Geospatial Clustering and Forecasting for Global Hotspots

A Project Report
Presented to
The Faculty of the College of
Engineering

San Jose State University
In Partial Fulfillment
Of the Requirements for the Degree
Master of Science in Computer Engineering
Master of Science in Software Engineering

By

Vamsi Krishna Chakravartula
Harshada Baswaraj Jivane
Lakshmi Vihita Kesiraju
Sum Mohan Reddy Mallannagari

May/2021

APPROVED

Dan Harkey, Project Advisor

David Bruck, Director, MS Computer Engineering

Dan Harkey, Director, MS Software Engineering

Rod Fatoohi, Department Chair

ABSTRACT

Geospatial Clustering and Forecasting for Global Hotspots

By

Chakravartula Vamsi Krishna, Jivane Harshada Baswaraj, Kesiraju Lakshmi Vihita, Mallannagari Sum Mohan Reddy

COVID-19 has made a global disruption in every aspect of our social fabric. The pandemic has taken a toll on the healthcare system, increasing the need for acute care that has overtaxed some hospitals. The lack of public health information systems impacted the balancing and distribution of the critical resources from areas of surplus to undersupply.

The shortage of resources and the incremental uncertainty in the incoming patient count does not allow for a stable servicing of the hospitals. On the other hand, social media accounts for an excess of information, including vital updates on the pandemic and false data. The incorrect details mislead the public into a state of turmoil, which calls for a single source of information to address these problems.

A predictive analysis-based web application that can support the hospitals with the surges in patient inflow, assist the public users in finding treatment and predict the hotspots based on the existing and historical admittance and infections data would be one of the project's goals. Clustering hotspot regions based on the environmental, COVID-19 cases similarities, and accurate social media data would help identify the patterns for applying control policies. A comprehensive suite of technologies along with cutting edge methodologies of clustering and forecasting algorithms will be implemented on Google Cloud Platform.

Acknowledgments

The authors are deeply indebted to Professor Dan Harkey for his constructive feedback and support in the preparation of this study.

Table of Contents

List of Figures

List of Tables

Chapter 1. Project Overview

1.1 Introduction to Covid-19 Analysis

Advancements in the technologies and analytical methods are key aspects in acknowledging the spatial spread of COVID-19 disease pandemic that include interactive web-based maps and dashboards for quick understanding of reasons for illness. Lack of access to better resources in few areas has been a major drawback in identifying the cases to provide treatment on time. According to the census provided by Johns Hopkins University along with ESRI, Redlands, California, with details on the number of cases, deaths, and recoveries was useful to track the cases globally, allocate resources, and to figure out the possible preventions.

This web-based application would monitor the patient inflow, availability in nearest hospitals to visualize the information of resources in order to provide improved services to the COVID patients. The inadequate knowledge on the virus is one of the key reasons for uncontrollable spread of the virus. Suggesting the nearest hospital with the equipment information could be helpful in admitting the patients in the right time for proper assistance. Multiple media sources have influenced the spread of rumors about the disease. Accuracy in the news plays a vital role in social, economic and health consequences.

1.2 Proposed Areas of Study and Academic Contribution

COVID-19 has accounted for a massive global disruption. There have been numerous reports that the impact of the ongoing COVID-19 pandemic has disproportionately impacted the global communities .The goal of the project is to examine the socio-cultural and topographic nature of spatial hot spots of SARS-CoV-2 rates across the United States. In a non-Pharmaceutical way of intervention control policies such as physical distancing or social distancing, self-isolation play a key role. The existing health disparities are likely to be rapidly magnified in the context of COVID-19, and potentially extend well beyond the lifespan of the pandemic.[1]

Incorporating geographic information science and technology (GIS&T) into COVID-19 pandemic surveillance, modeling, and response enhances understanding and control of the disease. An application of the GIS&T is integrating geographic data in COVID-19 modeling and communicating the status of the disease or status of facilities for efficient functioning. Locations and availability of personal protective equipment, ventilators, hospital beds, and other items can be optimized with the use of GIS&T. [2] Geospatial clustering and predictive analysis techniques are used to achieve these objectives.

Classical clustering algorithms (e.g k-mean) are not designed for grouping spatial data. Spatial clustering is one of task mining for grouping objects spatial and extend of classical clustering. In the research, spatial algorithm used is CLARANS (Clustering Large Applications based on Randomized Search), with the reasoning that *k* partitioned clusters are better than other approaches and CLARANS was designed for big data with

efficiency (computation complexity or time) and effectiveness (average distortion over the distances)[3].

Different clustering techniques for spatial data mining have certain advantages and disadvantages [4]. Depending on the need of the application, a specific technique can be selected. Another correlation has been established[5], drawing a relation between Spatial Clusters Susceptible and Epidemic Outbreaks due to Under vaccination. The method described for finding critical sets, applied to detailed population and contact network models, provides an operational tool for public health agencies to prioritize limited surveillance and outreach resources towards the most critical clusters. A Visual Multi-Scale Spatial Clustering approach is using Graph theory tools that have natural features proved to be suitable for defining the spatial structure, especially spatial neighborhood relationships. [6]

Normalized mean squared error (NMSE) and mean absolute percentage error (MAPE) are utilized to assess the accuracies of Autoregressive Integrated Moving Average Model (ARIMA), Long Short-Term Memory model (LSTM) and Random Forests (RF). The findings indicate that RF, the only multivariate model among the three models, performs best, and the results can be used to aid in strategic decision-making on inpatient beds resource planning in response to predictable discharges[7]. Automatic predictive analytics framework (PRAF) for geospatial human geographic data consists of a feature selection procedure and a predictor based on a neural network, handling problems that fit into a Big Data environment[8].

Spatial-Temporal Density-Based Spatial Clustering of Applications with Noise (ST-DBSCAN) is used to spatially-temporally cluster the tweets, summarize word frequencies for each cluster and model the potential topics by the Latent Dirichlet Allocation (LDA) algorithm.[9].

1.3 Current State of the Art

Unsupervised machine learning can be very powerful in its own right, and clustering is by far the most common expression of this group of problems. Python language has evolved to support machine learning applications. Various Python libraries are available to use as per the need of the application such as sklearn. Algorithms such as CLARANS, DBSCAN and BIRCH have been widely used to address the problems of geo-spatial clustering [10].

With the clusters as a base, predictive analytic models forecast metric values, estimating a numeric value for new data based on learnings from historical data. A cutting-edge forecasting tool by Facebook, Prophet produces high-quality forecasts using an additive regression model that is robust to missing data and shifts in the trend, and typically handles outliers[11][12].

