AUTOMATED HEALTH GRAPH GENERATOR

A project report submitted in partial fulfillment of the requirements

for the degree of

Bachelor of Technology

in

Electronics and Communication Engineering

By

Y.Madhu	(S170762)
K.Ramana	(S171061)
CH.Siva Keshavulu	(S170773)
K.Shivani	(S170231)
S.Indravathi	(S171025)
M.Ramya Sri	(S170194)
K.Sivani	(S171079)
K.Vijaya Durga	(S170168)

Under the Guidance Of

Dr H Srinivasa Vara Prasad, M.Tech, Ph.D.

Assistant Professor (Dept of ECE)

Department Of Electronics and Communication Engineering

Rajiv Gandhi University of Knowledge Technologies

SRIKAKULAM – 532410

APRIL – 2023

Certificate

This is to certify that the work entitled "**AUTOMATED HEALTH GRAPH GENERATOR**" titled submitted by Y.MADHU (S170762), K.RAMANA (S171061), CH.SIVA KESHAVULU (S170773), K.SHIVANI (S170231), S.INDRAVATHI (S171025), M.RAMYA SRI (S170194), K.SIVANI (S171079), K.VIJAYA DURGA (S170168) to the Department of ECE, Rajiv Gandhi University of Knowledge and Technologies ,Srikakulam for the submission of major project report in IV year B-Tech in ECE is a Bonafede work carried out under our supervision and guidance during the academic year 2022-23. The results embodied in this work have not been submitted to any other university or institute for the award of any degree or diploma. This thesis, in our opinion, is worthy of consideration for the credits of Major project for Semester – I & II, IV-year B-Tech in accordance with the regulations of the unit and award of the degree of Bachelor of Technology in accordance with the regulations of the institute.

Project Supervisor

Dr.H.Srinivasa Vara Prasad, M.Tech, Ph.D,
Asst.Professor(C),
Department of ECE,
RGUKT – Srikakulam.

Head of the Department,

Mr.N Ramesh Babu, M.Tech,
Asst.Professor(C),
Department of ECE,
RGUKT – Srikakulam.

ACKNOWLEDGEMENT

We would like to acknowledge and express our gratitude to the following people for their magnificent support in sharing their wisdom and knowledge to our successful completion of project. We are very thankful to Dr H Srinivasa VaraPrasad, M.Tech, Ph.D., Assistant professor, Project guide, Dept. Of ECE for the valuable inputs, motivation and persistence guidance throughout our project. We are very thankful to Mr.N.Ramesh Babu, M.Tech, Assistant professor, Head of the Department, Dept. Of ECE for his valuable inputs, motivation throughout our project. We are very thankful to all teaching and non-teaching staff, Dept. Of ECE for his valuable inputs, motivation throughout our project. We extend our gratitude for giving this opportunity to RGUKT-Srikakulam. Electronics and Communication Engineering IIIT – SRIKAKULAM.

Electronics and Communication Engineering
IIIT – SRIKAKULAM

Date:10/04/2023
Place: Srikakulam

Y.Madhu	(S170762)
K.Ramana	(S171061)
CH.Siva Keshavulu	(S170773)
K.Shivani	(S170231)
S.Indravathi	(S171025)
M.Ramya Sri	(S170194)
K.Sivani	(S171079)
K.Vijaya Durga	(S170168)

TABLE OF CONTENTS

List Of Figures

Full Forms

CSV	Comma-separated values
AST	Aspartate aminotransferase
ALT	Alanine transaminase
GGTP	Gamma-glutamyl transpeptidase
ALP	Alkaline phosphate
HDL	High-density lipoprotein
LDL	Low-density lipoprotein
VLDL	Very low-density lipoprotein
ESR	Erythrocyte sedimentation rate
ALC	Absolute leucocyte count
DLC	Differential leucocyte count
TLC	Total leucocyte count
RDW	Red cell distribution
MCV	Mean corpuscular volume
MCH	Mean corpuscular haemoglobin
MCHC	Mean corpuscular haemoglobin concentration
RBC	Red blood cells
PCV	Packed cell volume

ABSTRACT

It is very difficult to understand the medical reports for normal people who didn't study MBBS. And a specialist is needed to analyse the medical reports and tells about the patient health problem, condition and health status of the person, medicine, curation, remedy, solution for the problem, and if need of any changes in the food habits that the patient should follow. To derive meaningful insights from voluminous healthcare data, it is essential to convert it into machine understandable knowledge. Currently, machine understandable domain specific healthcare knowledge curation framework does not exist for complete body of human. By these automated systems helps extraction of medical reports data and the data is saved in the machine understandable format. The better way to convey the medical health reports is graphs or images. After the extraction of medical health data, graph is generated based on the extracted data. The automated graph helps in monitoring and recording all the vital information of a particular patient by maintaining all the records. Most prior work on information extraction has focused on extracting information from texts in digital documents. With this automated graph generator, patient data can be extracted, moved, integrated, and accessed seamlessly so that the patient is never inconvenienced.

CHAPTER 1:

INTRODUCTION

It is very difficult to understand the medical reports for normal people who didn't study MBBS. And a specialist is needed to analyse the medical reports and tells about the patient health problem, condition and health status of the person, medicine, curation, remedy, solution for the problem, and if need of any changes in the food habits that the patient should follow.

So, we need a analysing tool which analyse the human health reports such as Blood report, Lipid test, Diabetes test, Urinalysis, Kidney tests, Thyroid test etc. and tells about the Patient health problem, condition of the person, medicine for the problem and food habits for the person. In this we convert the health reports data into the graph format because the graphs are easy to understand by anyone. Automation has played a major part in the technological revamping of healthcare, whether it is integrating digital technology, advancing procedures.

Eventually, this results in the patients receiving better accuracy and an idea of one's health status from graph which is generated rather than the human-facing element of it. Thus, making procedures more productive and worth. And this tool helps in better understanding of the health status of a particular patient. Thus, the data of individual health data or the condition of each patient is stored in the database. Thus, this data can be accessed, stored, and retrieved whenever needed. In this way the health status is stored, accessed, and recorded.

By this tool even the normal people can know the status of one's own health condition. As we know that the graphical representation is universal, simple, language, everyone can understand it. While, this can take various forms, such as improving patient satisfaction, improving medical practices, and integrating technology by propagating the healthcare system further into the future. The use of modern technology can make and store a large amount of data, all while maintaining a high level of performance.

An automatic health graph generator is a system that utilizes wearable sensors, mobile applications, and machine learning algorithms to collect and process health data in real-time. The system is designed to generate personalized health status graphs that provide real-time monitoring and analysis of an individual's health status.

The use of automated health graph generators has the potential to improve patient outcomes by providing real-time monitoring and personalized management of health conditions. These systems have been tested and validated in various studies, showing promising results in accurately reflecting participants' health status. Overall, automatic health graph generators have the potential to revolutionize healthcare by providing personalized management of chronic diseases and improving overall health outcomes.

Health monitoring is an important aspect of modern healthcare, especially for chronic disease management. Traditionally, patients are required to visit healthcare facilities for periodic check-ups and consultation with healthcare professionals. However, this approach can be time-consuming and inconvenient for patients, and may not provide timely information on changes in health status. Wearable sensors have emerged as a promising solution to this problem by enabling continuous monitoring of physiological parameters such as heart rate, blood pressure, and temperature. These sensors provide real-time data that can be used to generate health status graphs, which are useful for both patients and healthcare professionals.

Health status graphs provide a visual representation of a patient's health status over a period of time, and can be used to identify trends and patterns that may indicate changes in health status. By analysing these graphs, healthcare professionals can make informed decisions about treatment plans, medication adjustments, and other interventions. Patients can also use these graphs to track their progress and make lifestyle changes to improve their health.

The automatic generation of health status graphs is an area of active research, and has the potential to improve patient outcomes by providing real-time monitoring and personalized management of health conditions. In this paper, we review the existing literature on automatic health graph generation using wearable sensors and machine learning algorithms.

Overall, health status graphs provide a valuable tool for both healthcare professionals and patients to monitor and manage health conditions. By providing real-time monitoring and personalized management, automatic generation of health status graphs using wearable sensors and machine learning algorithms has the potential to improve patient outcomes and revolutionize healthcare.

Health status graphs can serve as a motivational tool for patients to make positive changes to their lifestyle and track their progress. By regularly monitoring their health status and tracking changes over time, patients can see the impact of their lifestyle changes and medication adjustments on their health. This can help patients stay motivated and make adjustments to their health plan as needed to achieve their health goals. Additionally, health status graphs can help patients and their healthcare providers work together to identify patterns and trends, and make informed decisions about treatment plans and lifestyle changes. Overall, health status graphs can be a powerful tool for patient empowerment and improved health outcomes.

Health status graphs can help healthcare professionals to identify patterns and trends in a patient's physiological parameters over time, such as blood pressure and glucose levels. If a patient's blood pressure or glucose levels have been consistently high, healthcare professionals may recommend adjustments to their medication or lifestyle to manage their condition more effectively. For example, a healthcare professional may recommend dietary changes, exercise, or medication adjustments to help the patient manage their high blood pressure or diabetes. By monitoring a patient's health status using these graphs, healthcare professionals can make informed decisions about the best course of treatment for the patient.

To detect table data in an image, we can use the pytesseract library along with the cv2 library for image processing. Here is an example code that imports these libraries as well as other useful Python libraries:

```
import cv2
import pytesseract
import numpy as np
import pandas as pd
from PIL import Imag
```

Note that pytesseract requires the Tesseract OCR engine to be installed on your system. You can download the engine from the official website:

To read a CSV file in Python, you can use the csv module. Here's an example code snippet to read a CSV file:

```
import csv
# Open the CSV file
with open('example.csv', newline='') as csvfile:
    # Create a CSV reader object
    reader = csv.reader(csvfile, delimiter=',', quotechar='"')
    # Read each row in the CSV file
    for row in reader:
        # Do something with the row data
        print(row)
```

In this code, we first import the csv module. Then we open the CSV file using the open() function and pass the file name as a parameter. We also specify the newline parameter as an empty string to ensure that we are reading the file correctly regardless of the platform-specific line endings.

Next, we create a csv.reader object and pass the file object csvfile to it. We also specify the delimiter character (, in this example) and the quote character (" in this example) used in the CSV file.

Finally, we loop through each row in the CSV file using a for loop, and do something with the row data. In this example, we just print each row to the console. This code reads all CSV files in the directory specified by dir_path, concatenates them into a single data frame called merged_data, and then writes the merged data to a new CSV file. The index=False argument in the to csv method is used to exclude the index column from the output file.

Assigning new column names:

Assign new column names by setting the columns attribute of the DataFrame. For example:

```
import pandas as pd
# Create a DataFrame with some data
df = pd.DataFrame({'A': [1, 2, 3], 'B': [4, 5, 6]})
# Assign new column names
df.columns = ['X', 'Y']
# Display the updated DataFrame
print(df)
```

Sorting dataframe elements in the array form:

If you have a column containing arrays and you want to sort the elements within each array, you can use the apply method with a lambda function that sorts each array. For example:

```
import pandas as pd
# Create a DataFrame with a column of arrays
df = pd.DataFrame({'A': [[3, 1, 2], [6, 5, 4], [9, 8, 7]]})
# Sort the elements within each array
df['A'] = df['A'].apply(lambda x: sorted(x))
# Display the updated DataFrame
print(df)
```

Deleting the main test names and related values in the arrays:

If you have a column containing arrays and you want to remove certain values from each array, you can use the apply method with a lambda function that filters out those values. For example:

```
import pandas as pd
# Create a DataFrame with a column of arrays
df = pd.DataFrame({'A': [['test1', 10], ['test2', 20], ['test3', 30]]})
```

```
# Remove the first element from each array

df['A'] = df['A'].apply(lambda x: [v for v in x if v != 'test1'])

# Display the updated DataFrame

print(df)
```

Changing the test names if the test names are repeated:

If you have a column containing arrays and you want to change the values of certain elements, you can use the apply method with a lambda function that replaces those values. For example:

```
import pandas as pd

# Create a DataFrame with a column of arrays

df = pd.DataFrame({'A': [['test1', 10], ['test2', 20], ['test1', 30]]})

# Replace 'test1' with 'test4' in the first element of each array

df['A'] = df['A'].apply(lambda x: ['test4' if v == 'test1' and i == 0 else v for i, v in
enumerate(x)])

# Display the updated DataFrame

print(df)
```

Removing the units and other string data in normal ranges:

Assuming that you have a column in your DataFrame that contains strings with a normal range and a unit (e.g., "10-20 mg/dL"), you can remove the unit and other string data using regular expressions. Here's an example:

```
import pandas as pd

import re

# Create a DataFrame with a column of strings containing normal ranges and units

df = pd.DataFrame({'A': ['10-20 mg/dL', '5.2-6.8 mmol/L', '70-110 mg/dL']})

# Remove the unit and other string data using regular expressions

df['A'] = df['A'].apply(lambda x: re.sub(r'[^\d.-]+', '', x))

# Display the updated DataFrame

print(df)
```

Converting data into float type:

Assuming that you have a column in your DataFrame that contains strings representing numbers, you can convert them to float type using the astype method. Here's an example:

```
import pandas as pd
# Create a DataFrame with a column of strings representing numbers
df = pd.DataFrame({'A': ['10.5', '20.7', '30.2']})
# Convert the strings to float type
df['A'] = df['A'].astype(float)
# Display the updated DataFrame
print(df)
```

Separating lower limit and upper limit values:

Assuming that you have a column in your DataFrame that contains strings representing a normal range (e.g., "10-20"), you can separate the lower limit and upper limit values using the str.split method. Here's an example:

```
import pandas as pd
# Create a DataFrame with a column of strings representing normal ranges
df = pd.DataFrame({'A': ['10-20', '5.2-6.8', '70-110']})
# Split the strings into lower limit and upper limit values
df[['lower', 'upper']] = df['A'].str.split('-', expand=True)
# Convert the values to float type
df[['lower', 'upper']] = df[['lower', 'upper']].astype(float)
# Display the updated DataFrame
print(df)
```

Plotting points(1=Below normal,2=Normal , 3=Above normal):

Assuming that you have a column in your DataFrame that contains numerical values, you can create a new column with the corresponding category (below normal, normal, or above normal) using the apply method with a lambda function. Here's an example:

```
import pandas as pd

# Create a DataFrame with a column of numerical values

df = pd.DataFrame({'A': [15, 25, 35, 5, 10]})

# Create a new column with the corresponding category

df['category'] = df['A'].apply(lambda x: 1 if x < 10 else (2 if x <= 30 else 3))

# Display the updated DataFrame

print(df)
```

Creating the DataFrame only for test name and plotting points:

Assuming that you have a DataFrame with columns "test_name" and "result_value", you can create a new DataFrame with only the "test_name" column and a new column with the corresponding category (below normal, normal, or above normal) using the apply method with a lambda function. Here's an example:

```
import pandas as pd

# Create a DataFrame with columns "test_name" and "result_value"

df = pd.DataFrame({'test_name': ['test1', 'test2', 'test3', 'test4', 'test5'],

                   'result_value': [15, 25, 35, 5, 10]})

# Create a new DataFrame with only the "test_name" column and a new column with the
corresponding category

df_points = pd.DataFrame({'test_name': df['test_name'],

                          'category': df['result_value'].apply(lambda x: 1 if x < 10 else (2 if x <= 30
else 3))})

# Display the new DataFrame

print(df_points)
```

Reading the source file:Assuming that your source file is a CSV file, you can read it into a DataFrame using the read_csv function from pandas. Here's an example:

```
import pandas as pd

# Read the CSV file into a DataFrame

df = pd.read_csv('filename.csv')

# Display the DataFrame

print(df)
```

Plotting the graph for test names and result values:

Assuming that you have a DataFrame with columns "test_name" and "result_value", you can plot a graph using the plot method from pandas with the test names on the x-axis and the result values on the y-axis. Here's an example:

```
import pandas as pd
import matplotlib.pyplot as plt
# Create a DataFrame with columns "test_name" and "result_value"
df = pd.DataFrame({'test_name': ['test1', 'test2', 'test3', 'test4', 'test5'],
        'result_value': [15, 25, 35, 5, 10]})
# Plot a graph with the test names on the x-axis and the result values on the y-axis
df.plot(x='test_name', y='result_value', kind='bar')
# Show the graph
plt.show()
```

Plotting the graph for test names and conditions:

Assuming that you have a DataFrame with columns "test_name" and "condition", you can plot a graph using the barplot method from seaborn with the test names on the x-axis and the count of each condition on the y-axis. Here's an example:

