[이론 문제]

Web Crawling에 대해 설명하시오

Web Crawling에 필요한 파이썬 라이브러리는 무엇인가?

파이썬 Web Crawling으로 데이터를 읽어오는 절차를 설명하시오

상관분석이 무엇인지 정의하시오

상관분석에서 상관계수에 대해 설명하시오

공분산을 구하는 공식을 쓰시오

상관계수의 공식을 쓰시오

상관계수값이 0.76이 나왔다 높은 상관관계라고 할 수 있는 가? 그 이유는?

상관분석과 회귀분석의 차이점을 설명하시오

회귀분석에 대해 설명하시오

파이썬에서 회귀분석을 수행하기 위해 필요한 라이브러리는 무엇인가

회귀분석 결과에서 Prob(F-statistic)에 대해 설명하시오

회귀분석 결과에서 개별 독립변수의 p값 - P>[t]에 대해 설명하시오

회귀분석 결과에서 회귀식(회귀모델)을 만들려고 한다 어떤 데이터를 참조해야 하는 가?

회귀모델이 얼마나 쓸만한지 평가하는 대표적인 지표 4가지를 쓰시오

회귀모델을 평가하는 지표 중 MAE는 무엇인가?

회귀모델을 평가하는 지표 중 R-squared에 대해 설명하고
아래 모델의 평가결과를 보고 모델이 쓸만한지 판단하여 설명하시오

R-squared: 0.7701353268580156

인공신경망에 대해서 설명하시오

다층퍼셉트론에 대해 설명하시오

딥러닝에 대해 설명하시오

딥러닝의 장점에 대해 설명하시오

파이썬에서 딥러닝을 위해 사용하는 라이브러리는 무엇인가 ?

다음 회귀분석의 결과를 보고 회귀식을 작성하고
주요한 지표값을 설명하시오 (시험에서는 값이 바뀜)

OLS Regression Results

Dep. Variable:	weight	R-squared:	0.770
Model:	OLS	Adj. R-squared:	0.762
Method:	Least Squares	F-statistic:	93.81
Date:	Sat, 16 Nov 2024	Prob (F-statistic):	1.94e-10
Time:	10:06:17	Log-Likelihood:	-78.279
No. Observations:	30	AIC:	160.6
Df Residuals:	28	BIC:	163.4
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	78.1551	6.169	12.669	0.000	65.519	90.791
food	4.6684	0.482	9.686	0.000	3.681	5.656

Omnibus:	1.785	Durbin-Watson:	2.054
Prob(Omnibus):	0.410	Jarque-Bera (JB):	1.239
Skew:	0.238	Prob(JB):	0.538
Kurtosis:	2.125	Cond. No.	128.

[결과]
회귀식 --> weight = 4.6684 x food + 78.1551

[코드 문제]
코드의 빈곳을 채우는 문제로 출제 됨. 다음 코드들에서만 출제
각 커맨드가 어떤 기능을 하는 지 아는 것이 중요

- 코드 1

```
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By
import time

query = input('검색할 키워드를 입력하세요: ')

url = 'https://www.naver.com/'
driver = webdriver.Chrome()
driver.get(url)
time.sleep(3)

search_box = driver.find_element(By.CSS_SELECTOR,"#query")
search_box.send_keys(query)
search_box.send_keys(Keys.RETURN)
time.sleep(2)

driver.find_element(By.CSS_SELECTOR, '#dsstd' ).click()
time.sleep(1)
driver.find_element(By.CSS_SELECTOR, '#dsstd2' ).click()
time.sleep(3)

news_titles = driver.find_elements(By.CSS_SELECTOR, ".news_tit")
print(news_titles)

for i in news_titles :
    title = i.text
    href = i.get_attribute('href')
    print(title , " : ", href)
```

- 코드 2

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

w = pd.read_csv('ch5-1.csv')
print(w, end="\n\n")
print(w.head(10), end="\n\n")

w_n = w.iloc[:,1:5]
print(w_n, end='\n\n')
w_cor = w_n.corr(method = 'pearson')
print(w_cor, end='\n\n')

sns.pairplot(w_n)

plt.figure(figsize = (10,7))
sns.heatmap(w_cor, annot = True, cmap = 'Blues')
plt.show()
```

- 코드 3