Airflow[13] is an end-to-end platform for data science and machine learning where you can build and deploy models quickly and manage your ML workflows at scale. AWS[14], Microsoft Azure[15], and Google Cloud Platform[16] offer many options for implementing machine learning applications on the cloud with the best in class features

available to the users. Optimized storage solutions that can scale up with the vast datasets come along with the computational counterparts of the cloud providers.

PySpark is widely used for performing exploratory data analysis at scale, building machine learning pipelines, and creating ETLs for a data platform.It is a convenient language to learn especially for someone who already is familiar with Python and libraries such as Pandas.[17]

Chapter 2. Project Architecture

The architecture is a mechanism, which helps to determine how each component works together. Below is the architecture, that contains four blocks and the process of the interaction between the blocks, layers, and components that are required.

2.1 Introduction

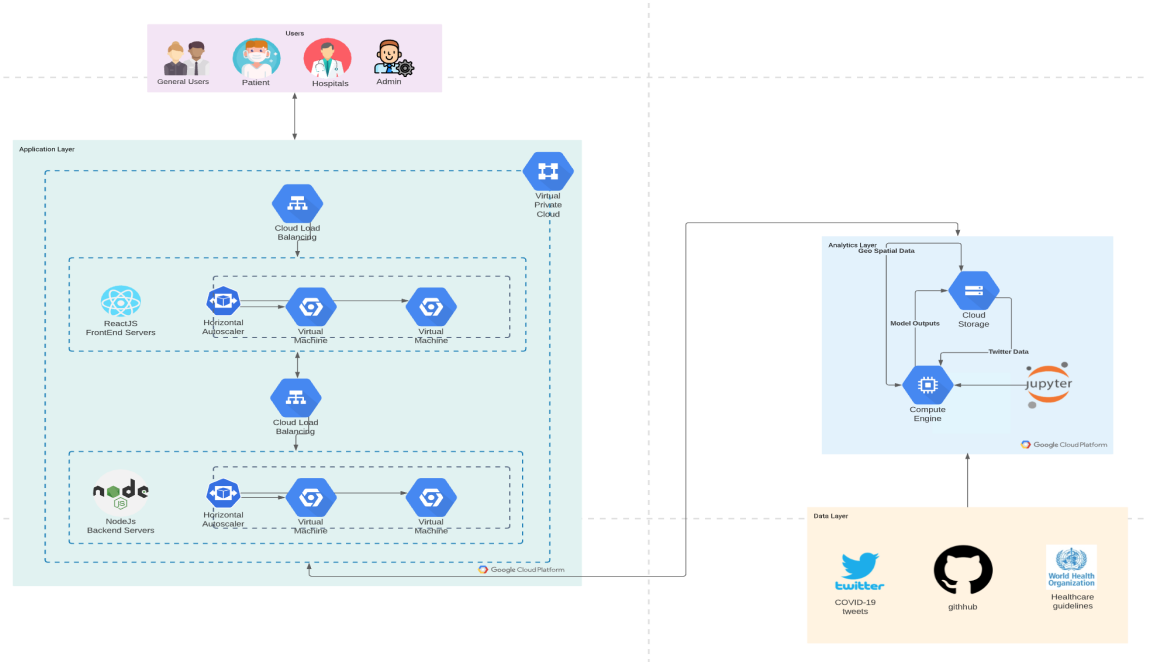


Figure 2.1.1 Architecture Diagram

2. 2 Architecture Subsystems

2.2.1 *Users*

The users will be of a varied category covering the general users, Hospital personnel, patients and application admins.

2.2.2 *Application Layer*

The application layer can be subdivided into 2 components that contain Frontend and Backend application servers. Each set of servers are governed by an auto-scaling system that can increase or decrease the number of active servers that are running in accordance with the incoming load. The traffic is directed to these auto-scaled servers through a load balancer that sits on top of each set, ensuring a stable load to each server. The backend server picks up data from Cloud Storage, processes it and sends it to the frontend servers. The web pages render the information provided by the backend.

Cloud providers terminologies:

	AWS	**Azure**	**GCP**
Compute	Amazon EC2	Azure Virtual Machines	Google Compute Engine
File Storage	Amazon S3	Azure Blob Storage	Google Storage
NoSQL	Amazon DynamoDB	Azure DocumentDB	Google Cloud Datastore
Function as a Service	Amazon Lambda	Azure Functions	Google Cloud Functions
Relational Database	Amazon RDS	Azure SQL Database	Google Cloud SQL
Container Scheduler	Amazon EC2 Container Service	Azure Container Service	Google Kubernetes Engine
App Deployment	Amazon Elastic Beanstalk	Azure Cloud Services	Google App Engine
Data Warehouse	Amazon Redshift	Azure SQL Data Warehouse	Google BigQuery

2.2.2.1 Cloud Terminologies

2.2.3 *Pub/Sub Layer*

The Pub-Sub layer will be serving as a message broker, improving the request handling capacities while ensuring better fault tolerance.

2.2.4 *Analytics Layer*

The analytics layer consists of the machine learning core which is the base of the complete application. It consists of multiple technological solutions that serve different purposes. The geospatial data is handled by the BigQuery, while the Cloud datastore handles the Tweets storage. The processing is done in the Compute Engine that uses Jupyter notebook to analyze the data and provide outputs. The results are then sent to the Cloud Storage component.

2.2.5 *Data Layer*

The data is sourced from multiple sources which include JHU's Github, Twitter feed data from APIs, Policies and Hospital data provided by the official government websites.

Chapter 3. Technology Descriptions

3.1 Client Technologies

3.1.1 React.js

React is used as the primary JavaScript library to build the UI, owing to the advantages of component-based structure and robust performance. The components structuring encourages code reusability and the concept of virtual DOM ensures responsiveness of the webpages.

Cascading Style Sheets (CSS) has been primarily used along with Bootstrap to style the components and the web pages altogether. The user interfaces have been kept minimalistic as to focus more on the functionalities without losing the aesthetics.

Ref: https://reactjs.org/

3.1.2 MapGL

Mapbox is an industry standard while coming to navigation and geospatial data representation using Maps. With two of the use cases requiring a map display of the data, Mapbox was chosen to showcase the geospatial clusters and the nearest hospital displays. While integrating Mapbox in React is not directly feasible, a wrapper provided by Vis.gl project is employed for including the maps in the application.

Ref: https://www.mapbox.com/

Ref: https://visgl.github.io/react-map-gl/

3.1.2 Chart.js

Line graphs are used to display the predictions of the hospital inflow, which have been implemented using the Chart.js. It is an open source JavaScript charting library that provides various graphs which are responsive and user friendly.