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
# Create a DataFrame with columns "test_name" and "condition"
df = pd.DataFrame({'test_name': ['test1', 'test2', 'test3', 'test4', 'test5'],
        'condition': ['normal', 'below normal', 'above normal', 'normal', 'normal']})
# Plot a graph with the test names on the x-axis and the count of each condition on the y-axis
sns.countplot(x='test_name', hue='condition', data=df)
# Show the graph
plt.show()
```

- Optimized Efficiency:

  Efficiency is important in any medical setting as it helps to reduce the waiting time for patients and ensures that healthcare providers can see more patients in a day. One way to optimize efficiency is to use electronic medical records (EMRs) to store and manage patient data. This can help to reduce the time spent on administrative tasks such as filing and retrieving patient records. Additionally, implementing automated appointment reminders and online scheduling systems can also help to streamline the process and reduce wait times.

- Preliminary Identification of Diseases:

  Early detection of diseases is crucial for effective treatment and management. To improve preliminary identification of diseases, healthcare providers can use advanced screening tools and techniques such as genetic testing, imaging technologies, and biomarker analysis. Additionally, investing in research and development of new diagnostic tools can help to improve accuracy and reduce the time required for diagnosis.

- Accuracy in Diagnosis:

  Accurate diagnosis is essential for providing the most effective treatment and ensuring the best possible outcomes for patients. To improve accuracy in diagnosis, healthcare providers can use a combination of clinical expertise, advanced diagnostic tools, and evidence-based practice. Additionally, implementing quality assurance programs and regular training for healthcare providers can help to ensure consistent and accurate diagnosis across different settings.

- Cost-Effective Alternatives:

  Healthcare costs continue to rise, making it increasingly important to identify cost-effective alternatives for treatment and management of diseases. One approach to reducing costs is to focus on prevention and early intervention. This can involve investing in public health programs, promoting healthy lifestyles, and providing education and resources to patients. Additionally, using generic drugs and reducing unnecessary testing and procedures can help to reduce overall healthcare costs.

- The clarity in Presentations and Disclosures:

  Clear communication is essential for building trust and ensuring that patients and healthcare providers are on the same page. To improve clarity in presentations and disclosures, healthcare providers can use plain language and avoid technical jargon that patients may not understand. Additionally, providing clear instructions and written materials can help to ensure that patients understand their diagnosis and treatment options. Finally, healthcare providers can encourage patients to ask questions and provide ample time to answer them.

1.1 Purpose

An automatic health report generator is a tool designed to automatically generate a report on an individual's health status. The purpose of this tool is to make it easier and faster for healthcare professionals to generate a report on a patient's health status without having to spend a significant amount of time manually compiling and analysing data.

Automated health report generators use algorithms and machine learning techniques to analyse patient data such as medical history, test results, and vital signs. This information is then used to generate a comprehensive report that provides an overview of the patient's health status, identifies potential health risks, and suggests appropriate interventions.

The benefits of an automatic health report generator include improved efficiency and accuracy in healthcare. By automating the process of generating health reports, healthcare professionals can save time and focus on other important aspects of patient care. Additionally, automated health reports can provide a more objective and standardized assessment of a patient's health status, reducing the potential for human error and improving the accuracy of diagnoses and treatment recommendations.

Overall, the purpose of an automatic health report generator is to provide healthcare professionals with a powerful tool to help them deliver more personalized and effective care to their patients.

- To know the health condition, reason for the problem if any, food habits to be followed by the patient to not only the specialists but also the normal persons.
- The main purpose is to reduce the necessity of a doctor or a specialist to verify the test reports of patients and to identify the problem.
- To let the patient should aware of the condition of their own health.
- To record access and update and monitor the vital data/information of each patient.

An automatic health report generator can help healthcare professionals to quickly and efficiently gather and analyse patient data to generate personalized reports that can guide treatment decisions and interventions. By providing more accurate and objective assessments of a patient's health status, healthcare professionals can deliver more effective care that is tailored to the patient's individual needs and circumstances. Additionally, the use of an automatic health report generator can help to improve patient outcomes, reduce healthcare costs, and enhance the overall quality of care provided to patients.

1.2 Scope

An Automated heath graph generator could potentially have a wide scope, depending on the specific use case and goals.

- In future, this is going to be used in hospitals in wide range.
- These are used in storing, accessing, and monitoring vital information of individual data of each patient.
- The health graph generator would need to create visual representations of the data, such as line graphs, and scatter plots. These graphs could be customized to show different types of data over different time periods.
- Machine Learning: As machine learning technology continues to develop, a automatic health graph generator may e ale use this technology to analyse health data ad make predictions about the future health outcomes. Thus could e particularly identifying individuals who are at risk of developing certain health conditions.

The scope of an automatic health graph generator is to automatically generate visual representations of an individual's health data. This tool can take various forms, such as a mobile application or a web-based platform, and is designed to collect and process health data from various sources, including wearable devices, electronic health records (EHRs), and patient self-reports.

The generated health graphs may include information such as vital signs (e.g., heart rate, blood pressure, and temperature), fitness data (e.g., steps taken, calories burned, and distance walked), and medical test results (e.g., blood glucose levels, cholesterol levels, and lipid profiles). By visually representing this data in an easy-to-understand format, an automatic health graph generator can provide patients and healthcare professionals with valuable insights into the individual's health status and can facilitate communication between them.

The scope of an automatic health graph generator is therefore to improve patient engagement and self-management, enable more informed decision-making, and enhance the overall quality of care provided to patients. Additionally, an automatic health graph generator can help to identify trends or anomalies in health data, which may indicate potential health risks or the need for further medical evaluation.

1.3 Language used

This project will be based on purely some machine learning and data science techniques which are developed by using multiple python libraries such as pandas, matplotlib, mplcursors etc. And this project includes some machine learning techniques such as Data Extraction, Data Selection, Feature Extraction, Feature Selection. To develop this we have to use Visual Studio Code, Google Collab, Jupyter Notebook.

1.3.1 Data Extraction

Firstly, to perform the task we need some data. The test reports are need to be uploaded or need to convert the reports into machine understandable language. The data is stored in digital documents, so data to be extracted from those digital documents. So the extracted data is converted to machine understandable language and ready to perform operations on it.

1.3.2 Data Selection

After extraction, need to select the data which is essential to perform the operations from the extracted data. This is to select the data which is only essential for performing the main operations involved in generating the plot.

1.3.3 Feature Extraction

To perform any type of operations on raw data that's only possible when the data contain some features. After the selections of data need them with specified columns called Features.

1.3.4 Feature Selection

Even the data contains features, need to select certain features which are essential to generate the desired automated graph. Selecting the features for performing the operations are involved in this.

1.4 History of Python

Python is a widely-used, interpreted, object-oriented, and high-level programming language with dynamic semantics, used for general-purpose programming. It's everywhere, and people use numerous Python-powered devices on a daily basis, whether they realize it or not.

Python was created by Guido van Rossum, and first released on February 20, 1991. While you may know the python as a large snake, the name of the Python programming language comes from an old BBC television comedy sketch series called Monty Python's Flying Circus.

One of the amazing features of Python is the fact that it is actually one person's work. Usually, new programming languages are developed and published by large companies employing lots of professionals, and due to copyright rules, it is very hard to name any of the people involved in the project. Python is an exception. Of course, Guido van Rossum did not develop and evolve all the Python components himself. The speed with which Python has spread around the world is a result of the continuous work of thousands (very often anonymous) programmers, testers, users (many of them aren't IT specialists) and enthusiasts, but it must be said that the very first idea (the seed from which Python sprouted) came to one head – Guido's.

Python is maintained by the Python Software Foundation, a non-profit membership organization and a community devoted to developing, improving, expanding, and popularizing the Python language and its environment.

Python 2.0 was released on October 16, 2000, with many major new features, including a cycle-detecting garbage collector (in addition to reference counting) for memory management and support for Unicode. However, the most important change was to the development process itself, with a shift to a more transparent and community-backed process.

In February 1991, Van Rossum published the code (labeled version 0.9.0) to alt.sources. Already present at this stage in development were classes with inheritance, exception handling, functions, and the core datatypes of list, dict, str and so on. Also in this initial release was a module system borrowed from Modula-3; Van Rossum describes the module as "one of Python's major programming units". Python's exception model also resembles Modula-3's, with the addition of an else clause. In 1994 comp.lang.python, the primary discussion forum for Python, was formed, marking a milestone in the growth of Python's userbase.

1.5 Project Requirements

1.5.1 Software requirements

- **Python**

As the python software is needed, few python libraries are used for the project. Those are:

1. Pandas
2. Matplotlib
3. Glob
4. Mplcursors

1.5.1.1 Pandas

Pandas is a Python package providing fast, flexible, and expressive data structures designed to make working with “relational” or “labelled” data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, real-world data analysis in Python. Additionally, it has the broader goal of becoming the most powerful and flexible open-source data analysis/manipulation tool available in any language. It is already well on its way toward this goal.

pandas is well suited for many different kinds of data:

1. Tabular data with heterogeneously-typed columns, as in an SQL table or Excel spreadsheet.
2. Ordered and unordered (not necessarily fixed-frequency) time series data.
3. Arbitrary matrix data (homogeneously typed or heterogeneous) with row and column labels.
4. Any other form of observational / statistical data sets. The data need not be labelled at all to be placed into a pandas data structure.
5. The two primary data structures of pandas, Series (1-dimensional) and Data Frame (2-dimensional), handle the vast majority of typical use cases in finance, statistics, social science, and many areas of engineering.
6. For R users, Data Frame provides everything that R’s data frame provides and much more.
7. Pandas is built on top of NumPy and is intended to integrate well within a scientific computing environment with many other 3rd party libraries.

Here are just a few of the things that pandas do well:

1. Easy handling of missing data (represented as NaN) in floating point as well as non-floating point data.

2. Size mutability: columns can be inserted and deleted from Data Frame and higher dimensional objects.

3. Automatic and explicit data alignment: objects can be explicitly aligned to a set of labels, or the user can simply ignore the labels and let Series, Data Frame, etc. automatically align the data for you in computations.

4. Powerful, flexible group by functionality to perform split-apply-combine operations on data sets, for both aggregating and transforming data.

5. Make it easy to convert ragged, differently-indexed data in other Python and NumPy data structures into Data Frame objects.

6. Intelligent label-based slicing, fancy indexing, and sub setting of large data sets.

7. Intuitive merging and joining data sets.

8. Flexible reshaping and pivoting of data sets.

9. Hierarchical labelling of axes (possible to have multiple labels per tick).

10. Robust IO tools for loading data from flat files (CSV and delimited), Excel files, databases, and saving / loading data from the ultrafast HDF5 format.

11. Time series-specific functionality: date range generation and frequency conversion, moving window statistics, date shifting, and lagging.

1.5.1.2 Matplotlib

Matplotlib is a cross-platform, data visualization and graphical plotting library for Python and its numerical extension NumPy. As such, it offers a viable open-source alternative to MATLAB.

Developers can also use matplotlib's APIs (Application Programming Interfaces) to embed plots in GUI applications.

A Python matplotlib script is structured so that a few lines of code are all that is required in most instances to generate a visual data plot. The matplotlib scripting layer overlays two APIs:

1. The pyplot API is a hierarchy of Python code objects topped by matplotlib.pyplot

2. An OO (Object-Oriented) API collection of objects that can be assembled with greater flexibility than pyplot. This API provides direct access to Matplotlib's backend layers.

The pyplot API has a convenient MATLAB-style stateful interface. In fact, matplotlib was originally written as an open source alternative for MATLAB. The OO API and its interface is more customizable and powerful than pyplot, but considered more difficult to use. As a result, the pyplot interface is more commonly used, and is referred to by default in this article.

Understanding matplotlib's pyplot API is key to understanding how to work with plots:

1. matplotlib.pyplot.figure: Figure is the top-level container. It includes everything visualized in a plot including one or more Axes.

2. matplotlib.pyplot.axes: Axes contain most of the elements in a plot: Axis, Tick, Line2D, Text, etc., and sets the coordinates. It is the area in which data is plotted. Axes include the X-Axis, Y-Axis, and possibly a Z-Axis, as well.

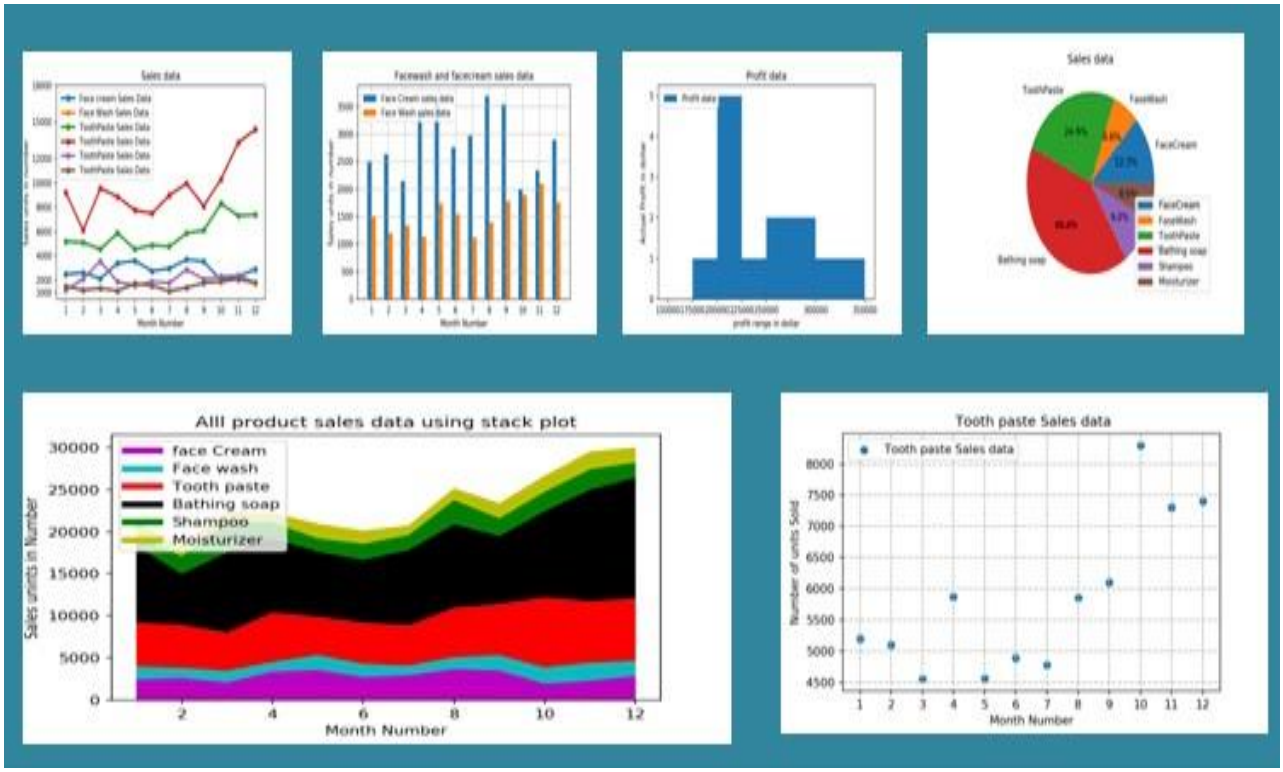


Figure-1.1: Types of plotting

Figure 1.1 : The above figure shows the different types of plots or graph that we ca generate by using matplotlib library in python. These includes scatter plot, stack plot, bar plots, pie charts etc. There are six different plots mentioned in the figure. By using different methods, we can generate different plots in python. This is all when matplotlib is imported only.

1.5.1.3 Glob

The glob module is a useful part of the Python standard library. glob (short for global) is used to return all file paths that match a specific pattern.

We can use glob to search for a specific file pattern, or perhaps more usefully, search for files where the filename matches a certain pattern by using wildcard characters.

These patterns are similar to regular expressions but much simpler.

1. Asterisk (*): Matches zero or more characters.

2. Question Mark (?) Matches exactly one character.

While glob can be used to search for a file with a specific filename, I find it especially handy for reading in several files with similar names. After identifying these files, they can then be concatenated into one data frame for further analysis.

Here we reviewed Python's glob module and two use cases for this powerful asset to Python's standard library.

We demonstrated the use of glob to find all files in a directory that match a given pattern. The files were then concatenated into a single data frame to be used for further analysis.

We also discussed the use of iglob to recursively search many directories for files containing a given string.

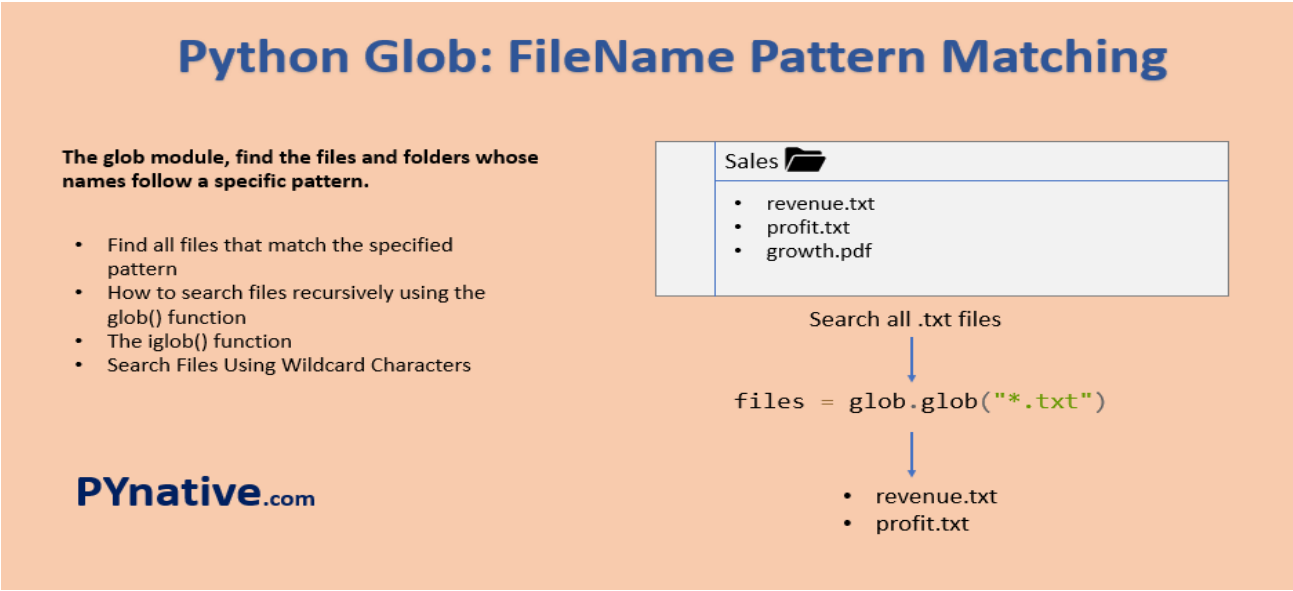


Figure-1.2: Working of glob function

Figure 1.2 : The above figure shows how a glob function works and how to use it.

1.5.1.4 mplcursors

Instead of providing a host of keyword arguments in Cursor's constructor, mplcursors represents selections as Selection objects and lets you hook into their addition and removal.

Specifically, a Selection has the following fields:

1. artist: the selected artist,
2. target: the (x, y) coordinates of the point picked within the artist.
3. index: an index of the selected point, within the artist data, as detailed below.
4. dist: the distance from the point clicked to the target (mostly used to decide which artist to select).
5. annotation: a Matplotlib Annotation object.
6. extras: an additional list of artists, that will be removed whenever the main annotation is deselected.

The exact meaning of index depends on the selected artist:

1. For Line2Ds, the integer part of index is the index of segment where the selection is, and its fractional part indicates where the selection is within that segment.
2. For step plots (i.e., created by plt.step or plt.plot, we return a special Index object, with attributes int (the segment index), x (how far the point has advanced in the x direction) and y (how far the point has advanced in the y direction). See Step plots for an example.
3. On polar plots, lines can be either drawn with a "straight" connection between two points (in screen space), or "curved" (i.e., using linear interpolation in data space).
4. In the first case, the fractional part of the index is defined as for cartesian plots.
5. In the second case, the index in computed first on the interpolated path, then divided by the interpolation factor (i.e., pretending that each interpolated segment advances the same index by the same amount).
6. For AxesImages, index are the (y, x) indices of the selected point, such that data[y, x] is the value at that point (note that the indices are thus in reverse order compared to the (x, y) target coordinates!).
7. For Containers, index is the index of the selected sub-artist.
8. For LineCollections and PathCollections, index is a pair: the index of the selected line, and the index within the line, as defined above.
9. As mplcursors **is** fundamentally a library for interactivity.

When mplcursors tries to blit an animation on top of the image, the animated artists will not be drawn, and disappear. More importantly, it also means that once an annotation is added, mplcursors cannot remove it.

Some complex plots, such as contour plots, may be partially supported, or not at all. Typically, it is because they do not subclass Artist, and thus appear to cursor as a collection of independent artists (each contour level, in the case of contour plots).

As a workaround, either switch off blitting, or unset the animated property on the relevant artists before using a cursor. (The only other fix I can envision is to walk the entire tree of artists, record their visibility status, and try to later restore them; but this would fail for Artist Animations which themselves fiddle with artist visibility).

Some complex plots, such as contour plots, may be partially supported, or not at all. Typically, it is because they do not subclass Artist, and thus appear to cursor as a collection of independent artists (each contour level, in the case of contour plots).

It is usually possible, again, to hook the "add" signal to provide additional information in the annotation text. See Contour plots for an example.

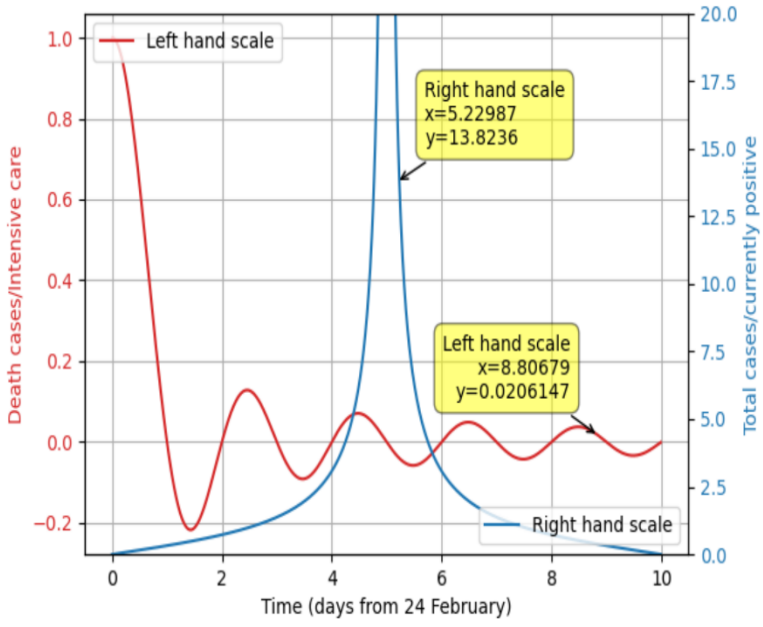


Figure-1.3: Text display at cursor position

Figure 1.3 : The above figure shows that when the cursor is moved onto the plotted lie the it shows some information regarding to that particular position. This is only possible when we imported mplcursors into the python code that is going to the performed and the particular information stored I am csv format file.

CHAPTER 2:

Literature Review

- Jiayi Yuan et al.: “An OpenCV-based Framework for Table Information Extraction”, in this paper, they propose a novel OpenCV-based framework to extract the metadata and specific values from PDF tables. Specifically, they first highlight the visual outline of the tables. Then, they located tables using horizontal and vertical lines and get the coordinates of tabular frames in each PDF page. Once the tables are successfully detected, for each table, they detected the cross-page scenarios and use the Optical Character Recognition (OCR) engine to extract the specific values in each table cell. Differing from other machine learning based methods, they proposed method can achieve table information extraction accurately without labeled data. They conduct extensive experiments on real-world PDF files.

- Shubham Paliwal, et al.: “TableNet: Deep Learning model for end-to-end Table detection and Tabular data extraction from Scanned Document Images”, In this paper, they propose TableNet: a novel end-to end deep learning model for both table detection and structure recognition. The model exploits the interdependence between the twin tasks of table detection and table structure recognition to segment out the table and column regions. This is followed by semantic rule-based row extraction from the identified tabular sub-regions. The proposed model and extraction approach was evaluated on the publicly available ICDAR 2013 and Marmot Table datasets obtaining state of the art results. Additionally, they demonstrate that feeding additional semantic features further improves model performance and that the model exhibits transfer learning across datasets.

- Mohammad Minouei et al.: “Continual Learning for Table Detection in Document Images”, This paper explores this issue in the table detection problem. While there are multiple datasets and sophisticated methods for table detection, the utilization of continual learning techniques in this domain has not been studied. We employed an effective technique called experience replay and performed extensive experiments on several datasets to investigate the effects of catastrophic forgetting. The results show that our proposed approach mitigates the performance drop by 15 percent.

- Andrei Puha et al.: “Enhancing Open Data Knowledge by Extracting Tabular Data from Text Images” In this paper they present an algorithm which enhances nowadays knowledge by extracting tabular data from scanned pdf documents in an efficient way. The proposed workflow consists of several distinct steps: first the pdf documents are converted into images, subsequently images are preprocessed using specific processing techniques. The final steps imply running an adaptive binarization of the images, recognizing the structure of the tables, applying Optical Character Recognition (OCR) on each cell of the detected tables and exporting them as csv. After testing the proposed method on several low-quality scanned pdf documents, it turned out that our methodology performs alike dedicated OCR paid software and they have integrated this algorithm as a service in our platform that converts open data in Linked Open Data.

- Sebastian Schreiber et al.: “DeepDeSRT: Deep Learning for Detection and Structure Recognition of Tables in Document Images”, This paper presents a novel end-to-end system for table understanding in document images called DeepDeSRT. In particular, the contribution of DeepDeSRT is two-fold. First, it presents a deep learning-based solution for table detection in document images. Secondly, it proposes a novel deep learning-based approach for table structure recognition, i.e., identifying rows, columns, and cell positions in the detected tables. In contrast to existing rule-based methods, which rely on heuristics or additional PDF metadata (like, for example, print instructions, character bounding boxes, or line segments), the presented system is data-driven and does not need any heuristics or metadata to detect as well as to recognize tabular structures in document images. Furthermore, in contrast to most existing table detection and structure recognition methods, which are applicable only to PDFs, DeepDeSRT processes document images, which makes it equally suitable for born-digital PDFs (as they can automatically be converted into images) as well as even harder problems, e.g., scanned documents.

- T. Kasar et al.: “Learning to Detect Tables in Scanned Document Images using Line Information”, in this paper presents a method to detect table regions in document images by identifying the column and row line-separators and their properties. The method employs a run-length approach to identify the horizontal and vertical lines present in the input image. From each group of intersecting horizontal and vertical lines, a set of 26 low-level features are extracted and an SVM classifier is used to test if it belongs to a table or not. The performance of the method is evaluated on a heterogeneous corpus of French, English and Arabic documents that contain various types of table structures and compared with that of the Tesseract OCR system.

- Rohit Sahoo et al.: “Auto-Table-Extract: A System To Identify And Extract Tables From PDF To Excel”, In this paper, they have proposed a Machine Learning based system called Auto-Table-Extract. This tool identifies and extracts the tables from PDF documents and dumps the data into excel sheets. It works with all kinds of PDF containing bordered, borderless, or partially bordered tables. This system can extract data from both searchable and scanned PDF. The system’s performance is commensurate to other table detection and extraction methods, but it overcomes limitations of both detecting borderless as well as partially bordered tables and proves to be an efficient solution for the detection of tables from diverse documents.

- Wenbin Sun et al.: “A Review of Recent Advances in Vital Signals Monitoring of Sports and Health via Flexible Wearable Sensors”, To this end, the paper reviews recent advances in flexible wearable sensors for monitoring vital signals in sports and health. More precisely, emerging wearable devices and systems for health and exercise-related vital signals (e.g., ECG, EEG, EMG, inertia, body movements, heart rate, blood, sweat, and interstitial fluid) are reviewed first. Then, the paper creatively presents multidimensional and multimodal wearable sensors and systems. The paper also summarizes the current challenges and limitations and future directions of wearable sensors for vital typical signal detection. Through the review, the paper finds that these signals can be effectively monitored and used for health management (e.g., disease prediction) thanks to advanced manufacturing, flexible electronics, IoT, and artificial intelligence algorithms; however, wearable sensors and systems with multidimensional and multimodal are more compliant.

- WEIJIA LU ,“A Clinical Prediction Model in Health Time Series Data Based on Long Short-Term Memory Network Optimized by Fruit Fly Optimization Algorithm”, Aiming the problems that the clinical data of different patients is difficult for reasonable representation and the time interval between medical events is different, which lead to the difficulty of clinical prediction, a clinical prediction model based on the long short-term memory (LSTM) network optimized by fruit fly optimization algorithm in health time series data is proposed. First, FastText method is used to represent the interpretable vector of medical events, which can extract the concept relationship rich in medical information more effectively. Then, considering the strong dependence of clinical data on time stamp, LSTM network is used to model clinical events for better extraction of long-term and short-term information, so as to improve the prediction performance of the model. Finally, the fruit fly optimization algorithm is used to find the optimal super parameters of LSTM network, which can improve the training efficiency and prediction precision of the network.

- Shubhangu Shukla et al.: “Health Care Management System Using Time Series Analysis”, The paper proposes a technique for risk prediction with graph-based proofs so as to observe and monitor the progress of a developing situation in the field of health. Making an automated system which will help to predict various health parameters of a host remotely is our primary objective. Providing a way to predict the future body temperature, pulse rate, hemoglobin, systolic and diastolic pressure with the help of an existing data set record, in an optimized manner is our primary concern using which we can forecast the health of a patient. Forecasting the health condition is being done by using Time series algorithm to predict body temperature, pulse rate, hemoglobin, systolic and diastolic pressure of the host patient, in an efficient manner.

- “ IOT Based Real-Time Remote Patient Monitoring System”, This paper proposes an Internet of Things (IoT) based real-time remote patient monitoring system that is able to guarantee the integrity of the real-time electrocardiogram (ECG). Message Queuing Telemetry Transport (MQTT) protocol is used for transmitting the real-time ECG from the proposed system to the webserver. The doctor can access the webserver via smartphone or computer to monitor the real-time or previously recorded ECG data. The proposed system has been tested in both Local Area Network and Wide Area Network environments.

- Zach Colter et al.: “ Tablext: A combined neural network and heuristic based table extractor”, order to address these issues, a novel, open-source, general format table extractor tool, Tablext, is proposed. This tool uses a combination of computer vision techniques and machine learning methods to efficiently and effectively identify and extract data from tables. Tablext begins by using a custom Convolutional Neural Network (CNN) to identify and separate all potential tables. The identification process is optimized by combining the custom CNN with the YOLO object detection network. Then, the high-level structure of each table is identified with computer vision methods. This highlevel, structural meta-data is used by another CNN to identify exact cell locations. As a final step, Optical Characters Recognition (OCR) is performed on every individual cell to extract their content without needing machine-readable text. This multi-stage algorithm allows for the neural networks to focus on completing complex tasks, while letting image processing methods efficiently complete the simpler ones.

CHAPTER 3:

Report based General Plot

Introduction:

It is very difficult to understand the medical reports. Every time we need a doctor to analyse the medical reports. It takes lot of time, so we need to convey the information in easy way which is understandable by all. Graphs are easy to understand for everyone.

So, we need a analysing tool which analyse the human health reports such as Blood report, Lipid test, Diabetes test, Urinalysis, Kidney tests, Thyroid test etc and generate a graph regarding the health report. Automation has played a major part in the technological revamping of healthcare, whether it is integrating digital technology, advancing procedures.

The test reports are the converted to csv files, so that the machine can read and perform operations on the files using the padas library. For accessing the information from digital documents, it is essential to convert it into machine understandable language. By converting the test reports into CSV files, you can use pandas to create data frames, which are a useful data structure for organizing and manipulating large sets of data.

Objective:

- Converting a single test report into a general plot.

Procedure:

- Import python libraries padas and matplotlib
- Scan the health report and upload report in online jpg to csv convertor in the form of images. The online jpg to csv convertor detects the text data in the reports and saves csv format and graph will be generated based on the extracted data.
- Remove the unwanted data in the Data Frame like units and other data.
- Rename the column names for the Data Frame to performs operations using the column names
- Remove the nan values, by using the indexes in the data frame by converting the data frame into arrays.
- If test names are repeated then rename the test names by adding the indexes at the end of the test names.
- The data frame may contain the numeric data but in the string format, so convert the results into float values.
- Create a data new data frame with Test Names and Results array.
- Plot the graph with the help of Test Names and Results

Health Data Reports:

Test Name	Results	Units	Bio. Ref. Interval
HEMOGRAM (Flow Cytometery, SLS,Capillary Photometry)			
Hemoglobin	**11.00**	g/dL	11.50 - 15.00
Packed Cell Volume (PCV)	40.00	%	36.00 - 46.00
RBC Count	3.90	mill/mm3	3.80 - 4.80
MCV	80.00	fL	80.00 - 100.00
MCH	30.00	pg	27.00 - 32.00
MCHC	33.00	g/dL	32.00 - 35.00
Red Cell Distribution Width (RDW)	**11.00**	%	11.50 - 14.50
Total Leukocyte Count (TLC)	8.90	thou/mm3	4.00 - 10.00
Differential Leucocyte Count (DLC)			
Segmented Neutrophils	69.50	%	40.00 - 80.00
Lymphocytes	20.50	%	20.00 - 40.00
Monocytes	8.10	%	2.00 - 10.00
Eosinophils	1.80	%	1.00 - 6.00
Basophils	0.10	%	<2.00
Absolute Leucocyte Count			
Neutrophils	6.19	thou/mm3	2.00 - 7.00
Lymphocytes	1.82	thou/mm3	1.00 - 3.00
Monocytes	0.72	thou/mm3	0.20 - 1.00
Eosinophils	0.16	thou/mm3	0.02 - 0.50
Basophils	0.01	thou/mm3	0.01 - 0.10
Platelet Count	190.0	thou/mm3	150.00 - 450.00
ESR	10	mm/hr	0 - 20

Figure 3.1.1: Hemogram Test Report

Figure 3.1.1 : The above figure about the Hemogram test report. This test is necessary in diagnosing anemia, hematological cancers, infections, acute hemorrhagic states, allergies, and immunodeficiencies. This tests are also used to monitoring side effects of certain drugs.

IMG_20230205_01... • • •

	A	B	C	D	E
1	Test Name	Results	Units	Bio. Ref. Interval	
2	HEMOGRAM (Flow Cytometery, SLS, Capillary Photometry)				
3	Hemoglobin	11	g/dL	11.50 15.00	
4	Packed Cell Volume (PCV)	40	%	36.00 46.00	
5	RBC Count	3.9	mill/mm3	3.80 4.80	
6	MCV	80	fL	80.00 100.00	
7	MCH	30	pg	27.00 32.00	
8	MCHC	33	g/dL	32.00 35.00	
9	Red Cell Distribution Width (RDW)	11	%	11.50 14.50	
10	Total Leukocyte Count (TLC)	8.9	thou/mm3	4.00 10.00	
11	Differential Leucocyte Count (DLC)				
12	Segmented Neutrophils	69.5	%	40.00 80.00	
13	Lymphocytes	20.5	%	20.00 40.00	
14	Monocytes	8.1	%	2.00 10.00	
15	Eosinophils	1.8	%	1.00 6.00	
16	Basophils	0.1	%	<2.00	
17	Absolute Leucocyte Count				
18	Neutrophils	6.19	thou/mm3	2.00 7.00	
19	Lymphocytes	1.82	thou/mm3	1.00 3.00	
20	Monocytes	0.72	thou/mm3	0.20 1.00	
21	Eosinophils	0.16	thou/mm3	0.02 0.50	
22	Basophils	0.01	thou/mm3	0.01 0.10	
23	Platelet Count	190	thou/mm3	150.00 450.00	
24	ESR	10	mm/hr	0 20	

Figure 3.1.2: Hemogram Test Report

Figure 3.1.2: The above figure about the Hemogram test report. This test is necessary in diagnosing anemia, hematological cancers, infections, acute hemorrhagic states, allergies, and immunodeficiencies. This tests are also used to monitoring side effects of certain drugs.

Code:

