Python programming and practice

Initial Medical Diagnosis Program

Progress Report : 1

Date : 2023-11-26
Name : 조예지
ID : 222162

1. Introduction

1) Background

When someone feels unwell, they often think about going to the hospital, but sometimes it can be confusing to decide which specific department to visit. If you end up hearing advice like going from dermatology to orthopedics, it means you have to allocate time to visit another department. To address this issue, I have come up with the concept of an initial medical diagnosis program. If a person enters their symptoms, the program will provide a rough prediction of the associated illness and suggest which type of hospital or department they should visit. I believe that such a program would be more practical for users, helping them make informed decisions when seeking medical care.

2) Project goal

The goal is to analyze a CSV file containing past patient symptoms and the corresponding diseases linked to those symptoms. The objective is to predict a patient's symptoms based on the analysis and recommend a suitable hospital for the identified symptoms.

3) Differences from existing programs

Existing programs primarily focus on predicting the disease based on a patient's symptoms. Consequently, patients need to revisit the information to determine which hospital to visit after receiving the prediction results. This program, however, differs by having a feature that informs users 'which hospital to visit' right from the beginning, aiming to streamline the process and reduce the need for additional searches compared to traditional programs.

2. Functional Requirement

1) Function 1 (user_input_validator.py) (Validation of input values)

- It validates the input values.

- If a correct value is entered, it returns that value; otherwise, it prints a message instructing to enter the correct format and prompts for input again.

(1) Detailed function 1 (validate_yes_no_input())

- Function that only accepts Yes or No.

- User input is stored in lowercase, so it is case-insensitive.

- If the user enters Yes or No, the respective value is returned; otherwise, it prints a message instructing to enter the correct format and prompts for input again.

(2) Detailed function 2 (validate_integer_input())

- Function that only accepts integers.

- It uses the isdigit() method to check if the entered value is a number, and if so, it converts the value to an integer and returns it.

- If the input value is not a number, it prints a message instructing to enter the correct format and prompts for input again.

(3) Detailed function 3 (validate_gender_input())

- Function that only accepts gender (Male or Female).

- User input is stored in lowercase, so it is case-insensitive.

- If the user enters Male or Female, the respective value is returned; otherwise, it prints a message instructing to enter the correct format and prompts for input again.

2) Function 2 (medical_diagnosis.py)

(Disease diagnosis & Hospital recommendation)

- It loads a CSV file containing patient data, identifies the patient with the most similar symptoms based on the information provided by the user, checks the disease of that patient, and outputs it. Then, it recommends an appropriate hospital.

class MedicalDiagnosis:

(1) Detailed function 1 (__init__())

- It takes the CSV file path, loads the data, and initializes the user input and the list of matched patients.

(2) Detailed function 2 (load_data())

- It reads data from the given CSV file and returns it in the form of a dictionary. The header and data are stored in a dictionary with keys 'header' and 'data', respectively.

(3) Detailed function 3 (calculate_similarity_score())

- This method calculates the similarity between the given patient and user input.

- For each item, the method increments the score by 1 if the user input and patient data values match.

- The final accumulated score is returned.

(4) Detailed function 4 (find_similar_disease())

- If there are similar patients, this method retrieves data for one of them and compares it with the user input to find the disease of a similar patient.

- It checks whether each item matches the user input and patient data values. If they do not match, it sets the disease for that item to "Unknown."

- After checking all items, if "Unknown" is set for any item, it is determined that there is no similar disease. Otherwise, it prints the disease of the similar patient.

(5) Detailed function 5 (recommend_hospital())

- Recommends a hospital based on the extracted disease.

3. Progress

1) Feature implementation

(1) user_input_validator.py

validate_yes_no_input():

Input: Receive a parameter called message. This is a message to output to the user.
Output: The user receives a "Yes" or "No" response.
Action: Through an infinite loop, the user will continue to receive input until they enter "yes" or "no" correctly. User input is verified by converting to lowercase letters, and if it is not correct, it outputs an error message and receives input again.

validate_integer_input():

Input: Receive a parameter called message. This is a message to output to the user.
Output: Enter an integer from the user.
Action: Through an infinite loop, the user continues to receive input until an integer is entered. Use the isdigit() method to verify that the input consists of integers, and if it is not an integer, it outputs an error message and receives the input again.

validate_gender_input():

Input: Receive a parameter called message. This is a message to output to the user.
Output: The user receives a response with "Male" or "Female".
Action: Through an infinite loop, the input is continuously received until the user enters "mal" or "female" correctly. User input is verified by converting to lowercase letters, and if it is not correct, it outputs an error message and receives input again.

- What you have learned applied (e.g. repetitions, conditional statements, functions, classes, modules, etc.)

Repeat statement, conditional statement, function, module

- Code Screenshot

PY202309-P > Sources > user_input_validator.py

```python
# Yes 또는 No로 응답을 입력받는 함수
def validate_yes_no_input(message):
    while True:
        user_input = input(f"{message}").lower()
        if user_input in ['yes', 'no']:
            return user_input
        else:
            print("올바른 형식으로 입력하세요. 'Yes' 또는 'No'로 입력하세요.")

# 정수 입력 받는 함수
def validate_integer_input(message):
    while True:
        user_input = input(f"{message}")
        if user_input.isdigit():
            return int(user_input)
        else:
            print("숫자 형식으로 입력하세요.")

# 성별 유효성 확인 함수
def validate_gender_input(message):
    while True:
        user_input = input(f"{message}").lower()
        if user_input in ['male', 'female']:
            return user_input
        else:
            print("올바른 형식으로 입력하세요. 'Male' 또는 'Female'로 입력하세요.")

```

2) Test Results

(1) main.py

- Verify that only valid values are received among the user's inputs
- Test Results Screenshot

```
올바른 형식으로 입력하세요. 'Yes' 또는 'No'로 입력하세요.
발열이 있나요? (Yes/No) :yes
기침이 있나요? (Yes/No) :no
피로감이 있나요? (Yes/No) :20
올바른 형식으로 입력하세요. 'Yes' 또는 'No'로 입력하세요.
피로감이 있나요? (Yes/No) :yes
호흡 곤란이 있나요? (Yes/No) :d
올바른 형식으로 입력하세요. 'Yes' 또는 'No'로 입력하세요.
호흡 곤란이 있나요? (Yes/No) :no
나이를 입력하세요. :20
성별을 입력하세요. (Male/Female) :dfk
올바른 형식으로 입력하세요. 'Male' 또는 'Female'로 입력하세요.
성별을 입력하세요. (Male/Female) :male
수축기 혈압을 입력하세요. :200
총 콜레스테롤 수치를 입력하세요. :k
숫자 형식으로 입력하세요.
총 콜레스테롤 수치를 입력하세요. :200
```

4. Changes in Comparison to the Plan

x

5. Schedule

- Progress Indication

업무		11/3	11/26	12/1	12/15
제안서 작성		---------->			
기능1 유효성 검사	세부기능1 세부기능2 세부기능3		---------->		
기능2	.....			---------->	