|  |  |
| --- | --- |
| 교육 제목 | python 8 |
| 교육 일시 | 2021. 09. 30 |
| 교육 장소 | 영우글로벌러닝 |
| **교육 내용** | |
| 오전 | import pandas as pd  df=pd.read\_csv("./auto-mpg.csv", header=None)  df.columns=['mpg','cylinders','displacement',  'horespower', 'weight', 'ecceleration',  'model\_year', 'origin', 'name']  print(df.head())  print("\n")  print(df.tail())  ----------------------------------------  print(df.shape) # 데이터프레임의 차원을 보여줌  # 데이터프레임의 행의 갯수와 열의 갯수를 출력  print("rows : {}, columns : {}".format(df.shape[0], df.shape[1]))  df.info() # 데이터의 정보 : 컬럼의 갯수, 각 columns의 row의 갯수  -------------------------------------  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['mpg'].value\_counts() # 같은 값들의 갯수를 출력합니다.  # df.value\_counts()  ---------------------------------------  df\_mpg=pd.read\_csv("./auto-mpg.csv", header=None)  df\_mpg.columns=['mpg','cylinders','displacement',  'horespower', 'weight', 'ecceleration',  'model\_year', 'origin', 'name']  df\_mpg[['mpg','cylinders']].plot(kind='box')  ----------------------------------------------------------------------------  # 판다스 내장 그래프 활용  df=pd.read\_excel("./남북한발전전력량.xlsx")  df\_ns=df.iloc[[0,5], 3:]  df\_ns.index=['South','North']  df\_ns.columns=df\_ns.columns.map(int) # 컬럼명을 int로 변경  df\_ns.columns  # 데이터 전처리 : 컬럼의 데이터 타입 변경, 원하는 정보만 추출  # 테이터 가공, NaN 데이터 처리  ----------------------------------------  # df\_ns.plot()  df\_ns\_t=df\_ns.T # 데이터프레임의 행과 열을 바꿈  #df\_ns\_t.plot()  df\_ns\_t.plot(kind='line')  ------------------------------------------  df\_ns\_t = df\_ns.T # 데이터프레임의 행과 열을 바꿈  df\_ns\_t.plot(kind='line')  df\_ns\_t.plot(kind='bar')  df\_ns\_t.plot(kind='hist')  df\_mpg = pd.read\_csv("./auto-mpg.csv", header=None)  df\_mpg.columns = ['mpg','cylinders','displacement',  'horsepower','weight', 'acceleration',  'model\_year','origin','name']  df\_mpg[['mpg','cylinders']].plot(kind='box')  df\_mpg.plot(x="weight", y="mpg", kind="scatter")  -----------------------------------------------------------------------------  import pandas as pd  import matplotlib.pyplot as pit  from matplotlib import font\_manager, rc  font\_path="./malgun.ttf"  font\_name=font\_manager.FontProperties(fname=font\_path).get\_name()  rc('font', family=font\_name)  df=pd.read\_excel("./시도별 전출입 인구수.xlsx", header=0)  #print(df.head())  #df=df.fillna(value=0)  # 데이터 전처리 작업  df=df.fillna(method='ffill') # 전출지 누락된 데이터를 앞 데이터로 채움  df.head()  # 서울에서 다른 지역으로 이동한 데이터만 추출하여 정리  mask=(df['전출지별']=='서울특별시') & (df['전입지별']!='서울특별시')  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) # 컬럼명 변경  df\_seoul.set\_index('전입지', inplace=True) # '전입지' 컬럼을 인덱스로 설정  # df\_seoul  df\_seoul.T.plot()  ---------------------------------------  mask=(df['전출지별']=='부산광역시') & (df['전입지별']!='부산광역시')  df\_busan=df[mask] # True or False 구별, True만 추출  df\_busan=df\_busan.drop(['전출지별'], axis=1)  df\_busan.rename({'전입지별':'전입지'}, axis=1, inplace=True)  df\_busan  --------------------------------------  df\_1=df\_seoul.loc['경기도'] # x축과 y축으로 구성됨  plt.style.use('ggplot')  plt.plot(df\_1.index, df\_1.values, marker='o') # plt.plot(x축, y축)  plt.title("서울에서 경기도로 전출한 인구수", size=20) # 제목  plt.xlabel("연도") # x축 라벨  plt.ylabel("인구수") # y축 라벨  #plt.xticks(rotation="vertical", np.arange(1970, 2017, step=10))  #pit.xticks(rotation=45) # x라벨의 기울기  plt.legend(labels=["서울->경기"], loc='upper right')  plt.show() |
| 오후 | # %load "a.py"  # !/usr/bin/env python  # in[]:  # %load 파일명.py 해당 파일의 코딩 내용을 본 프로그램으로 불러옴  import pandas as pd  import matplotlib.pyplot as plt  from matplotlib import font\_manager, rc  font\_path="./malgun.ttf"  font\_name=font\_manager.FontProperties(fname=font\_path).get\_name()  rc('font', family=font\_name)  --------------------------------  # 서울에서 충청남도, 경상북도, 강원도로 이동한 인구 데이터만 선택해서 사용  df=pd.read\_excel("./시도별 전출입 인구수.xlsx", engine='openpyxl', header=0)  # NaN 데이터 처리 fillna() 함수 이용, 이전 행의 자료로 변경  df.fillna(method='ffill')  # 서울에서 충청남도, 경상북도, 강원도로 이동한 데이터만 추출  #mask=(df['전출지별']=='서울특별시')&((df['전입지별']=='경상북도') |\  # (df['전입지별']=='충청남도') | (df['전입지별']=='강원도'))  mask = (df['전출지별'] == '서울특별시') & (df['전입지별'] != '서울특별시')  df\_data=df[mask]  # 년도가 string이므로 int로 변경  