```
import pandas as pd
import numpy as np
import matplotlib
import matplotlib.pyplot as plt
merge_data=pd.read_csv("IMG_20230205_013735.jpg1.csv")
del merge_data["Units"]
column_names=list(merge_data.columns)
```

```
# assigning new column names
new_column_names=[]
for k in range(len(merge_data.columns)):
    if(k==0):
        new_column_names.append("Investigation")
    elif(k==1):
        new_column_names.append("Result")
    elif(k==2):
        new_column_names.append("Normal Ranges")
    else:
        new_column_names.append(column_names[k])
merge_data.columns=new_column_names
# for storing the dataframe elements in the array form
results=[]
testname=[]
indexes=[]
Interval=[]
Results=merge_data['Result']
Testname=merge_data['Investigation']
interval=merge_data['Normal Ranges']
for j in range(len(Results)):
    results.append(Results[j])
for k in range(len(Testname)):
    testname.append(Testname[k])
for l in range(len(interval)):
    if type(interval[l])==float:
        indexes.append(l)
# for deleting the main test names and its related values in the arrays
for i in range(len(indexes)):
        results.pop(indexes[i]-i)
         testname.pop(indexes[i]-i)
# for changing the test names if the test names are repeated
test_names=[]
```

```
for j in range(len(testname)):
    for k in range(len(testname)):
        if(j!=k):
            if(testname[j]==testname[k]):
                testname[k]=testname[k].replace(testname[k],testname[k]+str(k))
        else:
            test_names.append(testname[k])
# for removing the : in test names
test_name=[]
for i in test_names:
    i=i.replace(':',' ')
    i=i.strip()
    test_name.append(i)
result=[]
for i in results:
    if(type(i)==str):
        i=i.replace(',','.')
    result.append(float(i))
# for creating the data frame only for testname and plotting points
data={'Test Name':test_name,'Result':results}
df=pd.DataFrame(data)
# for plotting the graph for test names and result values
plot1=plt.plot(df.loc[:,"Test Name"],result,marker="*")
plt.xticks(rotation=90)
plt.title("Graph for Test Names and Resultant Values")
plt.xlabel("Test Names")
plt.ylabel("Result Values")
plt.show()
```

RESULTS:

The output is a simple graph that generated on basis of various test names of medical health reports vs results of the reports as a numerical value respectively. The data is extracted from digital documents and then with test names and results as features and various tests and numerical values of results as datapoints the graph is automatically plotted. It sounds like the generator is designed to extract test names and results from digital medical reports, and then use this information to automatically generate a graph with the test names on the x-axis and the numerical results on the y-axis. The graph can then be used to visualize the data and identify trends or patterns over time.

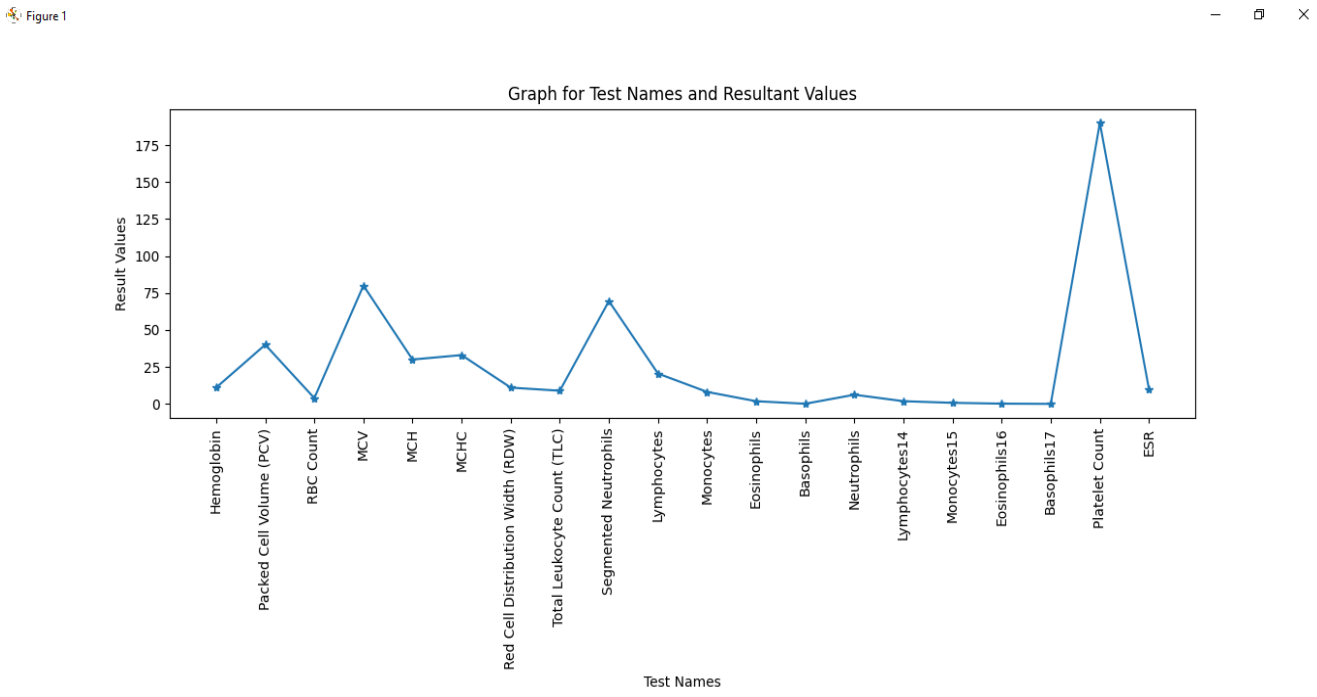


Figure 3.2: Report based general plot

Figure 3.2 : The above figure about the general plot for the health report of Hemogram. The graph is plotted based on the Test names on X- Axis and Result values on Y-Axis. The graph contains few similar Test Names, So the Test Names renamed by adding index value at the end of the Test Name.

CHAPTER 4:

General Plot of Different Reports

Introduction:

If the person feels that he is ill, then he/she will consults the doctor. Based on the symptoms told by the person the doctor will recommend the tests . The tests may be more than one based on the symptoms and health condition of the person. It is difficult understand all the health reports. The test reports are the converted to csv files, so that the machine can read and perform operations on the files using the padas library. We have to generate a single and continious graph for all the test reports.

Objective:

- Plotting Multiple reports as graphs in a single continuous graph.

Procedure:

1. Firstly, the reports are converted in csv format. And then storing all the files in a certain folder for reading similar files.
2. Thus, by merging all the test reports and having test names and results respectively.
3. After merging all the files, it creates a new Data Frame. We have to reset the index to perform operations on data frames.
4. We have to follow same process as done in the report based general plot to remove unwanted data, to rename the column names, to remove the nan values, renaming the test names if test names are repeated in the Data Frame.
5. The data frame may contain the numeric data but in the string format, so convert the results into float values.
6. Create a data new data frame with Test Names and Results array.
7. Plot the graph with the help of Test Names and Results

Health Data Reports:

Test Name	Results	Units	Bio. Ref. Interval
correlates well with hsCRP levels. It is a powerful independent risk determinant in the prediction of incident Diabetes.			
LIPID SCREEN, SERUM (Enzymatic)			
Cholesterol, Total	200.00	mg/dL	<200.00
Triglycerides	**201.00**	mg/dL	<150.00
HDL Cholesterol	61.00	mg/dL	>50.00
LDL Cholesterol, Calculated	98.80	mg/dL	<100.00
VLDL Cholesterol,Calculated	**40.20**	mg/dL	<30.00

Figure 4.1.1: Lipid Screen, Serum Test report

Figure 4.1.1 : The above figure about the Lipid Screen, Serum Test Report. It is a test to determine if your cholesterol level is normal or falls into a borderline, intermediate or high-risk category. It is used to monitor and screen for your risk of cardiovascular disease. This test is to help diagnose other medical conditions, such as liver disease.

IMG_20230205_01... • • •

	A	B	C	D
1	Test Name	Results	Units	Bio. Ref. Interval
2	LIPID SCREEN, SERUM (Enzymatic)			
3	Cholesterol, Total	200	mg/dL	<200.00
4	Triglycerides	201	mg/dL	<150.00
5	HDL Cholesterol	61	mg/dL	>50.00
6	LDL Cholesterol, Calculated	98.8	mg/dL	<100.00
7	VLDL Cholesterol, Calculated	40.2	mg/dL	<30.00
8				
9				

Figure 4.1.2 : Lipid Screen ,Serum Test report

Figure 4.1.2 : The above figure about the Lipid Screen, Serum Test Report. It is a test to determine if your cholesterol level is normal or falls into a borderline, intermediate or high-risk category. It is used to monitor and screen for your risk of cardiovascular disease. This test is to help diagnose other medical conditions, such as liver disease.

Test Name	Results	Units	Bio. Ref. Interval
LIVER & KIDNEY PANEL, SERUM (Spectrophotometry, Indirect ISE)			
Bilirubin Total	0.20	mg/dL	<1.00
Bilirubin Direct	0.10	mg/dL	0.00 - 0.30
Bilirubin Indirect	0.10	mg/dL	<1.10
AST (SGOT)	**41**	U/L	15.00 - 37.00
ALT (SGPT)	38	U/L	30 - 65
GGTP	39	U/L	5 - 55
Alkaline Phosphatase (ALP)	**180**	U/L	50 - 136
Total Protein	**6.20**	g/dL	6.40 - 8.20
Albumin	4.30	g/dL	3.4 - 5.0
A : G Ratio	**2.26**		0.90 - 2.00
Urea	19.00	mg/dL	14.9 - 38.5
Creatinine	0.90	mg/dL	0.6 - 1.0
Uric Acid	5.10	mg/dL	2.6 - 6.0
Calcium, Total	8.50	mg/dL	8.50 - 10.10
Phosphorus	3.50	mg/dL	2.50 - 4.90
Sodium	142.00	mEq/L	136.00 - 145.00
Potassium	3.90	mEq/L	3.50 - 5.10
Chloride	106.00	mEq/L	98.00 - 107.00
THYROID PROFILE,TOTAL, SERUM (CLIA)			
T3, Total	**2.00**	ng/mL	0.60 - 1.81
T4, Total	9.00	ug/dL	5.01 - 12.45
TSH	**8.60**	uIU/mL	0.35 - 5.50

Figure 4.2.1 : Liver &Kidney Panel Test report

Figure 4.2.1 : The above figure about the Liver &Kidney Panel Test report. These tests check the overall health of your liver. It helps to diagnose liver diseases, such as hepatitis. Monitor treatment of liver disease. It checks how badly a liver has been damaged or scarred by diseases, such as cirrhosis. It monitors the side effects of medicines. Thyroid tests are used to checks how well your thyroid is working and to find the cause of problems such as hyperthyroidism or hypothyroidism.

IMG_20230205_01...

	A	B	C	D
1	Test Name	Results	Units	Bio. Ref. Interval
2	LIVER & KIDNEY PANEL, SERUM (Spectrophotometry, Indirect ISE)			
3	Bilirubin Total	0.2	mg/dL	<1.00
4	Bilirubin Direct	0.1	mg/dL	0.00 0.30
5	Bilirubin Indirect	0.1	mg/dL	<1.10
6	AST (SGOT)	41	U/L	15.00 - 37.00
7	ALT (SGPT)	38	U/L	30 65
8	GGTP	39	U/L	5 55
9	Alkaline Phosphatase (ALP)	180	U/L	50- 136
10	Total Protein	6.2	g/dL	6.40 8.20
11	Albumin	4.3	g/dL	3.4 5.0
12	A : G Ratio	2.26		0.90 2.00
13	Urea	19	mg/dL	14.9 38.5
14	Creatinine	0.9	mg/dL	0.6 1.0
15	Uric Acid	5.1	mg/dL	2.6 6.0
16	Calcium, Total	8.5	mg/dL	8.50 10.10
17	Phosphorus	3.5	mg/dL	2.50 4.90
18	Sodium	142	mEq/L	136.00 - 145.00
19	Potassium	3.9	mEq/L	3.50 - 5.10
20	Chloride	106	mEq/L	98.00 - 107.00
21	T3, Total	2	ng/mL	0.60 1.81
22	T4, Total	9	ug/dL	5.01 - 12.45
23	TSH	8.6	ulU/mL	0.35 - 5.50

Figure 4.2.2 : Liver &Kidney Panel Test report

Figure 4.2.2 : The above figure about the Liver &Kidney Panel Test report. These tests check the overall health of your liver. It helps to diagnose liver diseases, such as hepatitis. Monitor treatment of liver disease. It checks how badly a liver has been damaged or scarred by diseases, such as cirrhosis. It monitors the side effects of medicines. Thyroid tests are used to checks how well your thyroid is working and to find the cause of problems such as hyperthyroidism or hypothyroidism.

Code:

```
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib.widgets import *
```

```
from glob import glob
# note: all the csv file names are with same name and the file name must ends with integer
# for Reading csv files in order
new_file=sorted(glob("IMG_20230205_013735.jpg*"))
# for merging all csv files into single file
merge_data=pd.concat(pd.read_csv(datafile).assign(sourcefilename=datafile)
for datafile in new_file)
# for Resetting the index of the merged csv file
merge_data=merge_data.reset_index()
# for delting the extra unwanted data in the merged csv file
del merge_data["index"]
del merge_data["Units"]
del merge_data["sourcefilename"]
# for deleting the column name Unitis if it is present in the merged csv file
for a in range(len(merge_data.columns)):
    if(merge_data.columns[a-1]=="Unitis"):
        del merge_data["Unitis"]
# assigning new column names
new_column_names=[]
for k in range(len(merge_data.columns)):
    if(k==0):
        new_column_names.append("Investigation")
    elif(k==1):
        new_column_names.append("Result")
    elif(k==2):
        new_column_names.append("Normal Ranges")
    else:
        new_column_names.append(merge_data.columns[k])
merge_data.columns=new_column_names
length=len(merge_data)
# for checking if the testname have float or nan value
test_index=[]
for i in range(length):
```

```
    if type(merge_data['Investigation'][i])==float:
        test_index.append(i)
# for storing the dataframe elements in the array form
results=[]
testname=[]
indexes=[]
Interval=[]
Results=merge_data['Result']
Testname=merge_data['Investigation']
interval=merge_data['Normal Ranges']
for k in range(len(Testname)):
    testname.append(Testname[k])
for j in range(len(Results)):
    results.append(Results[j])
for m in range(len(interval)):
    Interval.append(interval[m])
for i in range(len(test_index)):
        results.pop(test_index[i]-i)
        testname.pop(test_index[i]-i)
        Interval.pop(test_index[i]-i)
mad=len(testname)
for l in range(len(Interval)):
    if type(Interval[l])==float:
        indexes.append(l)
# for storing the test names in array form
main_testnames=[]
for k in indexes:
    main_testnames.append(merge_data['Investigation'][k])
# for deleting the main test names and its related values in the arrays
for i in range(len(indexes)):
        results.pop(indexes[i]-i)
        testname.pop(indexes[i]-i)
        Interval.pop(indexes[i]-i)
```

```
# for changing the test names if the test names are repeated
test_names=[]
for j in range(len(testname)):
    for k in range(len(testname)):
        if(j!=k):
            if(testname[j]==testname[k]):
                testname[k]=testname[k].replace(testname[k],testname[k]+str(k))
        else:
            test_names.append(testname[k])

# for removing the : in test names
test_name=[]
for i in test_names:
    i=i.replace(':',' ')
    i=i.strip()
    test_name.append(i)
result=[]
for i in results:
    if(type(i)==str):
        i=i.replace(',','.')
    result.append(float(i))
# for creating the data frame only for testname and plotting points
data={'Test Name':test_name,'Result':results}
df=pd.DataFrame(data)
df.loc[:,'Test Name']
# for plotting the graph for test names and result values
plot1=plt.plot(df.loc[:,"Test Name"],result,marker="*")
#cursor = mplcursors.cursor(plot1, hover=True)
plt.xticks(rotation=90)
plt.title("Graph for Test Names and Resultant Values")
plt.xlabel("Test Names")
plt.ylabel("Result Values")
plt.show()
```

RESULTS:

The output is a graph that generated on basis of various test names of medical health reports vs results of the reports as a numerical value respectively. The single graph is generated for different test reports. All the test reports are merged into single Data Frame , then graph is generated with the test names on the x-axis and the numerical results on the y-axis. The graph can then be used to visualize the data and identify trends or patterns over time.

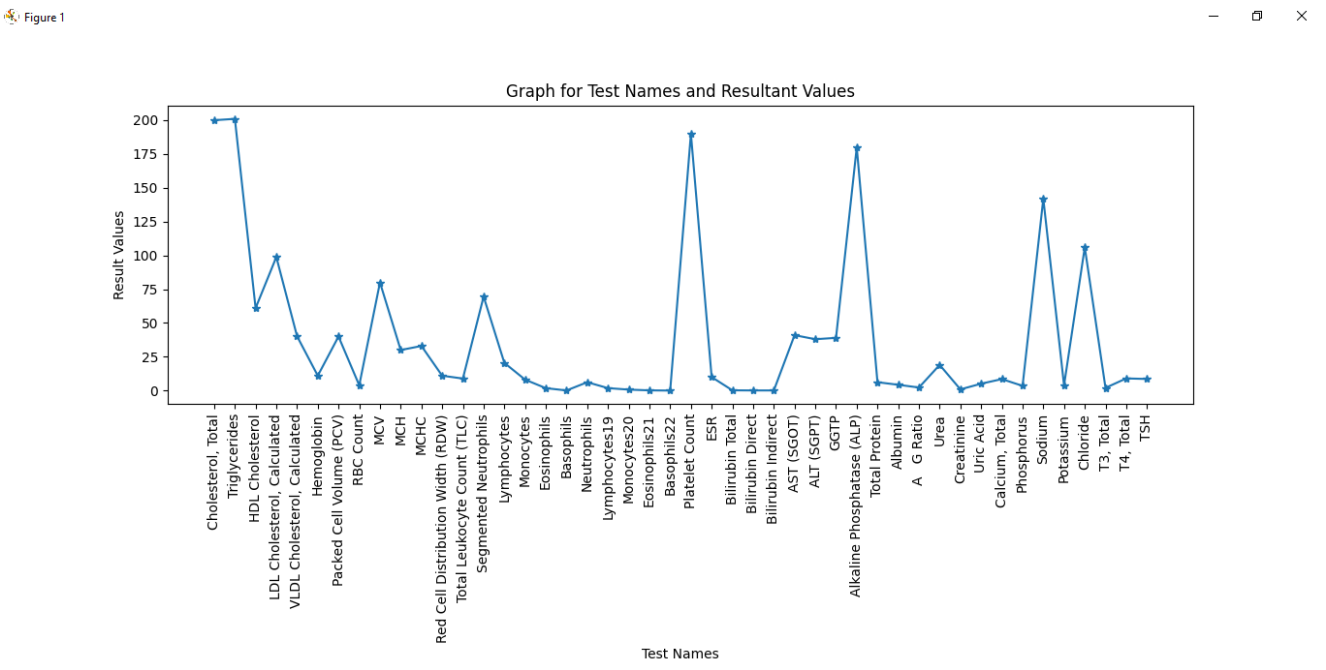


Figure 4.3: General plot for different health reports.

Figure 4.3 : The above figure about the general plot for different health reports. The graph is plotted based on the Test names on X- Axis and Result values on Y-Axis. The graph contains few similar Test Names, So the Test Names renamed by adding index value at the end of the Test Name.

CHAPTER 5:

Health Condition Plot

Introduction:

For better understanding the graph is modulated so that the patient's health can be known whether the condition is normal or above normal or below normal. And in this code the three conditions are differentiated by three different values:

Below normal → 1

Normal → 2

Above normal → 3

The above 1, 2, 3 are assigned to below normal, normal, above normal respectively within the csv file so that the graph is of three stages as it can be understandable by normal people. These are assigned by comparing the actual results of different tests with the reference ranges provided in the csv file. By satisfying the ranges conditions the values are automatically assigned.

After assigning these values, the resultant graph will have peaks some are higher and some are lower peaks. Higher peaks are above normal, lower inverted peaks are below normal, and straight lines are normal conditions of health.

Objective:

- To show the condition of health status on the plot by comparing the results and ranges in the report.

Procedure:

1. Store the normal ranges data in array form to eliminate the string data in normal ranges.
2. After removing the unwanted data it generates arrays which contains lower limit value and upper limit value and that all arrays stored in a new array.
3. Store the lower limit value and upper limit value in new array.
4. Create two new arrays to store even index values as lower limit values and odd index values as upper limit values.
5. After creating lower limit and upper limit values compare with result values and store the condition in new array.

6. Create a data new data frame with Test Names and Condition array.

7. Plot the graph with the help of Test Names and Condition.

Code:

```
# for removing the units and other string data in the Normal Ranges
lower_limit=[]
upper_limit=[]
array=[]
array1=[]
for i in Interval:
    j= i.replace('_', ' ').replace(', ', ' ').replace('-', ' ').replace('mg/dl', ' ').replace('mg/dL', ' ').replace('UP TO','0 ').replace('U/L', ' ').replace('gm/dl', ' ').replace('gms/dl', ' ').replace('mill/cumm','').replace('/uL','').replace('%', ' ').replace('Lakhs/cumm','').replace('fL','').replace('pg','').replace('gms', ' ').replace('CHILDRENS', ' ').replace('ADULTS', ' ').replace('<', '0 ').replace('>', '100 ').replace('g/dL', ' ').replace('mg/dL', ' ').replace('mEq/L', ' ').replace('ng/mL', ' ').replace('ug/dL', ' ').replace('ulU/mL', ' ').replace('mill/mm3', ' ').replace('thou/mm3', ' ').replace('mm/hr', ' ').split()
    if(len(j)>2):
        for k in range(len(j)):
            array1.append(float(j[k]))
        array1=sorted(array1)
        low=str(array1[0])
        high=str(array1[-2])
        j=[low,high]
        print(j)
    else:
        print(j)
    array.append(j)
# for converting the data into float type
array1=[]
for i in range(len(array)):
```

```
        for j in range(len(array[i])):
            array1.append(float(array[i][j]))
# for separating lower limit and upper limit values
for i in range(len(array1)):
    if i%2==0:
        lower_limit.append(array1[i])
    else:
        upper_limit.append(array1[i])
# for plotting points (Note: 1=Below normal , 2=Normal , 3=Above Normal)
plotting_points=[]
for i in range(len(result)):
    if(result[i]<lower_limit[i]):
        plotting_points.append(1)
    elif(result[i]>upper_limit[i]):
        plotting_points.append(3)
    else:
        plotting_points.append(2)

