HAP 618 – Fall 2023

**Final Project paper**

“**Hospital Room Simulation**”

Sree Sarada Alapaty

**Background:**

Healthcare systems often face challenges in managing patient influx and optimizing care resources. The average wait time for patients in United States is more than an hour and half. Patients who arrive at EDs with broken bones wait a painful 54 min, on average before receiving any pain medications. As a matter of fact, the number of patients who walk through the hospital door is larger than the number of doctors available, the reason for increased wait times.

The Python-based Hospital Room Simulation project is designed to address current issues of patient care and allocation of resources in the ever-changing field of healthcare. An understanding of the intricate links among various medical conditions, patient arrivals, and the best utilization of healthcare resources form the backbone of the project. The goal of this simulation project is to close this gap by providing a safe environment in which researchers and healthcare practitioners can test various scenarios, evaluate how different variables affect outcomes, and improve approaches.

The simulation uses python for dynamic modeling and pandas for data management to create a flexible and realistic framework. It makes it easier to explore healthcare dynamics in an informed manner but also advances the continuous development of effective patient care practices and resource optimization in a simulated yet real-world scenario. The data I will be using is from Kaggle and it’s called Healthcare Dataset with patient demographics(random).

**Design of Solution:**

Python is the only programming language used in the project; Notepad++ is the editor and it executes on terminal or command prompt. The simulation relies on a dataset that includes name, age, gender, and medical condition – all crucial patient details.

The simulation model is primarily intended to explore the effects of patient’s arrivals and the average duration of care for each patient.

Python is the primary language used in this project for data processing, code development, and simulation modeling. Python efficiently manages patient data by utilizing the Pandas package, which includes important demographics.

Variable patient arrivals and care time are introduced using the random module which enhances realism. Python’s easy-to-read syntax makes it ideal for complex simulation logic. Since the code is meant to be run on a terminal or command prompt, it demonstrates the language’s adaptability. Overall, Python is a great option for interpreting the complexity of patient care dynamics in a simulated healthcare facility because of its strengths in processing data, simulation, and comprehensibility of the code.

**Implementation and Testing:**

The program that is being used is Python, it’s important, without that the evaluation of the patient care times can’t be done. Here is the code for the Hospital room simulation.

A screenshot of a computer

Description automatically generated

Line 1: Imports pandas as pd.

Line 2: Imports random module.

Line 3: Imports statistics module.

Line 5: Mentioning the simulationdataset.xlsx file and naming it excel\_file.

Line 6: Reading the simulationdataset.xlsx file using panda’s library.

Line 8: Giving the simulation hours i.e., for 12 hrs.

Line 9: Using a DataFrame ‘df’ to extract specific columns like ‘Name’, ‘Age’, ‘Gender’, ‘Medical

Condition’, and the data that we get is converted to dictionaries, where each dictionary

corresponds to patient’s information with keys ‘Name’, ‘Age’, ‘Gender’, ‘Medical

Condition’, and their respective values.

Line 12: Initializing the total care time spent on patient care.

Line 13: Initializing the patient queue to keep track of patients who are waiting in queue.

Line 14: Initializing the total number of patients to zero, as the simulation progresses new patients arrive

and count adds up.

Line 17: Using the randint function from the random module, representing the number of patients arriving in simulation. Set it between zero to five.

Line 18: Adding the no.of patients arriving in current hour is added to the total number of patients.

Line 19: Using the extend method to add patients from the patient\_data list to patients\_in\_queue.

Determined by patients arriving which is generated before.

Line 21: Shuffle the ‘patient\_data’ using the random module.

Line 24: Defining a function ‘simulate\_care’ with two parameters; ‘patient’ (patients demographics) and

‘total\_care\_time\_hour’ (care time for the current hour).

Line 25: Extract patients’ medical condition from the dictionary.

Line 28: Checks if the medical condition is none.

Line 29: If the medical condition is none or unspecified, it is assigned to care time between ten and thirty

minutes.

Line 30: Else condition is used if the medical condition is not ‘None’.

Line 31 to 38: If the medical condition is not ‘None’, then the ‘care\_time\_range’ assigns each medical

medical condition a range for care times. If the medical condition is not specified in the

dictionary it gives range between ten and thirty minutes.

Line 40: care time range is determined by the medical condition, and a random integer within that range

is assigned to ‘care\_time’. (\*care\_time\_range uses unpacking operator to pass elements of tuple

‘care\_time\_range’ as separate argument to randint function.

Line 42: Check if the care time is greater than zero.

Line 43: ‘If’ condition checks if the sum of total care time of the current hour and care time is exceeding sixty minutes.

A screenshot of a computer

Description automatically generated

Line 44: If condition above exceeds 60 minutes, then it adjusts care time to fit within the remaining time

by calculating the difference between 60 minutes and current total care time.

Line 46: It returns the determined care time.

Line 49: Initializing a loop to iterate each hour from hour 1 to ‘simulation\_hours’ as defined before.

Line 50: Initializing total care time for the current hour to zero. Keeping track of total care time for patients in current hour.

Line 53: Generates a random number between 0 and 5 to simulate number of patients arriving in

hour.

Line 54: Adds patients arriving in that hour to total number of patients through the simulation.

