel 데이터를 데이터프레임 변환  df = pd.read\_excel('./남북한발전전력량.xlsx', engine= 'openpyxl', convert\_float=True)  df = df.loc[5:9] # 인덱스(행) 5번부터 8번까지 가져옴.  df.drop('전력량 (억㎾h)', axis='columns', inplace=True)  df.set\_index('발전 전력별', inplace=True)  df = df.T  # 증감율(변동률) 계산  df = df.rename(columns={'합계':'총발전량'}) # 컬럼명을 합계에서 총발전량으로 변경  df['총발전량 - 1년'] = df['총발전량'].shift(1) # shift(n) 왼쪽으로 n칸씩 민다  df['증감율'] = ((df['총발전량'] / df['총발전량 - 1년']) - 1) \* 100  df.head()  # 2축 그래프 그리기  ax1 = df[['수력','화력']].plot(kind='bar', figsize=(20, 10), width=0.7, stacked=True)  ax2 = ax1.twinx()  ax2.plot(df.index, df.증감율, ls='--', marker='o', markersize=20, color='green', label='전년대비 증감율(%)')  # ls : line style  ax1.set\_ylim(0, 500)  ax2.set\_ylim(-50, 50)  ax1.set\_xlabel('연도', size=20)  ax1.set\_ylabel('발전량(억 KWh)')  ax2.set\_ylabel('전년 대비 증감율(%)')  plt.title('북한 전력 발전량 (1990 ~ 2016)', size=30)  ax1.legend(loc='upper left')  plt.show()  <seaborn의 데이터셋으로 그래프 만들기>  # 라이브러리 불러오기  import matplotlib.pyplot as plt  import seaborn as sns    # Seaborn 제공 데이터셋 가져오기  titanic = sns.load\_dataset('titanic')    # 스타일 테마 설정 (5가지: darkgrid, whitegrid, dark, white, ticks)  sns.set\_style('darkgrid')  # 그래프 객체 생성 (figure에 2개의 서브 플롯을 생성)  fig = plt.figure(figsize=(15, 5))  ax1 = fig.add\_subplot(1, 2, 1)  ax2 = fig.add\_subplot(1, 2, 2)    # 그래프 그리기 - 선형회귀선 표시(fit\_reg=True)  sns.regplot(x='age', #x축 변수  y='fare', #y축 변수  data=titanic, #데이터  ax=ax1) #axe 객체 - 1번째 그래프  # 그래프 그리기 - 선형회귀선 미표시(fit\_reg=False)  sns.regplot(x='age', #x축 변수  y='fare', #y축 변수  data=titanic, #데이터  ax=ax2, #axe 객체 - 2번째 그래프  fit\_reg=False) #회귀선 미표시  plt.show()  <folium 라이브러리의 지도 활용하기>  import folium  seoul\_map = folium.Map(location=[37.55,126.98], zoom\_start=12)  seoul\_map.save('./seoul1.html')  seoul\_map2 = folium.Map(location=[37.55,126.98], titles='stamen Terrain',zoom\_start=12)  seoul\_map3 = folium.Map(location=[37.55,126.98], titles='stamen Toner',zoom\_start=15)  seoul\_map2.save('./seoul2.html')  seoul\_map3.save('./seoul3.html')  seoul\_map3  #문) iris 데이터를 sns모듈에서 load하여  #종별로 잎의 넓이와 길이를 산점도와 선그래프로 표시  # legend(종)와 제목('붓꽃의 잎의 길이와 넓이, 종의 관계')도 표시  import seaborn as sns  def scatter\_graph(X, Y):  sns.scatterplot(x=X, y=Y)    iris = sns.load\_dataset('iris')  iris.set\_index('species', inplace=True)  x\_data = iris.index  y\_data = iris['petal\_width']  scatter\_graph(x\_data, y\_data) # 잎의 넓이를 산점도(scatter)로  plt.title('붓꽃의 잎의 길이와 넓이, 종의 관계') # title  labels = iris[['petal\_width', 'petal\_length']]  plt.legend(labels) # 범주  plt.show() |