# for creating the data frame only for testname and plotting points
data={'Test Name':test_name,'condition':plotting_points}
df=pd.DataFrame(data)
df.loc[:,'Test Name']
# for plotting the graph for test names and conditions
plt.figure(figsize=(28,6))
plt.plot(df.loc[:,"Test Name"],df['condition'],marker="*")
plt.xticks(rotation=90)
ylab=["",'below normal','normal','above normal',""]
plt.gca().set_yticks(range(len(ylab)))
plt.gca().set_yticklabels(ylab)
plt.title("Graph for Test Names and Condition")
```

plt.xlabel("Test Names")

plt.ylabel("Condition")

RESULTS:

The output is a simple graph that generated on basis of various test names of medical health reports vs condition of the health reports. If the result value is below when compared to normal ranges then it is considered as below normal condition. If the result value is in between the normal ranges then it is considered as normal condition. If the result value is high when compared to normal ranges then it is considered as above normal.

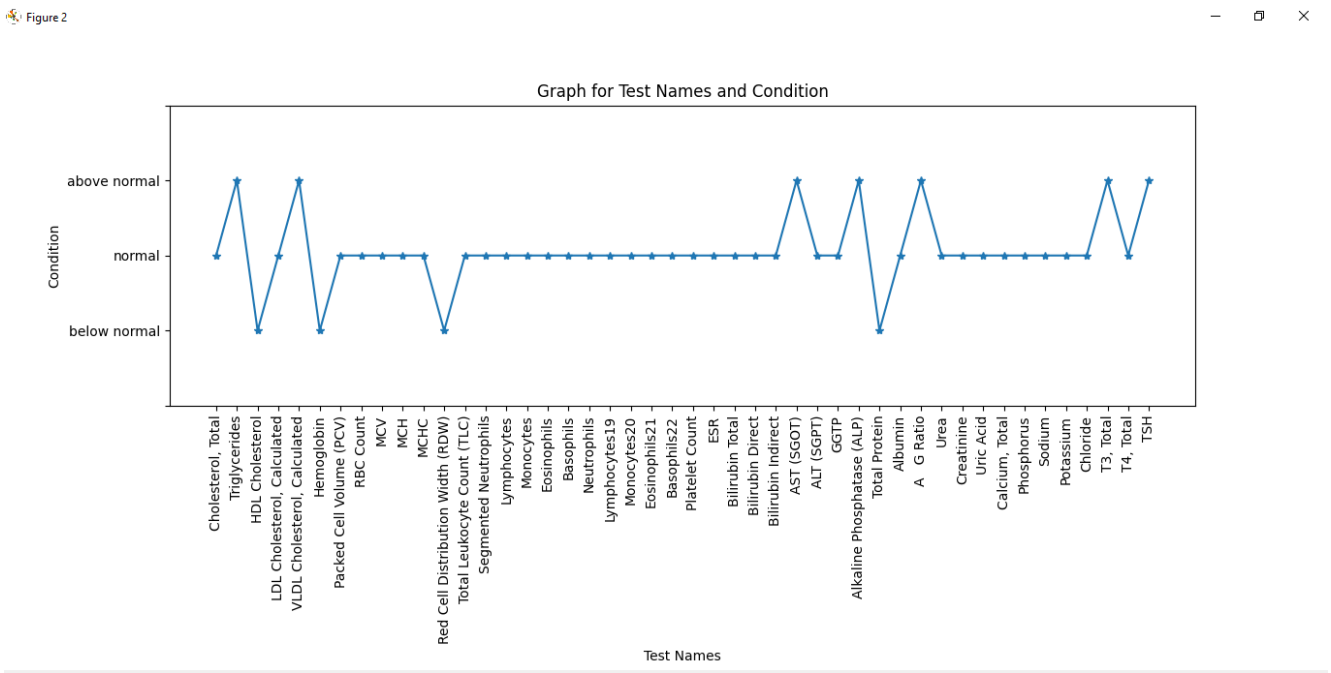


Figure 5.1 : Health condition plot .

Figure 5.1 The above figure about the Health condition plot .The graph is plotted based on the Test names on X- Axis and Condition on Y-Axis. The graph contains three stages below normal, normal, above normal as it can be understandable by normal people. Below normal indicates that the result value is less when compared to normal ranges. Normal indicates that the result value is present in between the normal ranges. Above normal indicates that the result value is high when compared to normal ranges. These are assigned by comparing the actual results of different tests with the reference ranges provided in the health report.

CHAPTER 6:

Health Report Analysis

Introduction:

We have generated a continuous health graph based on the health reports. But by seeing the graph it is difficult to say that how many tests are done for the person. With the help of main test name index values, we can classify the test names. We need a source file which contains all the data regarding the test names , conditions for the tests, reason for the condition, medicine for the problem, food culture to be followed to control the problem. The source file may contains data about the more number of tests but if the person only takes certain tests then, we have to compare the test names which is present in the health report to the source file. If the test names and conditions are matched then it creates a new data frame for the person. The new data frame contains only information about the test names which are matched in the health report. With the help of check boxes we can display the required information only.

Objective:

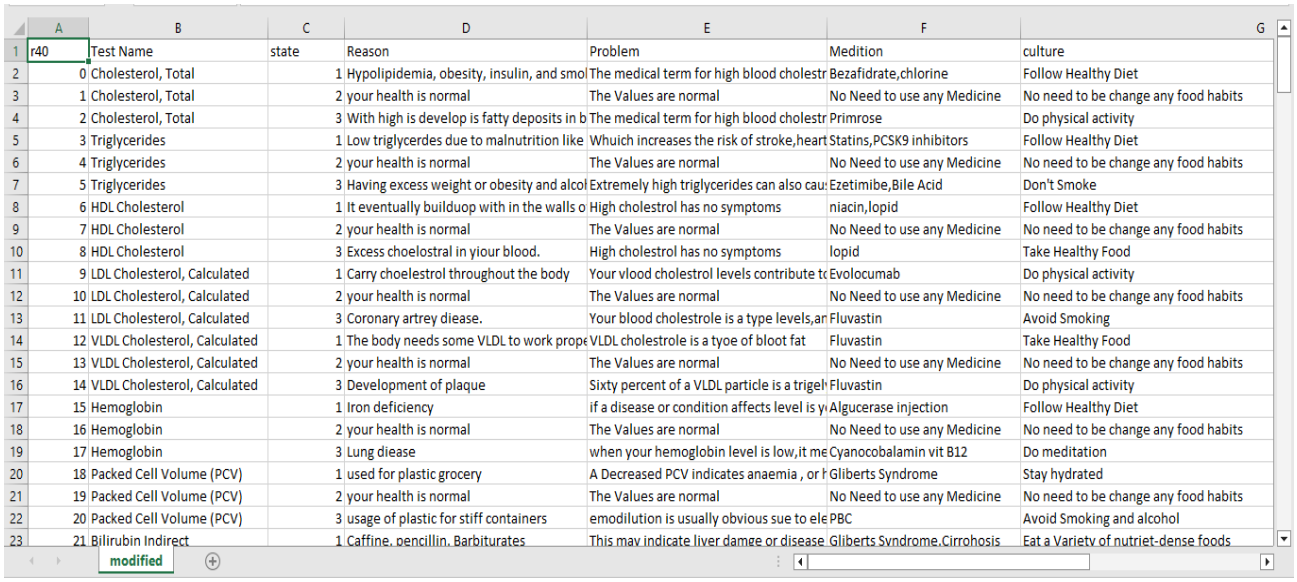
- Showing the information of reason, medicine to use, food culture to follow by the patient.

Procedure:

- Read the source file which contains the data about Test Names Problem , Reason for the problem, Medicine for the problem, food habits to follow.
- Create a new arrays to store test names, problems, Reason, medicine and culture.
- If the test names and condition is matched in the source file then the data regarding the test names , condition , Problem, Reason, Medicine, Food culture are appended in the respective created arrays.
- Create a new data frame contains test names, condition, reason, problem, medicine, food culture.
- Create a new variables to store the values of reason, medicine, food culture.
- Create a check boxes, if we click on the particular check box then the data related to the check box is displayed.

Data:

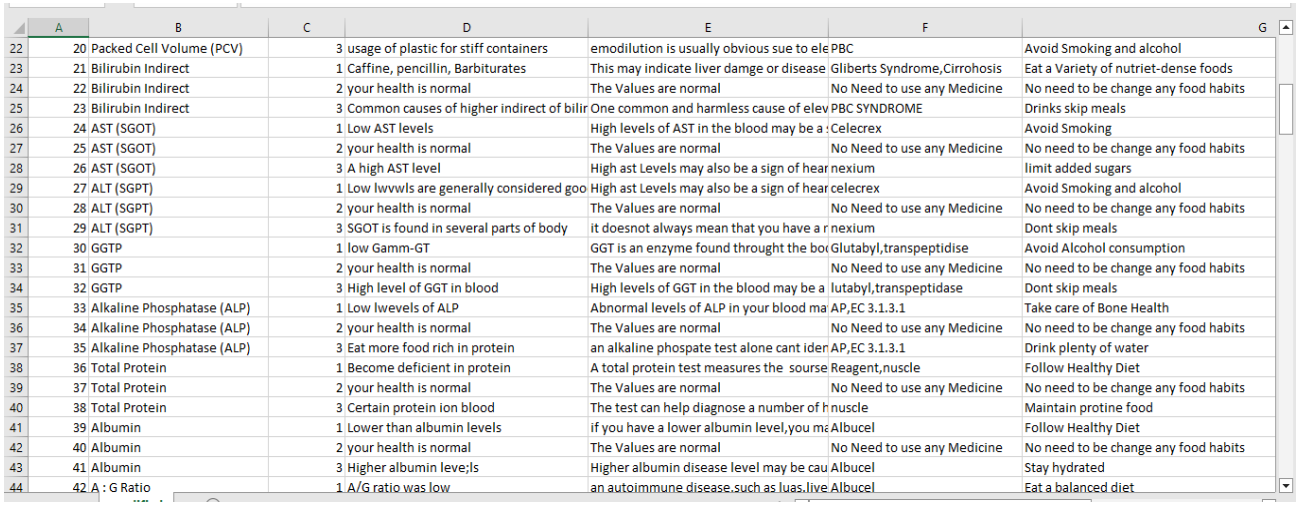
Csv file consisting medication reason problem and food culture to be followed by the patient.



r40	Test Name	state	Reason	Problem	Medition	culture
0	Cholesterol, Total	1	Hypolipidemia, obesity, insulin, and smol	The medical term for high blood cholestr	Bezafidrate,chlorine	Follow Healthy Diet
1	Cholesterol, Total	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
2	Cholesterol, Total	3	With high is develop is fatty deposits in b	The medical term for high blood cholestr	Primrose	Do physical activity
3	Triglycerides	1	Low triglycerdes due to malnutrition like	Whuich increases the risk of stroke,heart	Statins,PCSK9 inhibitors	Follow Healthy Diet
4	Triglycerides	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
5	Triglycerides	3	Having excess weight or obesity and alcol	Extremely high triglycerides can also cau	Ezetimibe,Bile Acid	Don't Smoke
6	HDL Cholesterol	1	It eventually builduop with in the walls o	High cholestrol has no symptoms	niacin,lopid	Follow Healthy Diet
7	HDL Cholesterol	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
8	HDL Cholesterol	3	Excess choelostral in yiour blood.	High cholestrol has no symptoms	lopid	Take Healthy Food
9	LDL Cholesterol, Calculated	1	Carry choelestrol throughout the body	Your vlood cholestrol levels contribute to	Evolocumab	Do physical activity
10	LDL Cholesterol, Calculated	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
11	LDL Cholesterol, Calculated	3	Coronary artrey diease.	Your blood cholestrole is a type levels,ar	Fluvastin	Avoid Smoking
12	VLDL Cholesterol, Calculated	1	The body needs some VLDL to work prope	VLDL cholestrole is a tyoe of bloot fat	Fluvastin	Take Healthy Food
13	VLDL Cholesterol, Calculated	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
14	VLDL Cholesterol, Calculated	3	Development of plaque	Sixty percent of a VLDL particle is a trigel	Fluvastin	Do physical activity
15	Hemoglobin	1	Iron deficiency	if a disease or condition affects level is y	Algucerase injection	Follow Healthy Diet
16	Hemoglobin	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
17	Hemoglobin	3	Lung diease	when your hemoglobin level is low,it me	Cyanocobalamin vit B12	Do meditation
18	Packed Cell Volume (PCV)	1	used for plastic grocery	A Decreased PCV indicates anaemia , or h	Gliberts Syndrome	Stay hydrated
19	Packed Cell Volume (PCV)	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
20	Packed Cell Volume (PCV)	3	usage of plastic for stiff containers	emodilution is usually obvious sue to ele	PBC	Avoid Smoking and alcohol
21	Bilirubin Indirect	1	Caffine, pencillin, Barbiturates	This may indicate liver damge or disease	Gliberts Syndrome,Cirrohosis	Eat a Variety of nutriet-dense foods

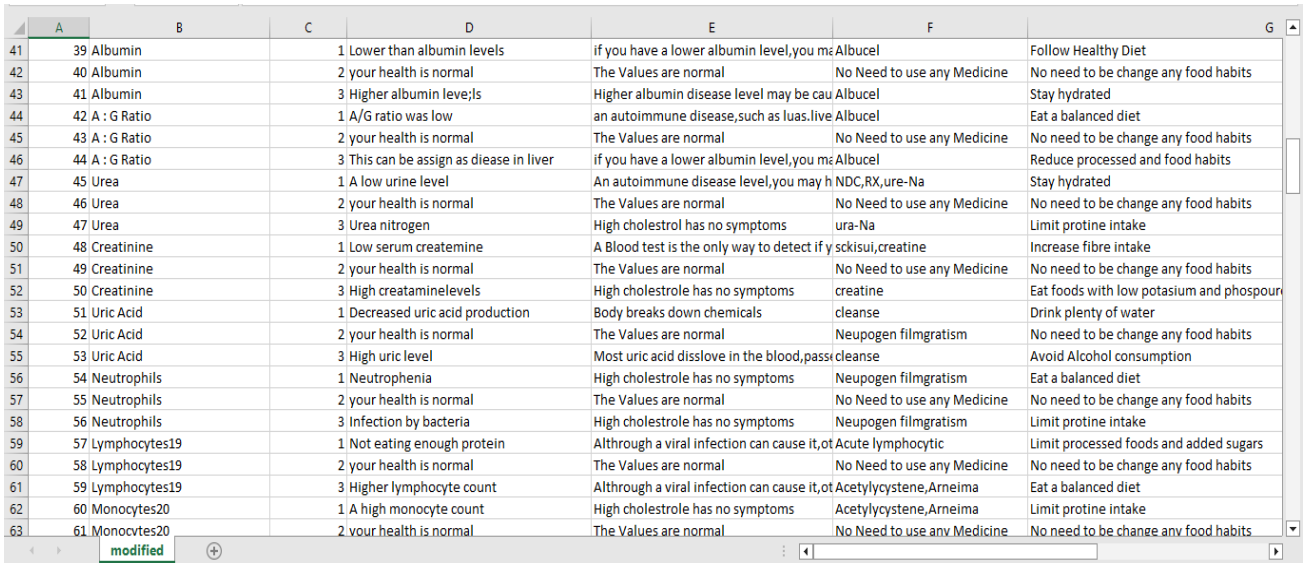
modified

Figure-6.1.1: Data regarding medicine, reason, food-culture.



20	Packed Cell Volume (PCV)	3	usage of plastic for stiff containers	emodilution is usually obvious sue to ele	PBC	Avoid Smoking and alcohol
21	Bilirubin Indirect	1	Caffine, pencillin, Barbiturates	This may indicate liver damge or disease	Gliberts Syndrome,Cirrohosis	Eat a Variety of nutriet-dense foods
22	Bilirubin Indirect	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
23	Bilirubin Indirect	3	Common causes of higher indirect of bilir	One common and harmless cause of elev	PBC SYNDROME	Drinks skip meals
24	AST (SGOT)	1	Low AST levels	High levels of AST in the blood may be a	Celecrex	Avoid Smoking
25	AST (SGOT)	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
26	AST (SGOT)	3	A high AST level	High ast Levels may also be a sign of hear	nexium	limit added sugars
27	ALT (SGPT)	1	Low lwvwls are generally considered goo	High ast Levels may also be a sign of hear	celecrex	Avoid Smoking and alcohol
28	ALT (SGPT)	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
29	ALT (SGPT)	3	SGOT is found in several parts of body	it doesnot always mean that you have a r	nexium	Dont skip meals
30	GGTP	1	low Gamm-GT	GGT is an enzyme found throught the bo	Glutabyl,transpeptidise	Avoid Alcohol consumption
31	GGTP	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
32	GGTP	3	High level of GGT in blood	High levels of GGT in the blood may be a	lutabyl,transpeptidase	Dont skip meals
33	Alkaline Phosphatase (ALP)	1	Low lwevels of ALP	Abnormal levels of ALP in your blood ma	AP,EC 3.1.3.1	Take care of Bone Health
34	Alkaline Phosphatase (ALP)	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
35	Alkaline Phosphatase (ALP)	3	Eat more food rich in protein	an alkaline phospate test alone cant iden	AP,EC 3.1.3.1	Drink plenty of water
36	Total Protein	1	Become deficient in protein	A total protein test measures the sourse	Reagent,nuscle	Follow Healthy Diet
37	Total Protein	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
38	Total Protein	3	Certain protein ion blood	The test can help diagnose a number of h	nuscle	Maintain protine food
39	Albumin	1	Lower than albumin levels	if you have a lower albumin level,you ma	Albucel	Follow Healthy Diet
40	Albumin	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
41	Albumin	3	Higher albumin leve;ls	Higher albumin disease level may be cau	Albucel	Stay hydrated
42	A : G Ratio	1	A/G ratio was low	an autoimmune disease,such as luas.live	Albucel	Eat a balanced diet

Figure-6.1.2: Data regarding medicine, reason, food-culture.



39	Albumin	1	Lower than albumin levels	if you have a lower albumin level,you ma	Albucel	Follow Healthy Diet
40	Albumin	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
41	Albumin	3	Higher albumin leve;ls	Higher albumin disease level may be cau	Albucel	Stay hydrated
42	A : G Ratio	1	A/G ratio was low	an autoimmune disease,such as luas.live	Albucel	Eat a balanced diet
43	A : G Ratio	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
44	A : G Ratio	3	This can be assign as diease in liver	if you have a lower albumin level,you ma	Albucel	Reduce processed and food habits
45	Urea	1	A low urine level	An autoimmune disease level,you may h	NDC,RX,ure-Na	Stay hydrated
46	Urea	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
47	Urea	3	Urea nitrogen	High cholestrol has no symptoms	ura-Na	Limit protine intake
48	Creatinine	1	Low serum createmine	A Blood test is the only way to detect if y	sckisui,creatine	Increase fibre intake
49	Creatinine	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
50	Creatinine	3	High creataminelevels	High cholestrole has no symptoms	creatine	Eat foods with low potasium and phospour
51	Uric Acid	1	Decreased uric acid production	Body breaks down chemicals	cleanse	Drink plenty of water
52	Uric Acid	2	your health is normal	The Values are normal	Neupogen filmgratism	No need to be change any food habits
53	Uric Acid	3	High uric level	Most uric acid disslove in the blood,pass	cleanse	Avoid Alcohol consumption
54	Neutrophils	1	Neutrophenia	High cholestrole has no symptoms	Neupogen filmgratism	Eat a balanced diet
55	Neutrophils	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
56	Neutrophils	3	Infection by bacteria	High cholestrole has no symptoms	Neupogen filmgratism	Limit protine intake
57	Lymphocytes19	1	Not eating enough protein	Althrough a viral infection can cause it,ot	Acute lymphocytic	Limit processed foods and added sugars
58	Lymphocytes19	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
59	Lymphocytes19	3	Higher lymphocyte count	Althrough a viral infection can cause it,ot	Acetylycystene,Arneima	Eat a balanced diet
60	Monocytes20	1	A high monocyte count	High cholestrole has no symptoms	Acetylycystene,Arneima	Limit protine intake
61	Monocytes20	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits

modified

Figure-6.1.3: Data regarding medicine, reason, food-culture.

	A	B	C	D	E	F	G
63	61	Monocytes20	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
64	62	Monocytes20	3	Monocytosis count is too bhigh	High cholestrole has no symptoms	Acute lymphocytic	Eat a balanced diet
65	63	Eosinophils21	1	Low eoisinophil count	emodilution is usually obvious sue to ele	Crproterone	Limit processed foods and added sugars
66	64	Eosinophils21	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
67	65	Eosinophils21	3	paracitic infection	it is sign of a certainmedical conditions,ir	Acetylycystene,Arneima	Drink plenty of water
68	66	Basophils22	1	Low bascophill level	A low nasophil is calles basophenia , it ca	SBL-BC22 CAPS	Limit protine intake
69	67	Basophils22	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
70	68	Basophils22	3	Bascophilic disorder	A low nbasohil level serve allergies,or an	SBL	Limit protine intake
71	69	Platelet Count	1	decrease in platelet production	affect bone marrow can cause low platel	Thrombocytobia	Drink plenty of water
72	70	Platelet Count	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
73	71	Platelet Count	3	Inflammation	it can be related to a genetic disorder	Thrombocytobia	Eat a balanced diet
74	72	ESR	1	A low ESR rate	if you have na infection or chronic inflam	ramipiril,Acsend	Limit protine intake
75	73	ESR	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
76	74	ESR	3	Increase in ESR rate	there may be more proteins in your cells	ramipiril,Acsend	Drink plenty of water
77	75	Bilirubin Total	1	Caffine, pencillin	if you have bilibrium , your symptoms wi	Biliruburin totale	Eat a balanced diet
