|  |  |
| --- | --- |
| 교육 제목 | python 9 |
| 교육 일시 | 2021. 10. 01 |
| 교육 장소 | 영우글로벌러닝 |
| **교육 내용** | |
| 오전 | # 라이브러리 불러오기  import seaborn as sns  # titanic 데이터셋 가져오기  df=sns.load\_dataset('titanic')  ---  nan\_deck=df['deck'].value\_counts(dropna=False)  print(nan\_deck)  ---  # isnumm : NaN이면 True, notnull : 값이 존재하면 데이터의 값, NaN 반환  # 누락된 데이터 처리  # 제거, 치환  import seaborn as sns  # titanic 데이터셋 가져오기  df=sns.load\_dataset('titanic')  # 누락 데이터 확인의 과정  print(df.head().isnull())  print(df['deck'].head().isnull())  print("--------")  print(df.head().notnull) # 값이 존재하면 값을, 없으면 NaN을 출력  print("--------")  print(df.isnull().sum(axis=0)) # 누락된 데이터가 있는지 확인  ----------  # 누락된 자료가 500개 이상인 자료를 제거  df\_1=df.dropna(axis=1, thresh=500)  "deck" in (df\_1.columns)  ----------  df\_2=df.dropna(subset=['age'], how='any', axis=0)  print(len(df\_2))  len(df) - df['age'].isnull().sum(axis=0)  ----------------  # age 열의 NaN값을 다른 나이 데이터의 평균으로 변경하기  df\_1=df.dropna(axis=1, thresh=500)  # age열에 NaN 값이 있으면 행을 제거하는 방법  df\_2=df.dropna(subset=['age'], how='any', axis=0)  # age열의 NaN값을 다른 나이 데이터의 평균으로 변경하기  mean\_age = df['age'].mean(axis=0)  df\_age=df['age'].fillna(mean\_age)  # 앞의 값으로 수정  df\_age\_1=df['age'].fillna(method='ffill')  # 가장 빈번하게 나오는 값으로 수정  most\_cnt=df['age'].value\_counts(dropna=True).idmax()  df\_2=df['age'].fillna(most\_cnt)  -------------------  print(df\_age.isnull().sum(axis=0))  len(df\_age)  ------------------------------------------------------------------  # 중복 데이터 처리  import pandas as pd  # 중복 데이터를 갖는 데이터프레임 만들기  df=pd.DataFrame({'c1':['a','a','b','a','b'],  'c2':[1,1,1,2,2],  'c3':[1,1,2,2,2]})  print(df)  print()  df\_dup=df.duplicated() # 행의 중복을 체크  print(df\_dup)  # 특정 열의 중복 체크  df['c1'].duplicated()  # 데이터프레임에서 중복 행을 제거  df2=df.drop\_duplicates()  print(df2)  # 특정 열을 기준으로 중복 행을 제거  df2=df.drop\_duplicates(subset=['c2','c3'])  df2  ------------------------------------------------------------  import pandas as pd  # read\_csv() 함수로 df 생성  df=pd.read\_csv('./auto-mpg.csv', header=None)  ---------  # 열 이름을 지정  df.columns=['mpg','cylinders','displacement',  'horsepower', 'weight', 'ecceleration',  'model\_year', 'origin', 'name']  df.head()  # mpg(mile per galion)를 kpl(kilometer per liter)로 변환 (mpg\_to\_kpl = 0.425)  mpg\_to\_kpl=1.60934/3.78541  # mpg 열에 0.425를 곱한 결과를 새로운 열(kpl)에 추가  df['kpl']=(df['mpg']\*mpg\_to\_kpl).round(2)  df.kpl.head(3)  -------------  1. 데이터 타입 확인, unique(), '?'-> NaN 으로 처리  df\_hp=df.copy()  df\_hp=df.horsepower.replace('?',np.nan, inplace=True) # '?'를 np.nan 으로 변경  2. NaN 데이터 확인 후 -> 처리 -> 0.0 값으로 치환  df\_hp['horsepower'].fillna('0.0', inplace=True)  df\_hp.dropna(subset=['horsepower'], axis=0, inplace=True) # 누락데이터 행 삭제  3. 데이터 타입 변경 -> float으로 변경  df\_hp['horsepower']=df\_hp[horsepower'].astype('float') # 문자형을 실수형으로  ---------------  # horsepower 컬럼에 대해서 순서대로 처리하시오  import numpy as np  df.horsepower.unique()  df\_hp=df.copy()  df.horsepower.replace('?',np.nan, inplace=True) # 쓰레기값 처리  df\_hp.head()  -------------------------  # category, 데이터 타입을 범주형 또는 문자형, 숫자형으로 워나는 형태로 변경  print(df\_hp['origin'].unique()) # 정수형 데이터를 문자로 변경  df\_hp['origin'].replace({1:'USA', 2:'EU', 3:'JAPAN'}, inplace=True)  print(df\_hp['origin'].unique())  