```
import pandas as pd
import matplotlib.pyplot as plt
import statsmodels.formula.api as smf

w = pd.read_csv('ch5-1.csv')
w_n = w.iloc[:,1:5]

model_lm = smf.ols(formula = 'weight ~ egg_weight', data = w_n)
result_lm = model_lm.fit()
result_lm.summary()

print(result_lm.summary())

plt.figure(figsize = (10,7))
plt.scatter(w.egg_weight, w.weight, alpha = .5)
plt.plot(w.egg_weight, w.egg_weight*2.3371 - 14.5475, color = 'red')
plt.text(66, 132, 'weight = 2.3371egg_weight - 14.5475', fontsize = 12)
plt.title('Scatter Plot')
plt.xlabel('egg_weight')
plt.ylabel('weight')
plt.show()



from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
import numpy as np

predicted_values = result_lm.predict()

mse = mean_squared_error(w_n['weight'], predicted_values)
mae = mean_absolute_error(w_n['weight'], predicted_values)
rmse = np.sqrt(mse)
r_squared = r2_score(w_n['weight'], predicted_values)

print("Mean Squared Error (MSE):", mse)
print("Mean Absolute Error (MAE):", mae)
print("Root Mean Squared Error (RMSE):", rmse)
print("R-squared:", r_squared)
```

- 코드 4

```
import pandas as pd

w = pd.read_csv("ch7-1.csv")
print(w.head(), end='\n\n')

from sklearn.model_selection import train_test_split

x_data = w.iloc[:,0:2].values
y_data = w.iloc[:,2].values

x_train, x_test, y_train, y_test = train_test_split(x_data, y_data, test_size=0.2)

from sklearn.neural_network import MLPRegressor
model_mlp = MLPRegressor().fit(x_train, y_train)

print(model_mlp.get_params(), end='\n\n')

y_pred_mlp = model_mlp.predict(x_test)

df_x_test = pd.DataFrame(x_test, columns=['egg_weight','acc_food'])
df_y_pred = pd.DataFrame(y_pred_mlp, columns=['predict'])
df_y_test = pd.DataFrame(y_test , columns=['real'])
df = pd.concat([df_x_test, df_y_test, df_y_pred], axis=1)
print(df, end='\n\n')



from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
R2 = r2_score(y_test, y_pred_mlp)
print("R2 = ", R2, end='\n\n')
```

- 코드 5

```
import tensorflow as tf
from tensorflow.keras.preprocessing import image
import numpy as np

model = tf.keras.models.load_model('xray_classification_model.h5')
print(model.summary(), end='\n\n')

image_path = r'person1_virus_11.jpeg'


img = image.load_img(image_path, target_size=(150, 150))
img_array = image.img_to_array(img)
img_array = np.expand_dims(img_array, axis=0)
img_array /= 255.0  # 이미지를 0-1 범위로 정규화

predictions = model.predict(img_array)

print(predictions, end='\n\n')
if predictions[0][0] > 0.5:
    print("이 이미지는 PNEUMONIA에 속합니다.")
else:
    print("이 이미지는 NORMAL에 속합니다.")
```

- 코드 6 : FastAPI

```
from fastapi import FastAPI
from fastapi import Form

app = FastAPI()

@app.get("/")
def read_root():
    return {"Hello": "World"}

@app.get("/item")
def read_item(item_id: int, name: str = None, age: int = 0):
    return {"item_id": item_id, "name": name, "age": age}

@app.post("/user")
def read_user_form(name: str = Form(...), studentcode: str = Form(...) , major: str = Form(...) ):
    return {"msg": f"{major} {name}님 ({studentcode}) 반갑습니다."}


@app.post("/plus")
async def plus_form(num1: int = Form(...), num2 : int = Form(...)):
    result = num1 + num2
    return {"msg": f"{num1} + {num2} = {result}"}


from fastapi import File, UploadFile
from fastapi.responses import FileResponse
import shutil
from pathlib import Path

@app.post("/uploadfile/")
async def create_upload_file(file: UploadFile = File(...)):
    save_path = Path("static/uploads") / file.filename
    save_path.parent.mkdir(parents=True, exist_ok=True)

    with save_path.open("wb") as buffer:
        shutil.copyfileobj(file.file, buffer)

    return {"filename": file.filename, "location": str(save_path)}
```

```
@app.get("/files/{filename}")
async def get_file(filename: str):
    file_path = Path("static/uploads") / filename
    if file_path.is_file():
        return FileResponse(path=file_path, filename=filename)
    else:
        raise HTTPException(status_code=404, detail="File not found")
```