Ref : https://www.chartjs.org/

3.2 Middle-Tier Technologies

3.2.1 Node.Js

The backend APIs were designed using Node.js. It is a non-blocking, single threaded Javascript framework, processing requests in an asynchronous fashion based on the event-driven model. Another advantage of Node.js is the seamless JSON support that avoids unnecessary binary format conversions. With the Frontend also written in Javascript, both these features add to a fast ecosystem that can be expanded with ease. Our APIs are primarily used to read JSON data and send the processed outputs.

Ref: https://nodejs.org/en/

3.2.2 Express.Js

Express.js is a web-based backend framework that supports node.js to provide strong HTTP servers for websites, tools and APIs. This can be used for single page, multi-page or hybrid server-based web applications. Routes are used to navigate within the

application depending on the request. Few templates are available in default to increase the efficiency of the framework.

Ref: https://expressjs.com/

3.2.4 Jupyter

Jupyter Notebook is an open-source web application system. It is an IDE that allows creating, editing, and sharing of documents that contain visualizations, live code, equations, and text for programming languages (Python). It is used for data cleaning, preprocessing, modeling, and visualization techniques.

3.3 Data-Tier Technologies

3.3.1 Cloud Storage

Google Cloud Storage is a file storage service for storing and accessing data on Google Cloud Platform. It is typically used to store unstructured data. Objects of any kind and size, and up to 5 TB could be stored.

Cloud Storage has many locations where we can store our data with redundancy options.

Ref: https://cloud.google.com/storage

3.4 Infrastructure

3.4.1 VPC

Virtual Private Cloud Controls the ability to mitigate the risk of data movement from Google Cloud services such as BigQuery and Cloud Storage. We basically create perimeters that protect the resources and data of services that we explicitly specify. Resources within a perimeter can be accessed only by the clients within authorized VPC networks.

3.4.2 Load balancing and Autoscaling

Cloud Load Balancing helps to scale applications and distribute compute resources in single/multiple regions close to users and meet application's high availability requirements. Load balancing logs all the requests received by it and these logs can be used for analyzing your user traffic as well as debugging.

Ref: https://cloud.google.com/load-balancing

3.4.3 Compute Engines

Google Compute Engine provides a scalable number of virtual machines to implement compute clusters. The virtual machines can be managed through command line interface (CLI), RESTful API, or Web console. They are secure and customizable as per the application's needs such as the number of CPU and memory size.