print(df['origin'].dtypes)  df['origin']=df['origin'].astype('category')  print(df['origin'].dtypes)  df['origin']=df['origin'].astype('object')  print(df['origin'].dtypes)  -------------------------------------------------------------------------------  import pandas as pd  import numpy as np  df=pd.read\_csv('./auto-mpg.csv', header=None)  # 열 이름을 지정  df.columns=['mpg','cylinders','displacement',  'horsepower', 'weight', 'ecceleration',  'model\_year', 'origin', 'name']  # horsepower열의 누락 데이터 처리 ('?'삭제 후 실수형으로 변경)  df['horsepower'].replace('?', np.nan, inplace=True)  df.dropna(subset=['horsepower'], axis=0, inplace=True)  df['horsepower']=df['horsepower'].astype('float')  -----------  # 3개의 구간(bin)으로 나누어서 범위로 처리를 하려고 함  count, bin\_div=np.histogram(df['horsepower'], bins=3)  print(bin\_div)  bin\_names=['저출력', '보통출력', '고출력']  df['hp\_bin']=pd.cut(x=df['horsepower'], # 데이터 배열  bins=bin\_div, # 경계 리스트  labels=bin\_names, # bin 이름  include\_lowest=True  )  df[['horsepower','hp\_bin']].head()  -------------  horse\_dummies=pd.get\_dummies(df['hp\_bin'])  horse\_dummies.head(10)  df=pd.get\_dummies(df['hp\_bin'], prefix='hp')  df  ------------------------------------------------------------  # titanic을 로드해서 처리  # 'age' 컬럼으로 데이터 처리 bin을 4로 ['유아기','청소년','청년','장년']  import seaborn as sns  df=sns.load\_dataset('titanic')  df['age'].isnull().sum(axis=0) # NaN 데이터를 처리, 평균나이로 대체  age\_avg=(df['age'].sum()/df['age'].count()).round()  df['age'].fillna(value=age\_avg, axis=0, inplace=True)  # df['age']=df['age'].round().astype('int') # float을 int로  ------------  bin\_names=['유아기','청소년','청년','장년']  bin\_values=[0,20,40,60,100]  df['age\_bin']=pd.cut(x=df['age'], # 데이터 배열  bins=bin\_values, # 경계 리스트  labels=bin\_names, # bin 이름  include\_lowest=True  )  df[['age','age\_bin']].head(10)  pd.get\_dummies(df['age\_bin'], prefix='대') |
| 오후 | import pandas as pd  import numpy as np  df=pd.read\_csv('./auto-mpg.csv', header=None)  # 열 이름을 지정  df.columns=['mpg','cylinders','displacement',  'horsepower', 'weight', 'ecceleration',  'model\_year', 'origin', 'name']  # horsepower열의 누락 데이터 처리 ('?'삭제 후 실수형으로 변경)  df['horsepower'].replace('?', np.nan, inplace=True)  df.dropna(subset=['horsepower'], axis=0, inplace=True)  df['horsepower']=df['horsepower'].astype('float')  # sklearn 라이브러리 불러오기  from sklearn import preprocessing  # 전처리를 위한 encoder 객체 만들기  label\_encoder=preprocessing.LabelEncoder()  onehot\_encoder=preprocessing.OneHotEncoder()  # label encoder로 문자열 범주를 숫자형 범주로 변환  onehot\_labeled=label\_encoder.fit\_transform(df['hp\_bin'].head(15))  print(onehat\_labeled)  print(type(onehot\_labeled))  # 2차운 행렬로 변경  onehot\_reshaped=onehot\_labeled.reshape(len(onehot\_labeled), 1)  print(onehat\_reshaped)  print(type(onehot\_reshaped))  # 희소행렬로 변환  onehot\_fitted=onehot\_encoder.fit\_transform(onehot\_reshaped)  print(onehot\_fitted)  print(type(onehot\_fitted))  ----------------------------------------------------------------------------------  from sklearn.datasets import load\_iris  iris\_data=load\_iris()  iris\_data # data, target, target\_names (인덱스명), future\_names  ---------------------------------------------------------------------------  import pandas