ANALYSING API DATA

This project report shall be considered for partial fulfilment of DEP Subject.

SANYAM JAIN (N229), DAKSH GEHLOT (N230), GAUTAM KUNDALIA (N243)

2022 - 2023

Research

Analysing API Data

Daksh Gehlot, Gautam Kundalia, Sanyam Jain, MBA Tech Computer Engineering, NMIMS MPSTME Shirpur

Abstract:

The use of APIs has been becoming quite common in recent times. One can see their application almost everywhere, be it an app or a website. In this research, we have analysed data retrieved from an API. The data was fetched through a Python script and was then saved into a MySQL database and into a CSV file. R was then used to run statistical analysis on the retrieved data.

1. Introduction

An API, which stands for Application Programming Interface, is a software intermediary which allows two applications to communicate with each other. Each time you use an app like Instagram, send an instant message, or check the weather on your phone, you're using an API.

To simplify, an API delivers a user response to a system and sends the system's response back to a user. It enables this process to be completed efficiently and without any huge delays. Modern APIs adhere to standards (like HTTP) that are developer friendly, easily accessible and broadly understood, and are used by major companies like Google, eBay, Amazon and Expedia.

2. Problem Statement

Predicting the climatical analysis of a particular place.

3. Expected Solution

Knowing the weather conditions of a particular place

4. Methodology

Using the platform R Studio to get datasets, create our own, do data extraction, data cleaning, and data visualisation tasks.

Using API to retrieve data on various climatic constraints and filtering that data using python

5. Tools Used

RStudio; an integrated development environment for R, a programming language for statistical computing and graphics.

Visual Studio Code, also commonly referred to as VS Code, is a source-code editor made by Microsoft with the Electron Framework, for Windows, Linux and macOS.

6. Working

In our program, we used two APIs, one is called the Open Weather Map API and the other is the Geocoder API. The first is used to identify the geo coordinates of a specific location that the user has specified, and the other to receive details about the local weather. The data is received as nested lists and dictionaries. We now extract the data and provide it to the user as seen in Figure I.

As demonstrated in Figure II, we have also utilised libraries like numpy to display the graphical representation between the various qualities throughout the period of days.

The incoming data is saved as a CSV file, from which it is later possible to execute data analysis using R programming language. The obtained result is shown in figure III.

7. Data Analysis with R

We have prepared a CSV file called "weather data.csv" for the data analysis section; this file

comprises information such as the current temperature, humidity, time, date, and location. We

have conducted some statistical analysis using the data that the API has returned. For example, calculating the average temperature and humidity across all the different cities and displaying a visual that shows the current temperature throughout different cities in India.

```
library(ggplot2)
DEP <- read.csv("weather_data.csv", header=TRUE)
print(DEP)
print("Average Temperature:")
sum(DEP$Current_Temp)/9
print("Average Humidity:")
sum(DEP$Humidity)/9
Cities <- c("Jodhpur", "Shirpur", "Mumbai", "Indore", "Ujjain", "Delhi", "Udaipur"
            ,"Agra", "Hyderabad")

IrisPlot <- ggplot(DEP, aes(City, Current_Temp)) + geom_point()
plot(IrisPlot, type='o')

plot(DEP$Current_Temp, type='o', col="blue", xlab="Cities", ylab="Temperature & Humidity",
     main="")
lines(DEP$Humidity, type='o', col="red")
```

Fig 1

```
> setwd("D:/NMIMS PS Material/Stock_Market_Analysis")
> library(ggplot2)
> DEP <- read.csv("weather_data.csv", header=TRUE)
> print(DEP)
      Date Current_Temp Feels_Like Humidity  Time      City
1 12-10-22        34.68      32.59       19 14:25   Jodhpur
2 12-10-22        32.77      34.30       44 14:26   Shirpur
3 12-10-22        26.99      31.13       94 14:35    Mumbai
4 12-10-22        32.10      32.99       43 14:45    Indore
5 12-10-22        29.50      29.45       43 14:46    Ujjain
6 12-10-22        29.05      30.81       58 14:47     Delhi
7 12-10-22        31.50      29.90       26 14:48   Udaipur
8 12-10-22        31.67      31.65       39 14:49      Agra
9 12-10-22        30.23      32.86       58 14:50 Hyderabad
```

Fig 2

```
> print("Average Temperature:")
[1] "Average Temperature:"
> sum(DEP$Current_Temp)/9
[1] 30.94333
> print("Average Humidity:")
[1] "Average Humidity:"
> sum(DEP$Humidity)/9
[1] 47.11111
> Cities <- c("Jodhpur", "Shirpur", "Mumbai", "Indore", "Ujjain", "Delhi", "Udaipur"
+             ,"Agra", "Hyderabad")
>
> IrisPlot <- ggplot(DEP, aes(City, Current_Temp)) + geom_point()
> plot(IrisPlot, type='o')
>
```

Fig 3

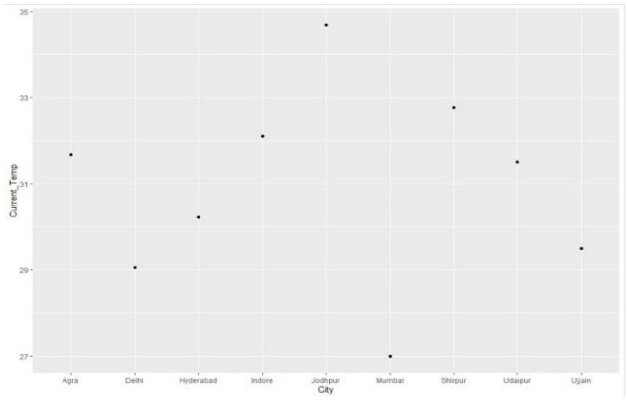

Fig 4

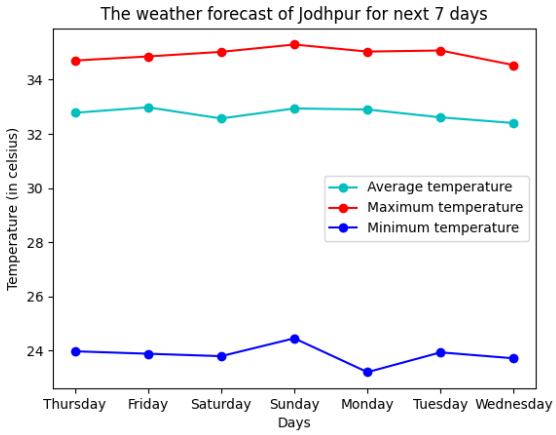

Fig 5

8. Conclusion

This paper discussed the working and implementation of "Open Weather Map" API. A brief introduction about this API was given, and also about APIs in general and its application was presented. This paper also explained about the use of programming languages like Python and R for Data Analysis and representation. The result showed the various climatic constraints of different cities and also graphs for better representation

9. References

1) Weather API via:
https://openweathermap.org/api/

2) API used for coordinates via:
https://opencagedata.com/

3) Other Useful free websites that helped us overcome the errors:

https://www.mulesoft.com
https://www.infoworld.com
https://pypi.org
https://www.w3schools.com
https://www.geeksforgeeks.org
https://realpython.com
https://stackoverflow.com

4) The source code for our project can be found at:
https://github.com/dakshgehlot/DEP-Project-SEM-3