Chapter 4. Project Design

4.1 Client Design

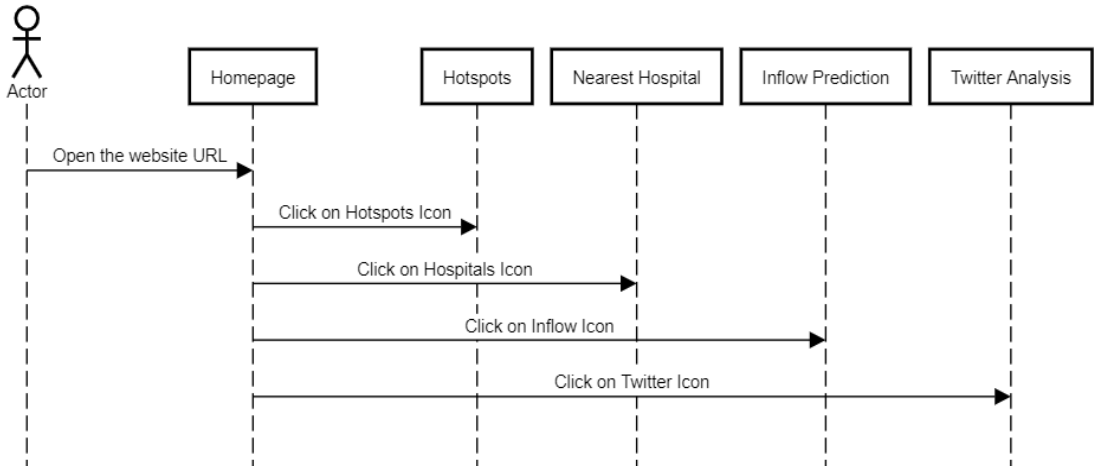


Figure 4.1.1 Block diagram for Client

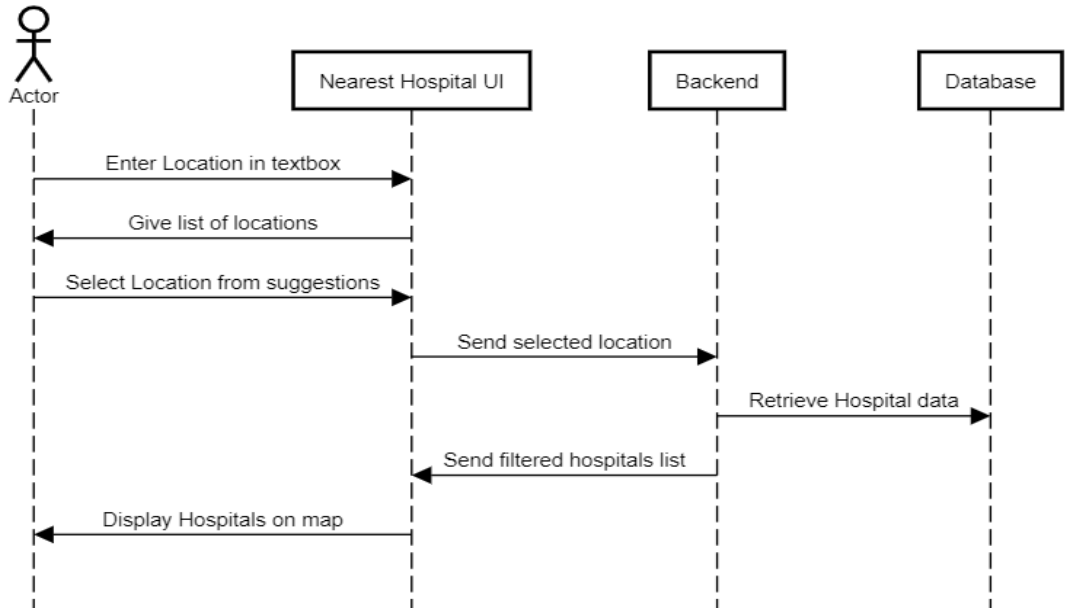


Figure 4.1.2 Nearest Hospital

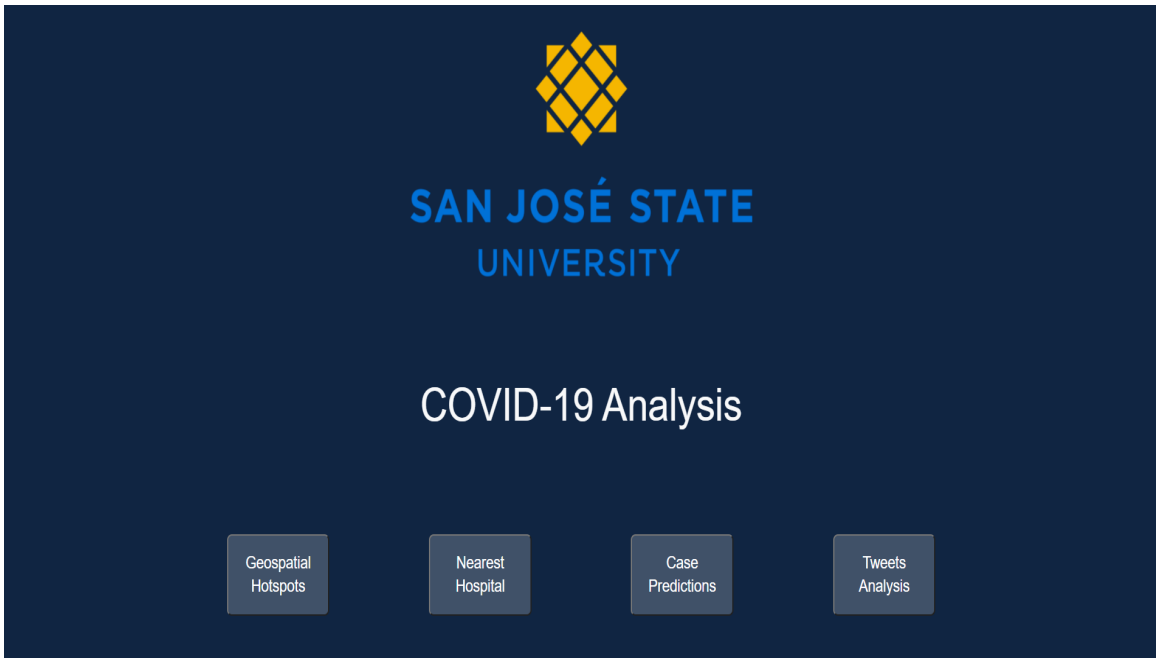


Figure 4.1.3 Web Application Design

4.2 Middle-Tier Design

The backend servers are meant to handle requests from the frontend and all the requests hit pre-specified API endpoints. The APIs can logically be structured feature wise. The processing logic is written in the services while the channeling of the requests to the correct is done in the Routes. The routes can be given in a high level as:

- /api/geospatial
- /api/nearestHospitals
- /api/inflow
- /api/tweetsPaginated

4.2.1 API Design

Component	**Endpoint**	**Sample Request**	**Sample Response**
Geospatial Clusters	/geospatial GET	{}	{ "type": "Feature", "geometry": { "type": "Point", "coordinates": [-75.39031349, 38.66143781] }, "properties": { "Country": "US", "Lat": 38.66143781, "Long": -75.39031349, "CumConfirmed": 0, "CumDeaths": 0, "DaysTo10X": 6.0, "EarlyMortality": 1.0028239087, "PeakMortality": 0.0180255571, "EarlyAccel": 3.2987406471, "CurrentAccel": 1.3703081353,

			"cluster": 4 } },

/nearestHospital	/nearestHospitals
	GET
		{
		"latitude":"37.7798721",
		"longitude":"-122.2821855"
		}
			{
			"NAME": "CENTRAL VALLEY GENERAL HOSPITAL",
			"ADDRESS": "1025 NORTH DOUTY STREET",
			"CITY": "HANFORD",
			"STATE": "CA",
			"ZIP": 93230,
			"ZIP4": "NOT AVAILABLE",
			"TELEPHONE": "NOT AVAILABLE",
			"TYPE": "GENERAL ACUTE CARE",
			"STATUS": "CLOSED",
			"POPULATION": 49,
			"COUNTY": "KINGS",
			"COUNTYFIPS": "06031",

			"COUNTRY": "USA", "LATITUDE": 36.33615885, "LONGITUDE": -119.645667298 },

Inflow Prediction	/inflow
	GET
		{
		"hospitalID":"Los Angeles,California"
		}
			{
			dates: [
			'2021-03-25', '2021-03-26',
			'2021-03-27', '2021-03-28',
			'2021-03-29', '2021-03-30',
			'2021-03-31', '2021-04-01',
			'2021-04-02', '2021-04-03'
			],
			deaths: [
			23020, 23055, 23080,
			23103, 23101, 23110,
			23143, 23189, 23235,
			23274
			],
			cases: [
			1216904, 1217686,

			1218359, 1218879, 1219237, 1219612, 1220217, 1220900, 1221506, 1222262] }

Twitter Analysis	/tweetsPaginated GET	{
		{
		"page":"1",
		"limit":"3"
		}
			{
			"Tweets": "According to available studies, #hydroxychloroquine does not have clinical benefits in treating #COVID19. - @WHO https://t.co/n1JDUNs6Gu",
			"Facts": "Studies show hydroxychloroquine does not have clinical benefits in treating COVID-19"
			},
			{
			"Tweets": "Do you know the symptoms of

			#COVID19?\n\nhttps://t.co/EkSdIKiuOO", "Facts": "How can you recognize the symptoms of COVID-19?" }, { "Tweets": "@aconsumingfire @EWErickson #Hydroxychloroquine (+/- #Azithromycin) is NOT effective treatment for #COVID19. Clinic\u2026 https://t.co/T1nWT5N09m", "Facts": "Studies show hydroxychloroquine does not have clinical benefits in treating COVID-19" },

4.2.1.1 API Design

4.3 Data-Tier Design

The following are the attributes for clustering hotspot and prediction process -

Province_State - The name of the state within the USA.

Country_Region - The name of the County (US).

Last_Update - The most recent date the file was changed.

Lat - Latitude.

Long - Longitude.

Confirmed - Aggregate case count for the state.

Deaths - Aggregate death toll for the state.

Recovered - Aggregate recovered case count for the state.

Active - Aggregated confirmed cases that have not been resolved. (Active cases = Total cases - total recovered - total deaths).

Mortality_Rate - Recorded deaths *100 / confirmed cases.

UID - Unique Identifier for each row entry.

ISO3 - Officially assigned county code identifiers.