df\_data.drop(['전출지별'], inplace=True, axis=1)  df\_data.rename({'전입지별':'전입지'}, axis=1, inplace=True)  # df\_data['ye']  df\_data.set\_index(['전입지별'], inplace=True)  # df\_data.columns.map(int)  # 인덱스는 년도  ------------------------------  df\_t=df\_data.T  fig=plt.figure(figsize=(20,5))  ax=fig.add\_subplot(1, 1, 1)  ax.plot(df\_t.index, df\_t["충청남도"], maker='o', color='red')  ax.plot(df\_t.index, df\_t["경상북도"], maker='+', color='b')  ax.plot(df\_t.index, df\_t["강원도"], maker='.', color='black')  ax.set\_title("서울-> 충남, 경북, 강원 이동 인구", size=20)  ax.set\_xlabel("기간")  ax.set\_ylabel("인구수")  legend\_labels = ["충청남도","경상북도","강원도"]  ax.legend(legend\_labels, loc='best', fontsize=15)  plt.show()  ----------------------------  col\_years=list(map(str,range(1970, 2018)))  df\_4=df\_seoul.loc[['충청남도','경상북도','강원도','전라남도'], col\_years]  plt.style.use('ggplot')  mask=(df['전출지별']=='서울특별시') & (df['전입지별']!='서울특별시')  df\_seoul=df[mask]  df\_seoul=df\_seoul.drop(['전출지별'], axis=1)  df\_seoul.rename({'전입지별': '전입지'}, axis=1, inplace=True)  df\_seoul.set\_index('전입지', inplace=True)  fig=plt.figure(figsize=(20,10))  ax1=fig.add\_subplot(2, 2, 1) # add\_subplot(행의 갯수, 열의 갯수,)  ax2=fig.add\_subplot(2, 2, 2)  ax3=fig.add\_subplot(2, 2, 3)  ax4=fig.add\_subplot(2, 2, 4)  ax1.plot(df\_t.index, df\_t["충청남도"], maker='o', color='red')  ax2.plot(df\_t.index, df\_t["경상북도"], maker='+', color='b')  ax3.plot(df\_t.index, df\_t["강원도"], maker='.', color='black')  ax4.plot(df\_t.index, df\_t["전라남도"], maker='o', color='olive')  ax1.set\_title("서울->충남", size=20)  ax2.set\_title("서울->경북", size=20)  ax3.set\_title("서울->강원", size=20)  ax4.set\_title("서울->전남", size=20)  ax.set\_xlabel("기간")  ax.set\_ylabel("인구수")  plt.show()  -------------------------------  fig1=pit.figure(figsize=(20, 10))  df\_4=df\_4.T  df\_4.plot(kind='area', stacked=True, alpha=0,2, figsize=(20,10))  pit.title('서울->타시도 인구 이동', size=30)  pit.ylabel('이동 인구 수', size=20)  pit.xlabel('기간', size=20)  pit.legend(loc='best', fontsize=15)  ----------------------------------------------------------------------------  # -\*- coding: utf-8 -\*-  # 라이브러리 불러오기  import pandas as pd  import matplotlib.pyplot as plt  # matplotlib 한글 폰트 오류 문제 해결  from matplotlib import font\_manager, rc  font\_path = "./malgun.ttf" #폰트파일의 위치  font\_name = font\_manager.FontProperties(fname=font\_path).get\_name()  rc('font', family=font\_name)  plt.style.use('ggplot') # 스타일 서식 지정  plt.rcParams['axes.unicode\_minus']=False # 마이너스 부호 출력 설정  # Excel 데이터를 데이터프레임 변환  df = pd.read\_excel('./남북한발전전력량.xlsx', engine= 'openpyxl', convert\_float=True)  df = df.loc[5:9]  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'})  df['총발전량 - 1년'] = df['총발전량'].shift(1)  df['증감율'] = ((df['총발전량'] / df['총발전량 - 1년']) - 1) \* 100  # 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(%)')  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()  --------------------------------------------------------------------------  # 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('whitegrid')  # 그래프 객체 생성 (figure에 3개의 서브 플롯을 생성)  fig = plt.figure(figsize=(15, 5))  ax1 = fig.add\_subplot(1, 3, 1)  ax2 = fig.add\_subplot(1, 3, 2)  ax3 = fig.add\_subplot(1, 3, 3)    # x축, y축에 변수 할당  sns.barplot(x='sex', y='survived', data=titanic, ax=ax1)  # x축, y축에 변수 할당하고 hue 옵션 추가  sns.barplot(x='sex', y='survived', hue='class', data=titanic, ax=ax2)  # x축, y축에 변수 할당하고 hue 옵션을 추가하여 누적 출력  sns.barplot(x='sex', y='survived', hue='class', dodge=False, data=titanic, ax=ax3)  # 차트 제목 표시  ax1.set\_title('titanic survived - sex')  ax2.set\_title('titanic survived - sex/class')  ax3.set\_title('titanic survived - sex/class(stacked)')  plt.show()  --------------------------------------------------------------------------  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()  -----------------------------------------------------------------------------  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 산점도로  y\_data=iris["petal\_lenght"]  scatter\_graph(x\_data, y\_data) # 잎의 길이를 산점도로  plt.title("붓꽃의 잎의 길이와 넓이, 종의 관계") # title  labels=iris[["petal\_width", "petal\_length"]]  plt.legend(labels) # 범주  plt.show() |