78	76	Bilirubin Total	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
79	77	Bilirubin Total	3	Higher than usual level of bilimin	You can have mildly high bilibruium and	Biliruburin totale	Limit processed foods and added sugars
80	78	Bilirubin Direct	1	Caffine, pencillin	You can have mildly high bilibruium and	Biliruburin totale	Eat a balanced diet
81	79	Bilirubin Direct	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
82	80	Bilirubin Direct	3	Bilirubin	Incase the reports show increased levels	Biliruburin totale	Limit protine intake
83	81	RBC Count	1	A low red blood cells	Incase the reports show increased levels	Ferrous Ascorbate	Drink plenty of water
84	82	RBC Count	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
85	83	RBC Count	3	A high red blood cells	Incase the reports show increased levels	Zinc Sulphate tab	Moderate protein intake

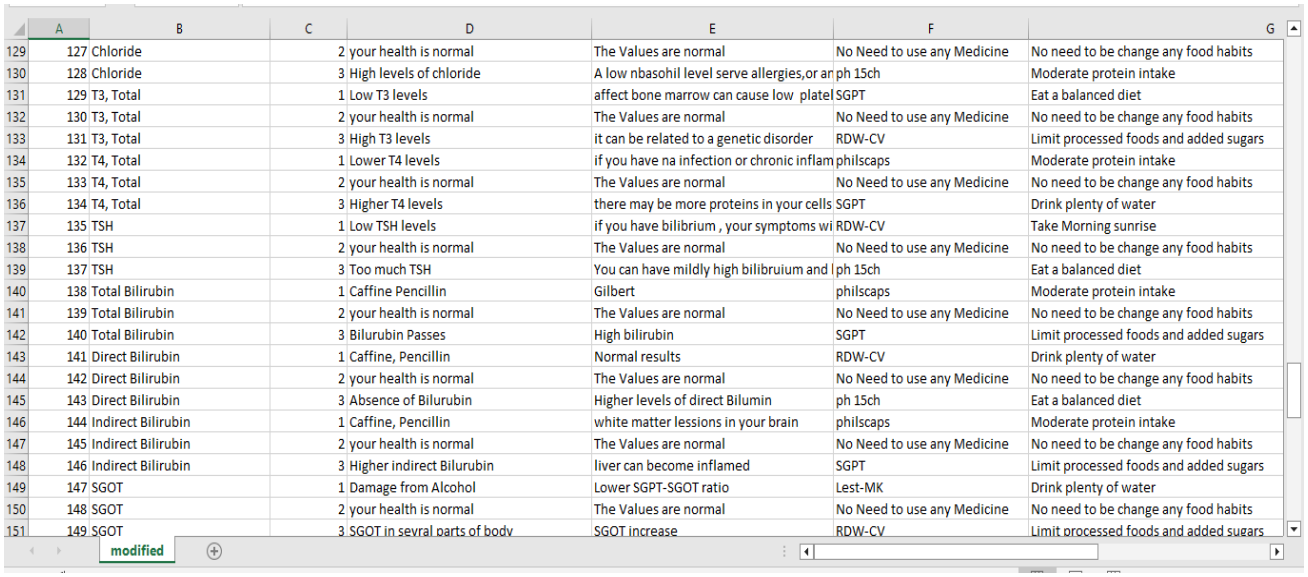
Figure-6.1.4: Data regarding medicine, reason, food-culture.

	A	B	C	D	E	F	G
85	83	RBC Count	3	A high red blood cells	Incase the reports show increased levels	Zinc Sulphate tab	Moderate protein intake
86	84	MCV	1	Low MCV	Incase the reports show increased levels	Favipiravir	Moderate protein intake
87	85	MCV	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
88	86	MCV	3	High MCV	Incase the reports show increased levels	Favipiravir	Limit protine intake
89	87	MCH	1	A low MCHJ	shuold immediately consult the doctor o	SBL-BC-22 CAPS	Moderate protein intake
90	88	MCH	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
91	89	MCH	3	High MCV	Incase the reports show increased levels	Marijuna	Eat a balanced diet
92	90	MCHC	1	A low corpuscular hemoglobin	in case of reports show increased levels (	MCH CAPS	Limit processed foods and added sugars
93	91	MCHC	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
94	92	MCHC	3	A high MCHC	in case of reports show increased levels (	MCH CAPS	Moderate protein intake
95	93	Red Cell Distribution Width (RI	1	A low RDW(below 10.02 %)	cirrohosis of the liver,inflammations etc	RDW 20.1	Limit protine intake
96	94	Red Cell Distribution Width (RI	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
97	95	Red Cell Distribution Width (RI	3	A high RDW	then it can be indicative of condition of t	RDW 20.1	Eat a balanced diet
98	96	Total Leukocyte Count (TLC)	1	A low TLC in blood	in case of reports show increased levels (	Leicocytic	Limit processed foods and added sugars
99	97	Total Leukocyte Count (TLC)	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
100	98	Total Leukocyte Count (TLC)	3	High white blood cells	in case of reports show increased levels (	Leicocytic	Eat a balanced diet
101	99	Segmented Neutrophils	1	Low neutrophil count	Incase the reports show increased levels	philscaps	Limit protine intake
102	100	Segmented Neutrophils	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
103	101	Segmented Neutrophils	3	High neutrophil	the patient should immediately consult t	Neutrophils	Limit protine intake
104	102	Lymphocytes	1	Not having enough proteins and calories	Higher albumin disease level may be cau	Chronic Lymphococytic	Moderate protein intake
105	103	Lymphocytes	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
106	104	Lymphocytes	3	Infection (bacteria, viral)	in case of reports show increased levels (	Chronic Lymphococytic	Eat a balanced diet
107	105	Monocytes	1	Monocytopenia	if you have albuminum level,you may ha	Chronic Lymphococytic	Limit protine intake

Figure-6.1.5: Data regarding medicine, reason, food-culture.

	A	B	C	D	E	F	G
107	105	Monocytes	1	Monocytopenia	if you have albuminum level,you may ha	Chronic Lymphococytic	Limit protine intake
108	106	Monocytes	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
109	107	Monocytes	3	High monocyte count	if you have albuminum level,you may ha	anemia	Moderate protein intake
110	108	Eosinophils	1	less Eosinopenia	if you have albuminum level,you may ha	philscaps	Eat a balanced diet
111	109	Eosinophils	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
112	110	Eosinophils	3	Pheripheral blood eosinphilia	Althrough a viral infection can cause it,ot	philscaps	Limit processed foods and added sugars
113	111	Basophils	1	Bascopenia	High cholestrole has no symptoms	philscaps	Limit protine intake
114	112	Basophils	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
115	113	Basophils	3	Bascophilia	High cholestrole has no symptoms	philscaps	Moderate protein intake
116	114	Calcium, Total	1	Vitamin D deficiency	emodilution is usually obvious sue to ele	D3,Blackmores	Moderate protein intake
117	115	Calcium, Total	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
118	116	Calcium, Total	3	Hypercalcemia	it is sign of a certainmedical conditions,ir	D3,Blackmores	Eat a balanced diet
119	117	Phosphorus	1	Diabeties starvation	A low nasophil is calles basophenia , it ca	RDW-CV	Moderate protein intake
120	118	Phosphorus	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
121	119	Phosphorus	3	High phosphorus	A low nbasohil level serve allergies,or an	ph 15ch	Limit processed foods and added sugars
122	120	Sodium	1	Chronic, seveare vomting or diahearia	affect bone marrow can cause low platel	ceftriaxone sodique	Limit protine intake
123	121	Sodium	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
124	122	Sodium	3	Dehydration	it can be related to a genetic disorder	ph 15ch	Eat a balanced diet
125	123	Potassium	1	Low potassium	if you have na infection or chronic inflam	ph 15ch	Moderate protein intake
126	124	Potassium	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
127	125	Potassium	3	High potassium	there may be more proteins in your cells	ph 15ch	Limit processed foods and added sugars
128	126	Chloride	1	Low levels of chloride	if you have bilibrium , your symptoms wi	RDW-CV	Moderate protein intake
129	127	Chloride	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits

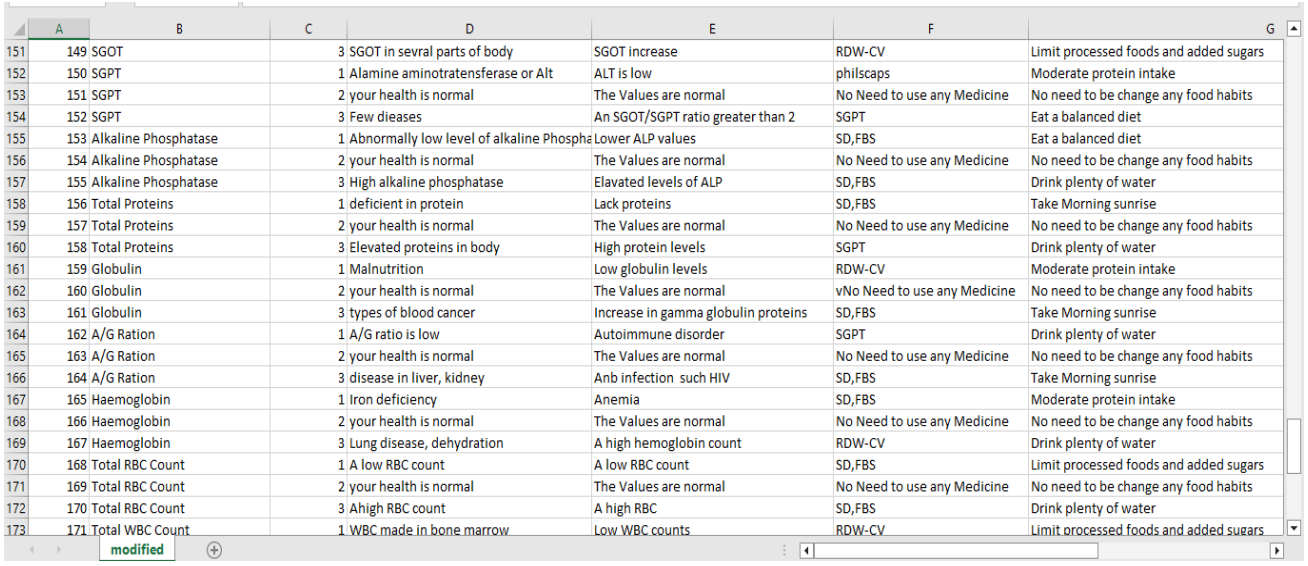
Figure-6.1.6: Data regarding medicine, reason, food-culture.



	A	B	C	D	E	F	G
129	127	Chloride	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
130	128	Chloride	3	High levels of chloride	A low nbasohil level serve allergies,or ar	ph 15ch	Moderate protein intake
131	129	T3, Total	1	Low T3 levels	affect bone marrow can cause low platel	SGPT	Eat a balanced diet
132	130	T3, Total	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
133	131	T3, Total	3	High T3 levels	it can be related to a genetic disorder	RDW-CV	Limit processed foods and added sugars
134	132	T4, Total	1	Lower T4 levels	if you have na infection or chronic inflam	philscaps	Moderate protein intake
135	133	T4, Total	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
136	134	T4, Total	3	Higher T4 levels	there may be more proteins in your cells	SGPT	Drink plenty of water
137	135	TSH	1	Low TSH levels	if you have bilibrium , your symptoms wi	RDW-CV	Take Morning sunrise
138	136	TSH	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
139	137	TSH	3	Too much TSH	You can have mildly high bilibruium and	ph 15ch	Eat a balanced diet
140	138	Total Bilirubin	1	Caffine Pencillin	Gilbert	philscaps	Moderate protein intake
141	139	Total Bilirubin	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
142	140	Total Bilirubin	3	Bilurubin Passes	High bilirubin	SGPT	Limit processed foods and added sugars
143	141	Direct Bilirubin	1	Caffine, Pencillin	Normal results	RDW-CV	Drink plenty of water
144	142	Direct Bilirubin	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
145	143	Direct Bilirubin	3	Absence of Bilurubin	Higher levels of direct Bilumin	ph 15ch	Eat a balanced diet
146	144	Indirect Bilirubin	1	Caffine, Pencillin	white matter lessions in your brain	philscaps	Moderate protein intake
147	145	Indirect Bilirubin	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
148	146	Indirect Bilirubin	3	Higher indirect Bilurubin	liver can become inflamed	SGPT	Limit processed foods and added sugars
149	147	SGOT	1	Damage from Alcohol	Lower SGPT-SGOT ratio	Lest-MK	Drink plenty of water
150	148	SGOT	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
151	149	SGOT	3	SGOT in sevral parts of body	SGOT increase	RDW-CV	Limit processed foods and added sugars

modified

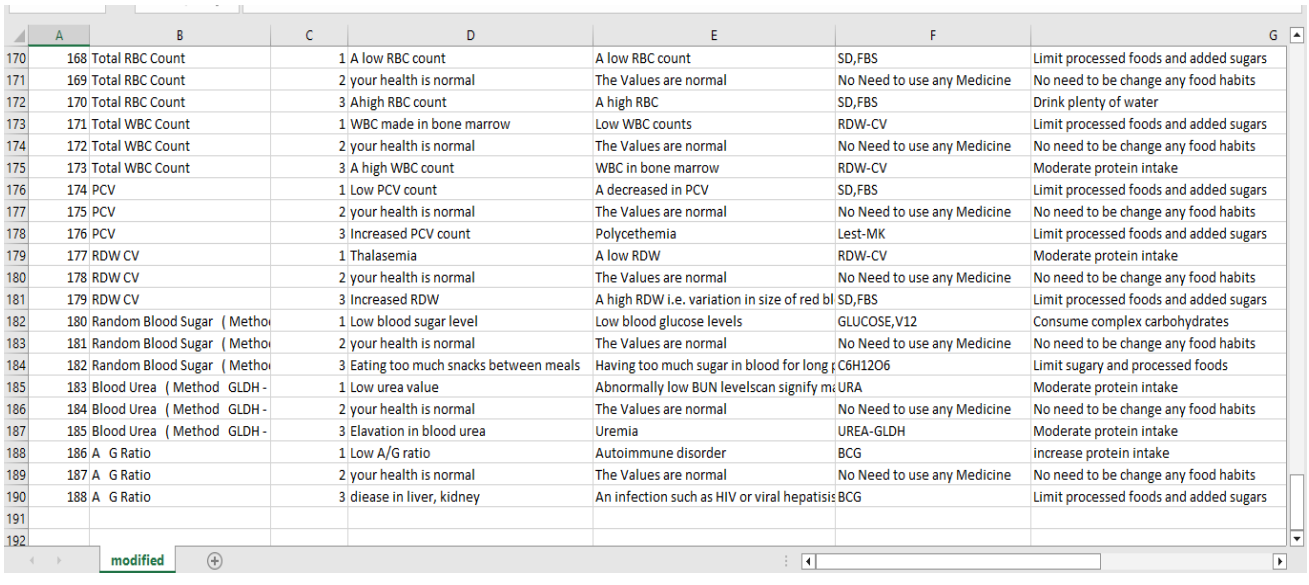
Figure-6.1.7: Data regarding medicine, reason, food-culture.



	A	B	C	D	E	F	G
151	149	SGOT	3	SGOT in sevral parts of body	SGOT increase	RDW-CV	Limit processed foods and added sugars
152	150	SGPT	1	Alamine aminotratensferase or Alt	ALT is low	philscaps	Moderate protein intake
153	151	SGPT	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
154	152	SGPT	3	Few dieases	An SGOT/SGPT ratio greater than 2	SGPT	Eat a balanced diet
155	153	Alkaline Phosphatase	1	Abnormally low level of alkaline Phospha	Lower ALP values	SD,FBS	Eat a balanced diet
156	154	Alkaline Phosphatase	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
157	155	Alkaline Phosphatase	3	High alkaline phosphatase	Elavated levels of ALP	SD,FBS	Drink plenty of water
158	156	Total Proteins	1	deficient in protein	Lack proteins	SD,FBS	Take Morning sunrise
159	157	Total Proteins	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
160	158	Total Proteins	3	Elevated proteins in body	High protein levels	SGPT	Drink plenty of water
161	159	Globulin	1	Malnutrition	Low globulin levels	RDW-CV	Moderate protein intake
162	160	Globulin	2	your health is normal	The Values are normal	vNo Need to use any Medicine	No need to be change any food habits
163	161	Globulin	3	types of blood cancer	Increase in gamma globulin proteins	SD,FBS	Take Morning sunrise
164	162	A/G Ration	1	A/G ratio is low	Autoimmune disorder	SGPT	Drink plenty of water
165	163	A/G Ration	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
166	164	A/G Ration	3	disease in liver, kidney	Anb infection such HIV	SD,FBS	Take Morning sunrise
167	165	Haemoglobin	1	Iron deficiency	Anemia	SD,FBS	Moderate protein intake
168	166	Haemoglobin	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
169	167	Haemoglobin	3	Lung disease, dehydration	A high hemoglobin count	RDW-CV	Drink plenty of water
170	168	Total RBC Count	1	A low RBC count	A low RBC count	SD,FBS	Limit processed foods and added sugars
171	169	Total RBC Count	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
172	170	Total RBC Count	3	Ahigh RBC count	A high RBC	SD,FBS	Drink plenty of water
173	171	Total WBC Count	1	WBC made in bone marrow	Low WBC counts	RDW-CV	Limit processed foods and added sugars

modified

Figure-6.1.8: Data regarding medicine, reason, food-culture.



	A	B	C	D	E	F	G
170	168	Total RBC Count	1	A low RBC count	A low RBC count	SD,FBS	Limit processed foods and added sugars
171	169	Total RBC Count	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
172	170	Total RBC Count	3	Ahigh RBC count	A high RBC	SD,FBS	Drink plenty of water
173	171	Total WBC Count	1	WBC made in bone marrow	Low WBC counts	RDW-CV	Limit processed foods and added sugars
174	172	Total WBC Count	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
175	173	Total WBC Count	3	A high WBC count	WBC in bone marrow	RDW-CV	Moderate protein intake
176	174	PCV	1	Low PCV count	A decreased in PCV	SD,FBS	Limit processed foods and added sugars
177	175	PCV	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
178	176	PCV	3	Increased PCV count	Polycethemia	Lest-MK	Limit processed foods and added sugars
179	177	RDW CV	1	Thalasemia	A low RDW	RDW-CV	Moderate protein intake
180	178	RDW CV	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
181	179	RDW CV	3	Increased RDW	A high RDW i.e. variation in size of red bl	SD,FBS	Limit processed foods and added sugars
182	180	Random Blood Sugar (Metho	1	Low blood sugar level	Low blood glucose levels	GLUCOSE,V12	Consume complex carbohydrates
183	181	Random Blood Sugar (Metho	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
184	182	Random Blood Sugar (Metho	3	Eating too much snacks between meals	Having too much sugar in blood for long p	C6H12O6	Limit sugary and processed foods
185	183	Blood Urea (Method GLDH -	1	Low urea value	Abnormally low BUN levelscan signify ma	URA	Moderate protein intake
186	184	Blood Urea (Method GLDH -	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
187	185	Blood Urea (Method GLDH -	3	Elavation in blood urea	Uremia	UREA-GLDH	Moderate protein intake
188	186	A G Ratio	1	Low A/G ratio	Autoimmune disorder	BCG	increase protein intake
189	187	A G Ratio	2	your health is normal	The Values are normal	No Need to use any Medicine	No need to be change any food habits
190	188	A G Ratio	3	diease in liver, kidney	An infection such as HIV or viral hepatisis	BCG	Limit processed foods and added sugars
191							
192							

modified

Figure-6.1.9: Data regarding medicine, reason, food-culture.

Figure 6.1.1 to Figure 6.1.9 : The above all figures contains the information about the Test Names, Reason, Problem, Medicine and Culture.

Code:

```
# source_file is the main source file
# for reading the source file
health_file=pd.read_csv("source_file.csv")
new_test_names=[]
problem_d=[]
new_med=[]
new_cul=[]
state=[]
reason=[]
for i in range(len(df['Test Name'])):
    for j in range(len(health_file['Test Name'])):
        if(df['Test Name'][i]==health_file['Test Name'][j]):
            if(df['condition'][i]==health_file['state'][j]):
                new_test_names.append(health_file['Test Name'][j])
                #problem_d.append(health_file['Problem'][j])
                new_med.append(health_file['Medition'][j])
                new_cul.append(health_file['culture'][j])
                state.append(health_file['state'][j])
                reason.append(health_file['Reason'][j])
new_data_sheet={'Test
Name':new_test_names,'state':state,'Reason':reason,'Medition':new_med,'Culture':new_cul}
newdf=pd.DataFrame(new_data_sheet)
df.loc[:,'Test Name']
fig = plt.figure()
gs0 = gridspec.GridSpec(1,1,figure=fig)
gs00 = gridspec.GridSpecFromSubplotSpec(5,4, subplot_spec=gs0[0])
ax1 = fig.add_subplot(gs00[:-1, :-1])
i=df['Test Name']
```

```
j=df['condition']

newdf.loc[:,'Reason']

newdf.loc[:,'Medition']

newdf.loc[:,'Culture']

tt=newdf['Reason'].values

tt1=newdf['Medition'].values

tt2=newdf['Culture'].values

for k in range(len(indexes)):

    for l in range(len(indexes)):

        if(k<len(indexes)-1):

            if(k==l):

                if(k==0):

                    plt.plot(df.loc[:,"Test
Name"][indexes[k]:indexes[k+1]],df['condition'][indexes[k]:indexes[k+1]],marker="*")

                else:

                    plt.plot(df.loc[:,"Test Name"][indexes[k]-l:indexes[k+1]-l],df['condition'][indexes[k]-
l:indexes[k+1]-l],marker="*")

        else:

            plt.plot(df.loc[:,"Test Name"][indexes[k]-k:len(df.loc[:,'Test Name'])],df['condition'][indexes[k]-
k:len(df['condition'])],marker="*")

scatter = plt.scatter(i,j,marker="*")

plt.xticks(rotation=90)

ylab=["",'below\nnormal','normal','above\nnormal',"",""]

plt.gca().set_yticks(range(len(ylab)))

plt.gca().set_yticklabels(ylab)

plt.title("Total Health Analysis")

plt.xlabel("Test Names")

plt.ylabel("Condition")

#leg = ax1.legend(fancybox=True, shadow=True)

legend=plt.legend(main_testnames,loc='upper right')

ax2 = fig.add_subplot(gs00[:,-1])

plt.axis('off')

texth=ax2.text(0.02, 0.75,"Reason
:",horizontalalignment='left',verticalalignment='top',wrap='True',fontsize=14,transform=ax2.transAxes)
```

```
text1h=ax2.text(0.02, 0.50,"Medicine
:",horizontalalignment='left',verticalalignment='top',wrap='True',fontsize=14,transform=ax2.transAxes)

text2h=ax2.text(0.02, 0.25,"Food Culture
:",horizontalalignment='left',verticalalignment='top',wrap='True',fontsize=14,transform=ax2.transAxes)

text3h=ax2.text(0.02, 0.99,"Problem
:",horizontalalignment='left',verticalalignment='top',wrap='True',fontsize=14,transform=ax2.transAxes)

text_display = ax1.annotate("", xy=(0,0), xytext=(5,5),textcoords="offset
points",bbox=dict(boxstyle='round',fc='linen',ec='k',lw=1),wrap='True')

text_display.set_visible(True)

text_display1 = ax2.annotate("", xy=(0.02,0.55), xytext=(5,5),textcoords="offset
points",bbox=dict(boxstyle='round',fc='linen',ec='k',lw=1),wrap='True')

text_display1.set_visible(True)

text_display2 = ax2.annotate("", xy=(0.02,0.30), xytext=(5,5),textcoords="offset
points",bbox=dict(boxstyle='round',fc='linen',ec='k',lw=1),wrap='True')

text_display2.set_visible(True)

text_display3 = ax2.annotate("", xy=(0.02,0.05), xytext=(5,5),textcoords="offset
points",bbox=dict(boxstyle='round',fc='linen',ec='k',lw=1),wrap='True')

text_display3.set_visible(True)

ax3 = fig.add_subplot(gs00[:1,:1])

labels=["Reason","Medicine","Food culture"]

activated = [False,False,False]

check = CheckButtons(ax3, labels, activated)

label_index_array=[]

def func(label):

    index = labels.index(label)

    print(index)

    if index==0:

        label_index_array.append(0)

    elif index==1:

        label_index_array.append(1)

    elif index==2:

        label_index_array.append(2)

  re=label_index_array.count(0)

    me=label_index_array.count(1)

    cu=label_index_array.count(2)

    if(label==labels[0]):
```

```
    if(re%2!=0):
        def motion_hover(event):
            annotation_visbility = text_display.get_visible()
            if event.inaxes == ax1:
                is_contained, annotation_index = scatter.contains(event)
                if is_contained:
                    data_point_location = scatter.get_offsets()[annotation_index['ind'][0]]
                    text_display.xy = data_point_location
                    text_label1 = '{}'.format([tt[n] for n in annotation_index['ind']])
                        text_display1.set_text(text_label1)
                        text_display1.set_visible(True)
                    fig.canvas.draw_idle()
                else:
                    if annotation_visbility:
                        text_display1.set_visible(False)
                        fig.canvas.draw_idle()
        fig.canvas.mpl_connect('motion_notify_event', motion_hover)
    else:
        def hover(event):
            text_display1.set_visible(False)
        fig.canvas.mpl_connect("motion_notify_event", hover)
if(label==labels[1]):
    if(me%2!=0):
        def motion_hover(event):
            annotation_visbility = text_display.get_visible()
            if event.inaxes == ax1:
                is_contained, annotation_index = scatter.contains(event)
                if is_contained:
                    data_point_location = scatter.get_offsets()[annotation_index['ind'][0]]
                    text_display.xy = data_point_location
                    text_label2 = '{}'.format([tt1[n] for n in annotation_index['ind']])
                    text_display2.set_text(text_label2)
                    text_display2.set_visible(True)
```