The following are the attributes for Predictive analysis of Patient Inflow -

date - The date for which the data is recorded for.

county - The county for which the data is recorded for.

state - The state for which the data is recorded for.

fips - The FIPS code of a particular county.

cases - Number of cases recorded for the county.

deaths - Number of deaths recorded for the county.

confirmed_cases - Number of confirmed cases from the recorded data.

confirmed_deaths - Number of confirmed deaths from the recorded data.

probable_cases - Number of cases suspected in the county.

probable_deaths - Number of deaths suspected in the county.

collection_week - the start date of the week for which the data is recorded

total_beds_7_day_avg - Total number of beds in the hospitals of a county of a week

previous_day_admission_adult_covid_confirmed_7_day_sum - Previous day sum of confirmed cases in adults for 7 days

previous_day_admission_adult_covid_confirmed_18-19_7_day_sum - Previous day sum of confirmed cases for the age group of 18-19

previous_day_admission_adult_covid_confirmed_20-29_7_day_sum - Previous day sum of confirmed cases for the age group of 20-29

previous_day_admission_adult_covid_confirmed_30-39_7_day_sum - Previous day sum of confirmed cases for the age group of 30-39

previous_day_admission_adult_covid_confirmed_40-49_7_day_sum - Previous day sum of confirmed cases for the age group of 40-49

'previous_day_admission_adult_covid_confirmed_50-59_7_day_sum - Previous day sum of confirmed cases for the age group of 50-59

previous_day_admission_adult_covid_confirmed_60-69_7_day_sum - Previous day sum of confirmed cases for the age group of 60-69

previous_day_admission_adult_covid_confirmed_70-79_7_day_sum - Previous day sum of confirmed cases for the age group of 70-79

previous_day_admission_adult_covid_confirmed_80+_7_day_sum - Previous day sum of confirmed cases for the age group of 80+

previous_day_admission_adult_covid_confirmed_unknown_7_day_sum - Previous day sum of confirmed cases for the unknown age group.

The following are the attributes for the Twitter analysis -

user_name - The name of the user who tweeted in the data.

user_location - The user location from where he tweeted.

user_description - The description of the user provided in the user's account.

user_created - The date when the user created the account.

user_followers - Number of followers the user has.

user_friends - Number of friends the user has.

user_favourites - Number of users liked his posts.

user_verified - Verification of the user’s account.

date - The date on which the user tweeted.

text - The tweet the user posted on account.

hashtags - The user tweeting about a particular topic or thing.

source - From where the data is extracted from.

is_retweet - If the tweet has any retweets.

The following are the attributes of facts data from WHO.

Organization - It says from which organization the fact statement is from.

Facts - The fact statement is given.

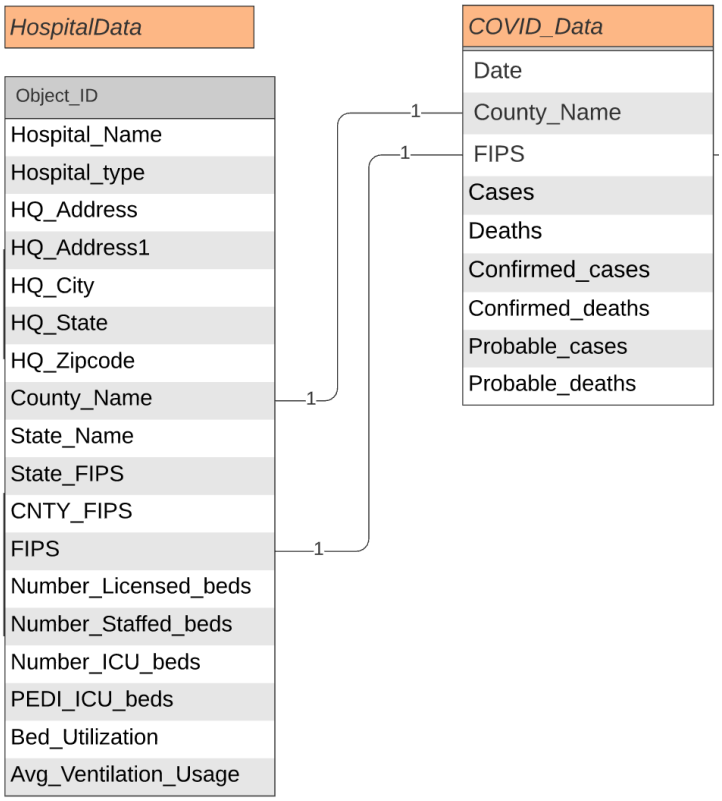


Figure 4.3.1 Database Entity diagrams

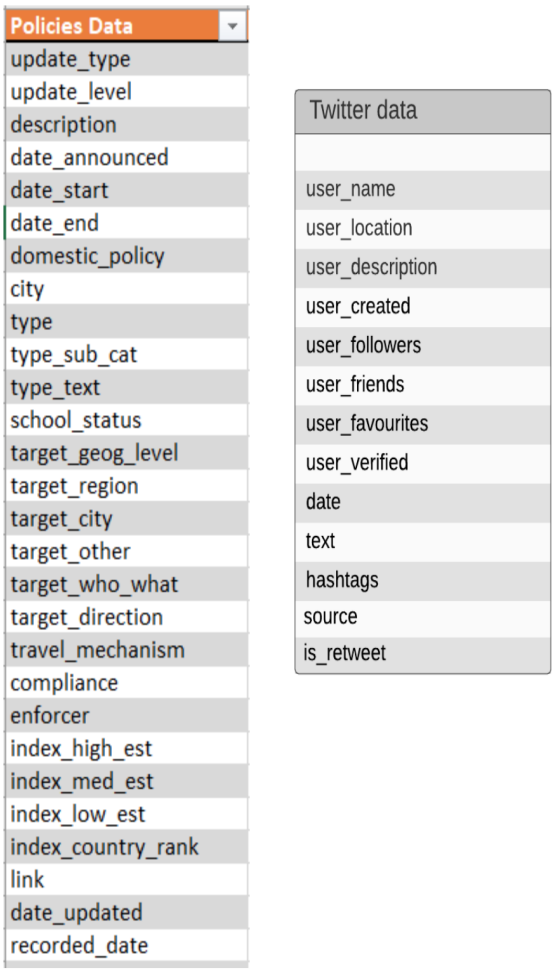


Figure 4.3.2 COVID-19

Chapter 5. Project Implementation

5.1 Geospatial Clusters

The web application should be able to display the USA map locating the hotspots geographically. The clustering of data points will help in visualizing these hotspots.