as pd  import numpy as np  import seaborn as sns  df=sns.load\_dataset('titanic')  ----------  arr=np.arange(24)  # print(arr, type(arr), " : ", arr.shape, "\n", arr.size, ":", arr.dtype)  arr=arr.reshape(3,4,2)  # print(arr, type(arr), " : ", arr.shape, "\n", arr.size, ":", arr.dtype)  arr.sum(axis=2)  print(arr.dtype.name, arr.dtype.itemsize)  a=np.arange(-5, 5, 0.5)  a  ---------------------------------------------------------------------  a\_list=[1.0, 2.0, 3.0]  print(type(a\_list))  arr\_list=np.array(a\_list)  type(arr\_list)  -------------------------------------------------------------  x1 = np.array([1.0, 2.0, 3.0])  y1 = np.array([5.0, 10.0, 15.0])  x2 = np.array([[1.0, 2.0],[ 3.0, 4.0]])  y2 = np.array([[5.0,10.0],[15.0,20.0]])  z1 = np.array([-1.0, -2.0])  z2 = np.array([[5.0],[10.0],[15.0]])  print("# ndarray basic operation")  print(x1 + y1)  print(x1 - y1)  print(x1 \* y1)  print(x1 / y1)  print(x2 + y2)  print(x2 \* y2)  print()  print("# ndarray broadcast")  print(x2 + z1)  print(x2 \* z1)  print(x1 + z2)  print(x1\*\*2)  print(x1>=2)  print()  print("# shape manipulation")  print(x2.flatten())  print(x2.reshape(2,2))  -----------------------------------------------------------------  import numpy as np  x = np.array([1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20])  a=[]  b=[]  c=[]  for i in x:  if i%3==0:  a.append(i)  if i%4==1:  b.append(i)  if i%3==0 and i%4==1:  c.append(i)  print(a)  print(b)  print(c)  print("3의 배수 : ", x[x%3==0])  print("4로 나누면 1이 남는 배수 : ", x[x%4==1])  a\_arr=list(np.array([x%3==0]) & np.array([x%4==1]))  print("3의 배수이면서 4의 나머지 값이 1인 수 : ", x[a\_arr])  ----------------------------------------------------------------------------  import pandas as pd  import numpy as np  import seaborn as sns  titanic=sns.load\_dataset('titanic')  df=titanic.loc[ : , ['age','fare']]  df['ten']=10  df.head()  # 매핑 함수를 이용하여 각 원소에 동일한 함수 실행  def add\_10(n):  return n+10  add\_10(20)  --------------------  # apply() 매소드를 활용하여 시리즈의 각 원소에 동일한 함수 실행  ar=df['age'].apply(add\_10)  # df['ten\_10']=df['age'].apply(add\_10)  df['age\_lamb']=df['age'].apply(lambda x: add\_10(x))  df.head()  ---------------------------  df\_map=df.applymap(add\_10)  df\_map.head()  ---------------------------------------------------------------------------  # 1. titanic 데이터를 load  import seaborn as sns  titanic=sns.load\_dataset('titanic')  # 2. age와 fare 컬럼만 추출  df=titanic.loc[ : , ['age','fare']]  null\_cnt=df['age'].isnull().sum()  if null\_cnt:  df['age'].fillna(value=0, axis=0, inplace=True)  df['age'].isnull().sum(axis=0) # NaN 데이터를 처리, 평균나이로 대체  age\_avg=df['age'].sum()/df['age'].count()  def age\_minus(age, age\_avg):  return age-age\_avg  # 3. age에서 평균나이를 차감한 나이를 'age\_avg' 컬럼으로 추가  df['age\_m']=df['age'].apply(age\_minus, age\_avg=age\_avg)  # 4. 나이의 구간을 4단계로 나눠서 (0~20 : 청소년, 21~70 : 장년, 71~ 노년)으로 컬럼 추가  bin\_labels=['청소년','장년', '노년']  bin\_values=[0,21,71,200]  df['age\_bin']=pd.cut(x=df['age'], # 데이터 배열  bins=bin\_values, # 경계 리스트  labels=bin\_labels, # bin 이름  include\_lowest=True  )  # 5. one\_hot\_encoding  df[bin\_labels]=pd.get\_dummies(df['age\_bin'])  df[['age','age\_m']]=df[['age','age\_m']].astype("int")  df.head() |