```
                    fig.canvas.draw_idle()
                else:
                        if annotation_visbility:
                                text_display2.set_visible(False)
                        fig.canvas.draw_idle()
            fig.canvas.mpl_connect('motion_notify_event', motion_hover)
        else:
            def hover(event):
                text_display2.set_visible(False)
            fig.canvas.mpl_connect("motion_notify_event", hover)
    if(label==labels[2]):
        if(cu%2!=0):
            def motion_hover(event):
                annotation_visbility = text_display.get_visible()
                if event.inaxes == ax1:
                    is_contained, annotation_index = scatter.contains(event)
                    if is_contained:
                        data_point_location = scatter.get_offsets()[annotation_index['ind'][0]]
                        text_display.xy = data_point_location
                        text_label3 = '{}'.format([tt2[n] for n in annotation_index['ind']])
                        text_display3.set_text(text_label3)
                        text_display3.set_visible(True)
                        fig.canvas.draw_idle()
                    else:
                        if annotation_visbility:
                            text_display3.set_visible(False)
                            fig.canvas.draw_idle()
            fig.canvas.mpl_connect('motion_notify_event', motion_hover)
        else:
            def hover(event):
                text_display3.set_visible(False)
            fig.canvas.mpl_connect("motion_notify_event", hover)
check.on_clicked(func)
```

Results:

In this we have plotted line plot and scatter plot in single graph. Line plot is used to represent the condition and scatter plot is used to represent the reason , medicine, culture when cursor is moved to particular point or particular test name. The X-axis of the scatter plot can represent the different tests that were performed during the complete body health check-up. The Y-axis can represent the results of these tests, which can be categorized as normal, below normal, and above normal. And the check boxes are also mentioned so that by clicking on check box data related to the selected check box will be displayed on left side of the output. When the cursor is moved onto the peaks the data i.e. reason problem or medicine are displayed. Each colour indicates unique test names.

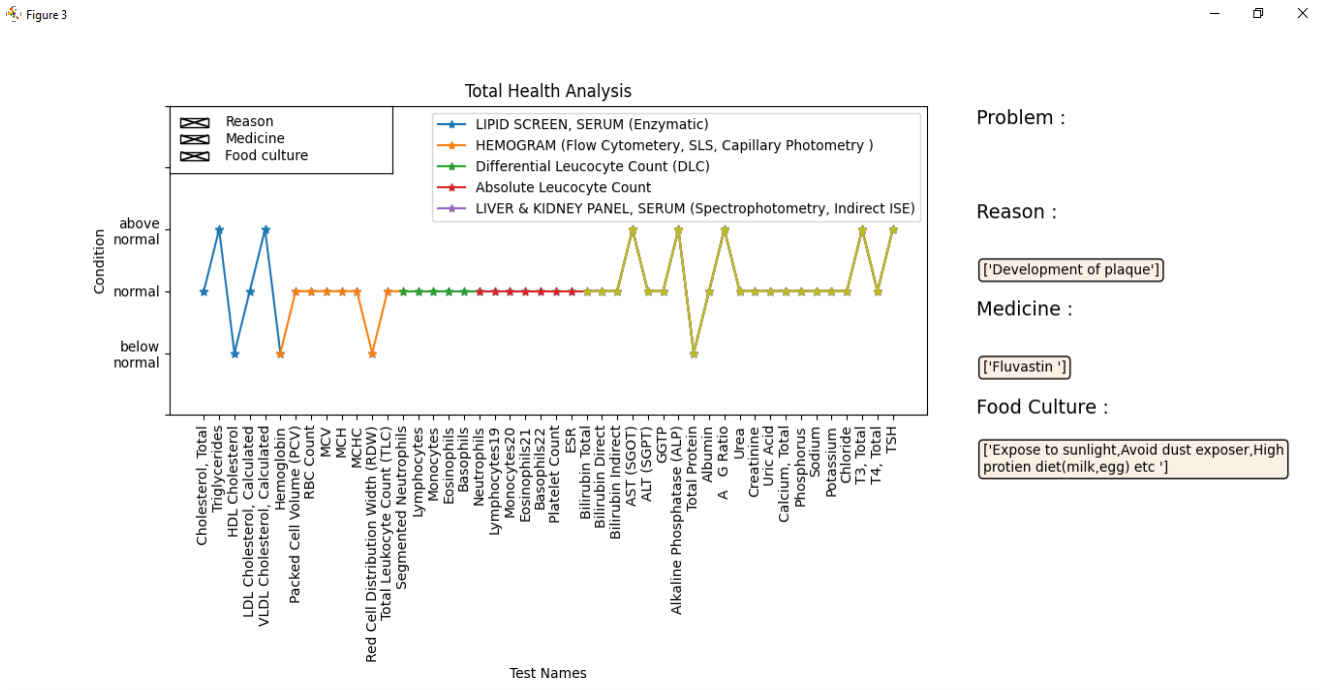


Figure-6.2: Health Report Analysis

Figure 6.2 : The above figure about the total health analysis. The graph is plotted based on the Test names on X- Axis and Condition on Y-Axis. The graph contains three stages below normal, normal, above normal. The graph contains checkboxes which are used to display the text which is necessary by the person. If cursor is moved to particular point(test name) then only it shows information regarding the Reason, Medicine, Food culture based on the check boxes input . Legends data represents the main test names in the health reports.

CHAPTER 7:

Situating Problem

Introduction:

We have generated the general plot with reports, and health status of a particular patient y classifying the health status into three conditions. These include below normal, normal, ad above normal, so that anyone can understand it easily whether they get affected or not. And the we prescribed the medicine along with food culture. Now in this, we are locating the problem that where it is situated. By clicking o the legend is shows the problem that where it is situated.

Objective:

- To locate the problem that where it is situated regarding the test report.

Procedure:

- Create a new array to store the main test name indexes.
- Append the index values of main test names and data frame last index value.
- Create a new array to store index differences and append the index difference values in that array.
- Create a new array to store legends data .
- If all the result values about a particular test are in between the normal ranges then append “NO SIGNIFICANT ABNORMALITY” otherwise store the test name in the particular test.
- Assign the legends data to main test names in the legends of the graph.
- If we click on the test names in the legends then it displays where the problem is situated.

Code:

```
# creating new indexes for
new_indexes=[]
for i in range(len(indexes)):
  if i==0:
     if(indexes[i]!=0):
```

```
            new_indexes.append(0)
              new_indexes.append(indexes[i])
        else:
              new_indexes.append(0)
     else:
      new_indexes.append(indexes[i])
if new_indexes[len(new_indexes)-1]!=len(Interval):
   new_indexes.append(mad)
# for index differences:
id=[]
for k in range(len(new_indexes)):
   if k<len(new_indexes)-1:
      if indexes[0]!=0:
        id.append(new_indexes[k+1]-new_indexes[k])
      else:
        id.append(new_indexes[k+1]-new_indexes[k]-1)
legends_data=[]
for k in range(len(new_indexes)):
   if k<len(new_indexes)-1:
      c=0
      problem_points=[]
      for l in range(new_indexes[k]-k,new_indexes[k+1]-(k+1)):
        if plotting_points[l]==2:
          c+=1
        else:
          problem_points.append(test_name[l])
      if c!=(id[k]):
          z="problem at",problem_points
          legends_data.append(z)
      else:
```

```
        legends_data.append("NO SIGNIFICANT ABNORMALITY")
main_test_names=legend.get_lines()
for k in main_test_names:
    k.set_picker(True)
    k.set_pickradius(10)
graphs={}
for k in range(len(main_testnames)):
    main_testnames[k]=plt.text(0.02,
0.95,legends_data[k],horizontalalignment='left',verticalalignment='top',wrap='True',fontsize=12,transform=ax2.
transAxes)
    main_testnames[k].set_visible(False)
for k in range(len(main_test_names)):
    graphs[main_test_names[k]]=main_testnames[k]
def on_pick(event):
    leg=event.artist
    isVisible=leg.get_visible()
    graphs[leg].set_visible(isVisible)
    legend.set_visible(isVisible)
    fig.canvas.draw()
    def hover(event):
        for k in range(len(main_testnames)):
            main_testnames[k].set_visible(False)
    fig.canvas.mpl_connect("motion_notify_event", hover)
plt.connect('pick_event',on_pick)
```

Results:

The main test names are classified based on the index values. If all the values are normal in between the main test names then there is no problem in the particular test report otherwise there is a problem in the test . If the result value is not in the normal range then the index of the test name is stored in a array. The problem data is assigned to legends data .legends acts like switch ion this case. If we click on the legends data then where the problem is situated is displayed .

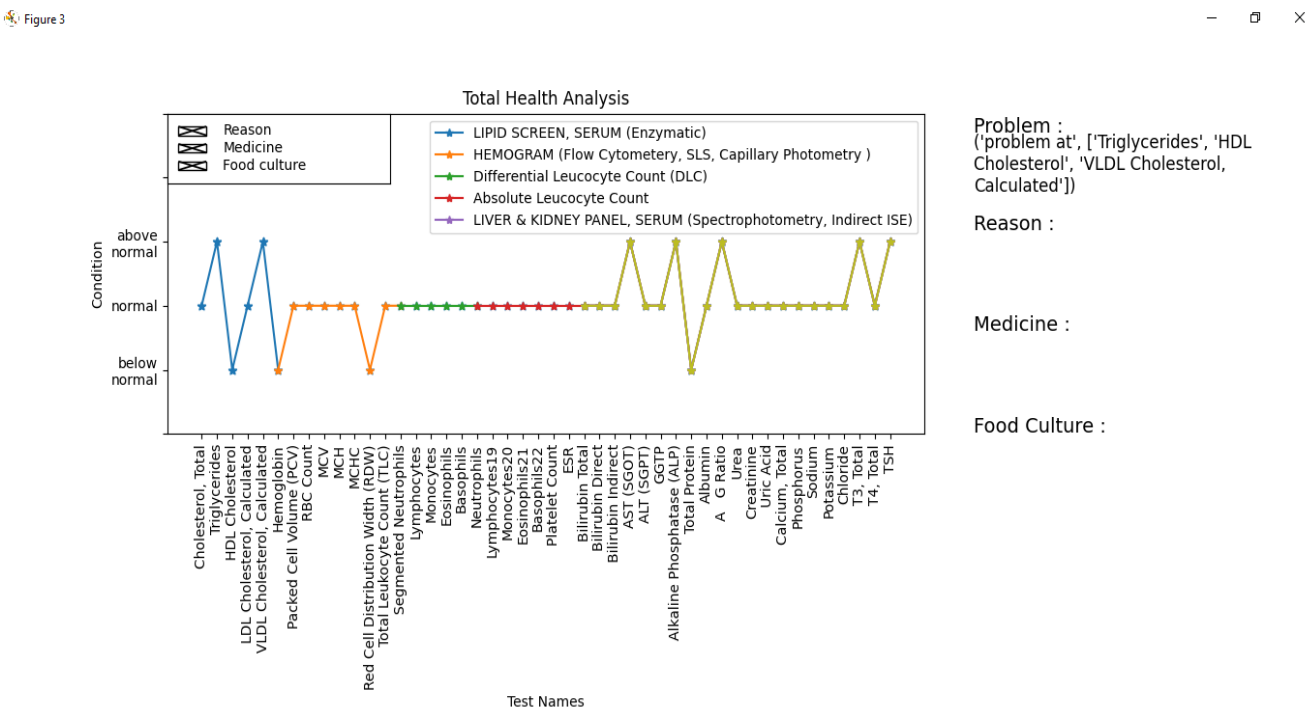


Figure 7.1: Situating Problem

Figure 7.1 : The above figure about the situating a problem. The graph is plotted based on the Test names on X- Axis and Condition on Y-Axis. The graph contains three stages below normal, normal, above normal. The graph contains checkboxes which are used to display the text which is necessary by the person. If cursor is moved to particular point(test name) then only it shows information regarding the Reason, Medicine, Food culture based on the check boxes input . Legends data represents the main test names in the health reports. When we click on legends data then it displays where the problem is situated for the particular main Test Name.

CHAPTER 8:

CONCLUSION

An automatic health graph generator can be a useful tool in healthcare settings for tracking and monitoring patient health data over time. By automating the graph generation process, healthcare professionals can save time and resources while also ensuring accuracy and consistency in the presentation of data. However, it is important to ensure that the data being used for graph generation is reliable and meaningful, and that appropriate data privacy and security measures are in place. Additionally, healthcare professionals should interpret the graphs in the context of the patient's overall health status and medical history, and not rely solely on the graphs for clinical decision-making. While an automatic health graph generator can be a valuable tool for presenting and visualizing patient health data over time, healthcare professionals should use it as a supplement to their clinical judgment and expertise, rather than relying solely on the graphs for making medical decisions. The graphs can provide useful insights and trends in the patient's health data, but they should not be used as the only basis for diagnosis or treatment decisions. Healthcare professionals should interpret the graphs in the context of the patient's overall health status, medical history, and any other relevant factors, and make decisions based on a comprehensive assessment of all available information.

CHAPTER 9:

FUTURE SCOPE

The future scope of automatic health graph generator is promising, as healthcare technology continues to evolve and improve. Some potential areas for development include:

1. Integration with electronic health records (EHRs): Automatic health graph generators could be integrated with EHRs to provide real-time tracking and visualization of patient health data. This could help healthcare professionals to quickly identify trends and patterns in a patient's health data and make informed decisions about treatment and care.

2. Machine learning and artificial intelligence (AI): With the help of machine learning and AI, automatic health graph generators could be improved to provide more accurate and personalized insights based on individual patient data. These technologies could also help to identify risk factors and predict potential health issues before they occur.

3. Wearable technology integration: The integration of wearable technology such as fitness trackers and smartwatches could allow for automatic health graph generators to capture real-time data from patients, providing more comprehensive and up-to-date health information.

4. Patient engagement and self-monitoring: Automatic health graph generators could be used to empower patients to take a more active role in monitoring and tracking their own health data. Patients could be provided with access to their own health graphs, which could help them to better understand their health status and make informed decisions about their own care.

Overall, the future of automatic health graph generators looks promising, with potential advancements in technology and healthcare data management enabling more accurate and personalized health data visualization for both healthcare professionals and patients alike.

CHAPTER 10:

REFERECES

1. A. Tengli, Y. Yang, and N. L. Ma, “Learning table extraction from examples,” in Proceedings of the 20th international conference on Computational Linguistics. Association for Computational Linguistics, 2004, p. 987.
2. M. O. Perez-Arriaga, T. Estrada, and S. Abad-Mota, “Tao: system for table detection and extraction from pdf documents,” in The Twenty-Ninth International Flairs Conference, 2016.
3. Gilani A, Qasim SR, Malik I, Shafait F. Table detection using deep learning. In: 2017 14th IAPR international conference on document analysis and recognition (ICDAR), Vol. 1. IEEE; 2017, p. 771–6.
4. Kasar T, Barlas P, Adam S, Chatelain C, Paquet T. Learning to detect tables in scanned document images using line information. In: 2013 12th international conference on document analysis and recognition. IEEE; 2013, p. 1185–9.
5. Puha A, Rinciog O, Posea V. Enhancing open data knowledge by extracting tabular data from text images. In: DATA. 2018.
6. D. W. Embley, M. Hurst, D. Lopresti, and G. Nagy, “Table-processing paradigms: a research survey,” International Journal of Document Analysis and Recognition (IJDAR), vol. 8, no. 2-3, pp. 66–86, 2006
7. S. Schreiber, S. Agne, I. Wolf, A. Dengel, and S. Ahmed, “Deepdesrt: Deep learning for detection and structure recognition of tables in document images,” in Document Analysis and Recognition (ICDAR), 2017 14th IAPR International Conference on, vol. 1. IEEE, 2017, pp. 1162–1167.
8. I. Kavasidis, S. Palazzo, C. Spampinato, C. Pino, D. Giordano, D. Giuffrida, and P. Messina, “A saliency-based convolutional neural network for table and chart detection in digitized documents,” arXiv preprint arXiv:1804.06236, 2018.
9. D. N. Tran, T. A. Tran, A. Oh, S. H. Kim, and I. S. Na, “Table detection from document image using vertical arrangement of text blocks,” International Journal of Contents, vol. 11, no. 4, pp. 77–85, 2015.
10. T. T. Anh, N. In-Seop, and K. Soo-Hyung, “A hybrid method for table detection from document image,” in Pattern Recognition (ACPR), 2015 3rd IAPR Asian Conference on. IEEE, 2015, pp. 131–135.
11. B. Couasnon and A. Lemaitre, “Recognition of Tables and Forms,” in ¨ Handbook of Document Image Processing and Recognition, D. Doermann and K. Tombre, Eds. London: Springer London, 2014, pp. 647–677.
12. A. S. Albahri et al.: “Based Multiple Heterogeneous Wearable Sensors: A Smart Real-Time Health Monitoring Structured for Hospitals Distributor”,IEEE ,Vol. 7, 2019
13. Arsalan Mosenia, et al.: “Wearable Medical Sensor-based System Design: A Survey”, IEEE,Vol. 3, 2017.
14. Lei Ru et al.:“A Detailed Research on Human Health Monitoring System Based on Internet of Things”,Hindawi,2021
15. I Ahmad et al.: “Emerging Technologies for Next Generation Remote Health Care and Assisted Living”, IEEE, 2022.
16. D. M. Bean, H. Wu, E. Iqbal, O. Dzahini, Z. M. Ibrahim, M. Broadbent, R. Stewart, and R. J. B. Dobson, Knowledge graph prediction of unknown adverse drug reactions and validation in electronic health records, Sci. Rep., vol. 7, no. 1, p. 16416, 2017.

17. R. Sharma and V. Khanna. Automatic Generation of Health Status Graphs for Chronic Disease Management: an automated system for generating health status graphs for chronic disease management. IEEE Transactions on Automation Science and Engineering, 2017.

18. Hoe Tung Yew et al.: “IoT Based Real-Time Remote Patient Monitoring”, IEEE International Colloquium on Signal Processing & its Applications (CSPA 2020).

19. E. Lee, Y. Wang, R. Davis, and B. Egan, Designing a low-cost adaptable and personalized remote patient monitoring system. 2017.

20. Shibo Zhang et al.: “Deep Learning in Human Activity Recognition with Wearable Sensors: A Review on Advances”,Sensors,2022.

21. David W. Embley, Spencer Machado et al.: “Data Extraction from Web Tables: the Devil is in the Details”, IEEE,2011.

22. Mehmet Yasin AKPINAR et al.: “Extracting Table Data from Images Using Optical Character Recognition Text”, IEEE , 2018.

23. Yingying Yuan et al.: “Flexible Wearable Sensors in Medical Monitoring”, MDPI, 2022.

24. Henry Friday Nweke et al.: “Mobile and Wearable Sensors for Data-driven 3 Health Monitoring System: State-of-the-Art and 4 Future Prospect”, ResearchGate, 2022.

25. S Shukla et al.: “Health Care Management System Using Time Series Analysis”, IEEE, 2019.

26. WEIJIA LU et al.: “A Clinical Prediction Model in Health Time Series Data Based on Long Short-Term Memory Network Optimized by Fruit Fly Optimization Algorithm”, IEEE, 2020.

27. Jiayi Yuan et al.: “An OpenCV-based Framework for Table Information Extraction”, IEEE, 2020.