5.1.1 Population Data

To calculate the intensity of hotspot, we needed to know the population to cases ratio. Hence, we collected the global population data and added calculated features to the dataset which were then used in modelling. For instance, one of the features calculates how many days it took for the cases to reach 10 times the current cases.

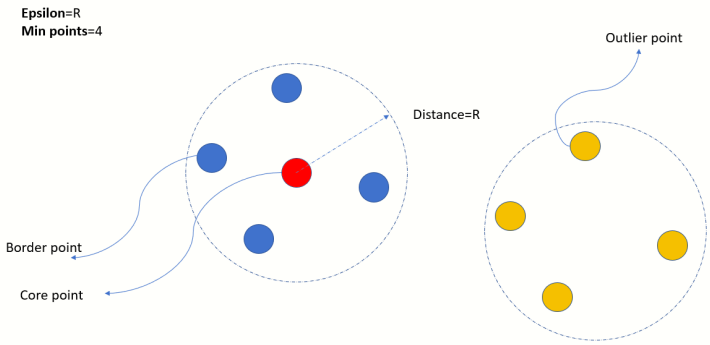
5.1.2 DBSCAN

Density-based spatial clustering of applications with noise is a density-based clustering which means it will group together data points that have many nearby neighbors. Each cluster will be a group of close neighbors. Unlike K-means clustering, DBSCAN doesn't need to know the number of clusters beforehand. This property of DBSCAN helped our application since we wouldn't know the number of hotspots beforehand.

1.**EPS:** Defines neighborhood area between data points

2.**MinPts**: Specifies the minimum number of data points required to form a neighborhood.

DBSCAN is very sensitive to its two parameters and these parameters influence each other in the result. They are hard to set up because they depend largely on the particular phenomena we are studying, and which type of clusters we want to detect. DBSCAN is more flexible when it comes to the size and shape of clusters than other partitioning methods, such as K-means. It is able to identify clusters that differ in size and shape from one another, which makes it more useful for messy, real life data.



5.1.2.1 DBSCAN

```
DBSCAN(D, eps, MinPts)
  C = 0
  for each unvisited point P in dataset D
    mark P as visited
    N = getNeighbors (P, eps)
    if sizeof(N) < MinPts
      mark P as NOISE
    else
      C = next cluster
      expandCluster(P, N, C, eps, MinPts)

expandCluster(P, N, C, eps, MinPts)
  add P to cluster C
  for each point P' in N
    if P' is not visited
      mark P' as visited
      N' = getNeighbors(P', eps)
      if sizeof(N') >= MinPts
        N = N joined with N'
    if P' is not yet member of any cluster
      add P' to cluster C
```

5.1.3 Libraries

plotly

This library helps in creating interactive graphs. It supports many types of plots like line charts, bar charts, bubble charts. They helped us to visualize the data and understand the patterns.

sklearn

Scikit-learn provides a variety of supervised as well as unsupervised learning algorithms. We utilized the DBSCAN clustering module from this library. We can adjust the parameters based on the need of application and structure of data.

eps : *float, default=0.5*

The maximum distance between two samples for one to be considered as in the neighborhood of the other. This is not a maximum bound on the distances of points within a cluster. This is the most important DBSCAN parameter to choose appropriately for your data set and distance function.

min_samples : *int, default=5*

The number of samples (or total weight) in a neighborhood for a point to be considered as a core point. This includes the point itself.

metric : *string, or callable, default='euclidean'*

The metric to use when calculating distance between instances in a feature array. If metric is a string or callable, it must be one of the options allowed by `sklearn.metrics.pairwise_distances` for its metric parameter. If metric is "precomputed", X is assumed to be a distance matrix and must be square. X may be a Glossary, in which case only "nonzero" elements may be considered neighbors for DBSCAN.

5.1.4 User Interface

Users can view the hotspots after opening the Hotspot tab on the web application.

On hovering over the cluster, user can read details about the cluster

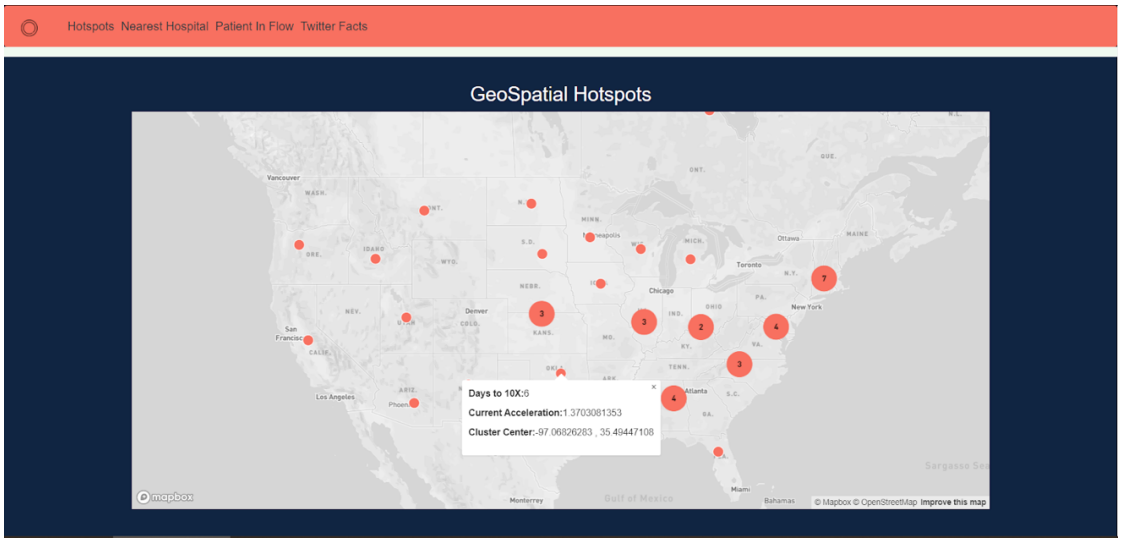


Figure 5.1.4.1 Geo Spatial Hotspots

5.2 Nearest Hospital

5.2.1 Haversine Formula

This feature will be implemented using a mathematical approach to find the nearest hospital. Haversine's formula finds out the distance between two points on a sphere by using their latitude and longitude values. The approximation of using a sphere can be neglected in comparison to using a more explicit elliptical formula as the distances are quite small and do not have a significant effect of the curvature. A simple representation of the formula and its usage to calculate the distance, is as follows:

$$
\begin{aligned}
d &= 2r \arcsin\left(\sqrt{\operatorname{hav}(\varphi_2 - \varphi_1) + \cos(\varphi_1)\cos(\varphi_2)\operatorname{hav}(\lambda_2 - \lambda_1)}\right) \\
&= 2r \arcsin\left(\sqrt{\sin^2\left(\frac{\varphi_2 - \varphi_1}{2}\right) + \cos(\varphi_1)\cos(\varphi_2)\sin^2\left(\frac{\lambda_2 - \lambda_1}{2}\right)}\right)
\end{aligned}
$$

Figure 5.2.1.1 Haversine Formula

Where

- φ_1, φ_2 are the latitude of point 1 and latitude of point 2 (in radians),
- λ_1, λ_2 are the longitude of point 1 and longitude of point 2 (in radians).