Line 56: Create a copy of the patient data and naming it as ‘patients\_for\_hour’.

Line 57: Randomly shuffle the list ‘patients\_for\_hour’. Create randomness in which patients will be

processed during the current hour.

Line 59: This extends the patients\_in\_queue with shuffled patients\_for\_hour list. This

subset includes patients\_arriving elements of shuffled list. Simulation adds specific no.of

patients to queue in the current hour.

Line 62: Initiates a loop that iterates ‘patients\_arriving’. (‘\_’) is used in the loop as a variable is not used.

Line 63: ‘if’ condition checks if there are any ‘patients\_in\_queue’, it is executed only if there are patients

in queue.

Line 64: Removes the repeated patient from the ‘patients\_in\_queue’ list using pop(0). Ensures patients are not repeated.

Line 65: ‘simulate\_care’ passes the ‘current\_patient’ and ‘total\_care\_time\_hour’ arguments. This

calculates the care\_time for patient’s based on their medical condition.

Line 67: Checks if the simulated care time is greater than zero. Means the patient needs care.

Line 68: if the condition is true, simulated care time is added to ‘total\_care\_time\_hour’.

Line 69: And ‘total\_care\_time’ is added to simulated ‘care\_time’. This is for calculating the total care time

for the entire simulation.

Line 71: Prints the information about care provided for patients. Includes the hour, patient’s name, and

Medical condition.

Line 72: Prints the total time of care provided to the patient during the current hour in minutes.

Line 75: Creates new list of ‘patients\_in\_queue’ by filtering patients’ medical condition as ‘None’ from

the queue.

Line 78: This ‘if’ condition checks if the ‘patients\_arriving’ in current hour is zero i.e., no patients arrived

or if the ‘total\_care\_time\_hour’ for current is zero.

Line 79: If the above condition is true, then it prints ‘No patients arrived’ during that hour.

Line 81: Calculates the average care time per patient. Condition ‘if’ total patients is greater than zero, it

divides ‘total\_care\_time’ by ‘total\_patients’ to get ‘average\_care\_time’.

Line 83: Prints the total number of patients.

Line 84: Prints the total care time overall in minutes.

Line 85: Prints average care time per patient. (:.2f) means (:) means format specification, (.2) means the

Number of digits after decimal point, and (f) means value is in float.

It is stored in ‘C:\Users\sarad\HAP618’\hsno.py.

The data set used for the simulation is also in C:\Users\sarad\HAP618\simulationdataset.xlsx.

Running the code in command prompt.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

This is the output generated after executing the code(hsno.py), gives a brief summary of the events of a healthcare facility during a period of twelve hours. There is patient’s information like their medical condition and amount of time spent providing care for them in that particular hour. Diabetes, asthama, obesity, hypertension, arthritis, and cancer are among the medical conditions that are used to categorize the patients.

An overview of the output states that 39 patients received treatment in total over the duration of the 12-hour period, total care time of 514 minutes. The average care time per patient is calculated to be 13.18 minutes. All these values for the variables vary when we run the simulation each time. In summary, the information provides an overview of the number of patients at the hospital, the array of medical conditions treated, and amount of time spent on each patient’s care.

A greater knowledge of patient demographics, common medical conditions, and the distribution of care time is made possible by this information. As such, this simulation provides insightful information for workload evaluation, allowing healthcare organization to allocate resources optimally, pinpoint areas for development, and raise overall patient care management efficiency.

**Conclusion & Future work:**

A carefully designed healthcare simulation is used in this python script to simulate patient arrivals and care duration over a twelve hour period. The script efficiently imports patient data from an Excel file by using the pandas library, obtaining crucial details like name, age, gender, and medical condition. Every hour, the simulation creates new patient lines based on a randomization of the number of patients arriving and the manner in which they are seen. Then simulates patient care according to their medical condition while maintaining realistic time limitations within predefined parameters. The script gaurantees a detailed picture of healthcare circumstances by carefully taking into account a variety of medical conditions and the time that go along with them.

Practical statistics are provided by the simulation’s output, such as the total number of patients, the total amount of care time, and the average care time per patient. The efficiency and task distribution of the simulated healthcare system are comprehensively outlined by these indicators.

Machine learning algorithms might be included into the script as it develops to anticipate patient arrivals, identify trends in medical conditions, and enhance treatment plans. In addition to showing a thorough grasp of the python programming, data manipulation, and simulation techniques, the script also shows a careful approach to modeling healthcare scenarios. This kind of modeling can be useful for evaluating how resources are used, streamlining workflows, and enhancing patient care plans in an evolving healthcare environment.

**References:**

1. <https://www.kaggle.com/datasets/prasad22/healthcare-dataset/data>, usability-10,

Domain: Public domain, simulationdataset.xlsx

2. Wait time in the hospitals: <https://www.cdc.gov/nchs/about/factsheets/factsheet_nhcs.htm>

3. http://stackoverflow.com/questions/

4. <https://www.tutorialspoint.com/python_pandas/python_pandas_dataframe.htm>

5. How to install python packages – packaging.python.org/en/latest/tutorials/installing-packages/

6.Reading a excel file – Dataquest.io/blog/reading-excel-file-python/

7. Class lecture notes in Python.