5.2.2 geodist package

The ‘geodist’ package from npm is a simple to use distance calculator between a set of geographical points represented by the latitudes and longitudes. The calculations are done using Haversine's formula.

The input parameters used here are the latitude and longitude of the given location and the ones of the hospitals in the dataset. Both the parameters need to be in a JSON format, represented as a key-value pair. The keys need to be exactly as “lat” and “lon” as per the implementation.

Usage code snippet:

```
var geodist = require('geodist')

var dist = geodist({lat: 41.85, lon: -87.65}, {lat: 33.7489, lon: -84.3881})
console.log(dist)
// => 587
```

Ref: https://github.com/cmoncrief/geodist#readme

5.2.3 User Interface

The user has access to an autocomplete text box which provides suggestions using text matching with the geographical locations. As the text is entered, a dropdown of locations is displayed, one of which is selected. After selecting the location, the user hits the Submit button. This initiates a request from the user to the backend, which processes the

hospital data set to filter out locations under a 15-mile radius. The resultant list is received back by the browser which displays them in the map.

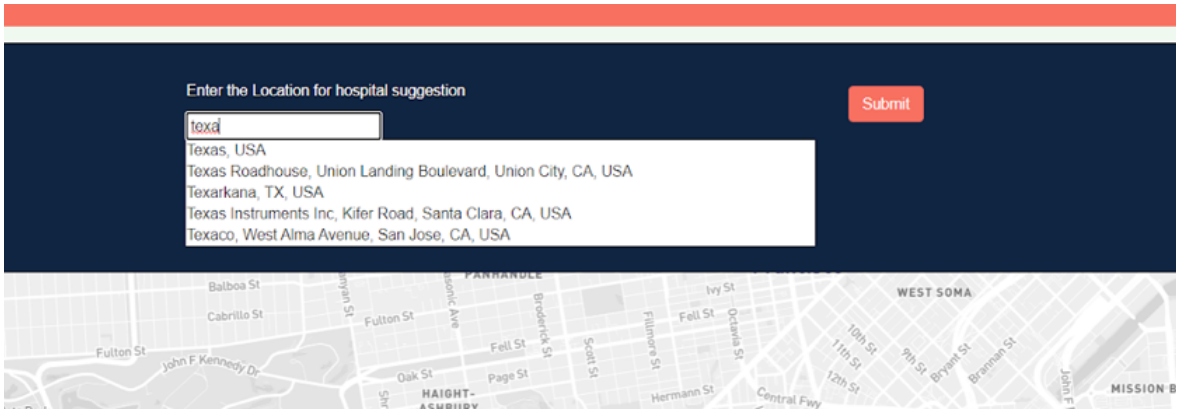


Figure 5.2.3.1 Location search

.

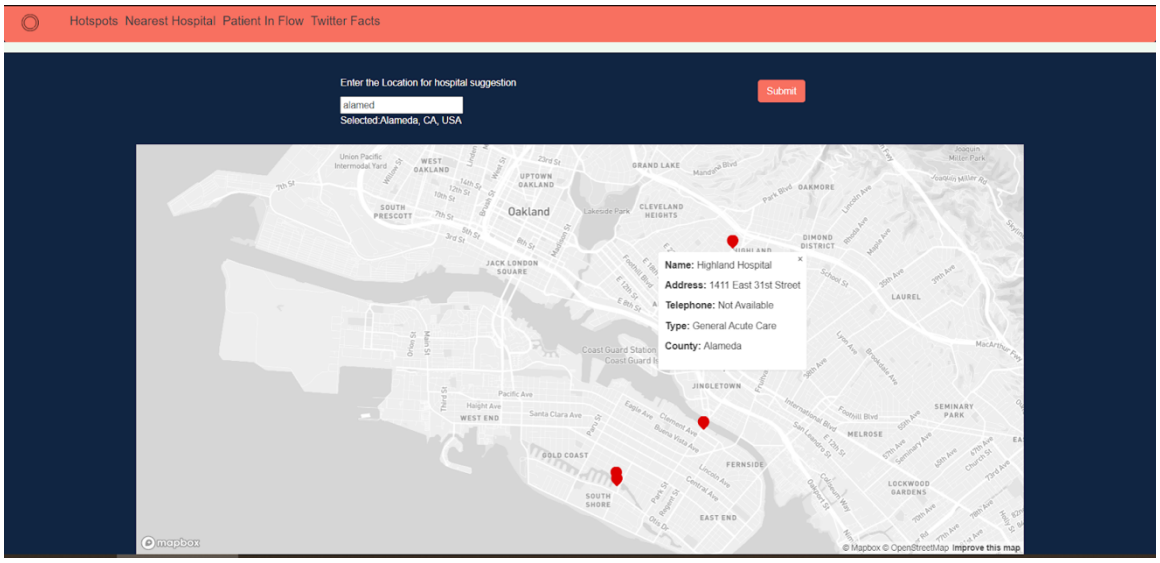


Figure 5.2.3.2 Showing Nearest Hospital

5.2.4 Data Flow

The dataset of hospitals is read as a JSON, which contains the latitude, longitude and the hospital details for each entry. This dataset is filtered with the input location coordinates received from the user. The geographical distance is calculated for each hospital using the geodist method. The resultant shortlisted hospitals that fall within a fixed threshold of distance, will be returned as a JSON. These filtered locations are displayed as markers on the map in the user interface. When clicked on a marker, it pops up a textbox with the hospital details.

```
await jsonData.map((hosp) => {
    // console.log(Math.abs(parseInt(req.query.latitude) - hosp.LATITUDE.toFixed(2)))
    let hospPoint = {
        lat: hosp.LATITUDE,
        lon: hosp.LONGITUDE
    }
    let distance = geodist(currPoint, hospPoint);
    if (distance < 15) {
        outJson.push(hosp);
        // console.log(hosp)
    }

})
```

Figure 5.2.4.1 Code snippet for geodist implementation.

5.3 Hospital Inflow Prediction

5.3.1 Linear Regression

Linear Regression is a well-used machine learning model to find the relationship between prediction variables and the target variable. A statistical relationship can be inferred

between the dependent and independent variables that best fits the data. A relational line is drawn between the variables to describe the correspondence.

The expression can be represented as $Y=a+bX,$

where Y is the dependent variable, X is the independent variable.

The intercept is a and b is the slope of the line

5.3.2 User Interface

The user can see the graph for the predicted cases based on the county selected from the dropdown as well as the covid data for the respective dates.

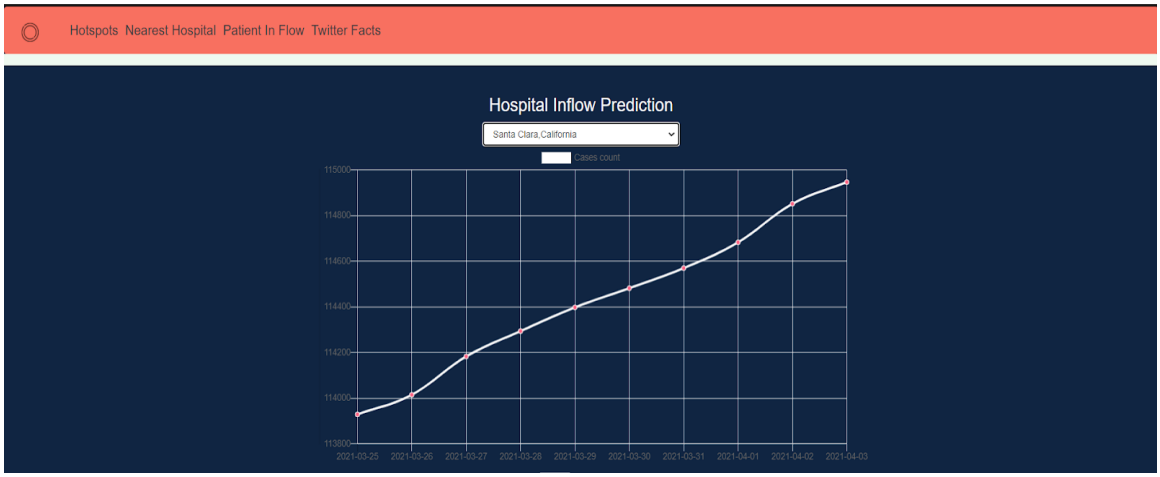


Figure 5.3.2.1 Case Count for COVID-19

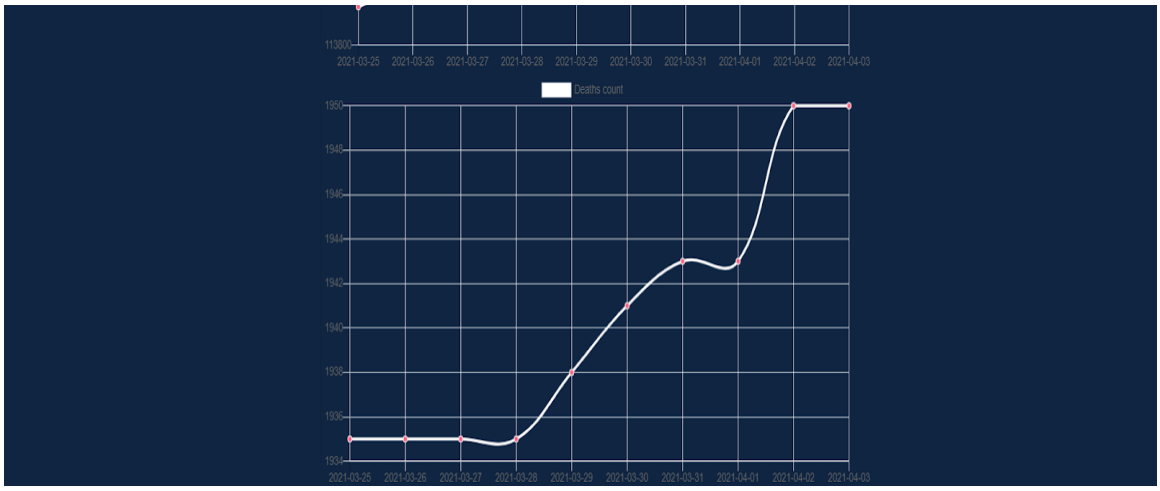
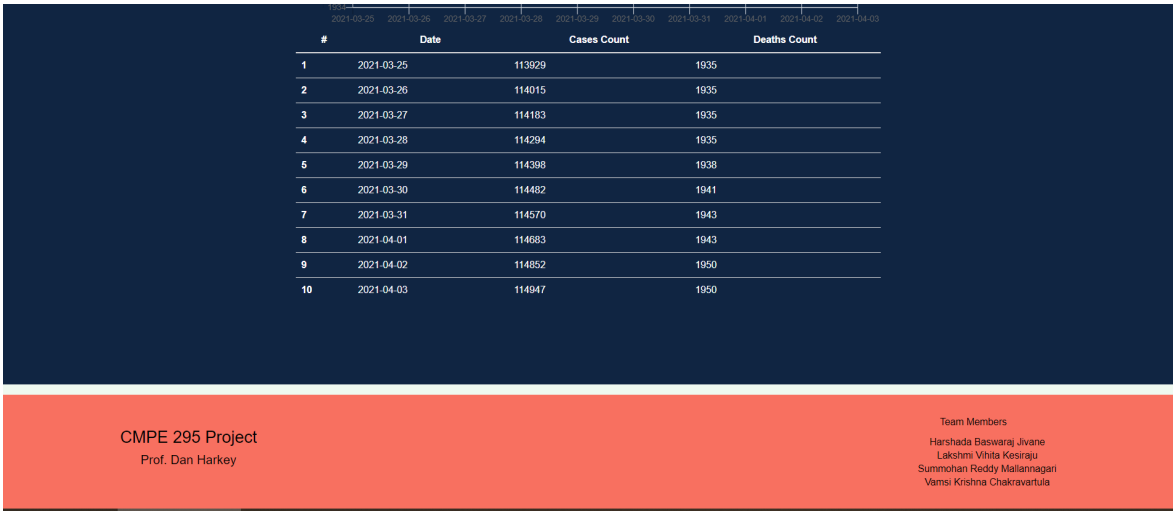


Figure 5.3.2.2 Death Count for COVID-19



#	Date	Cases Count	Deaths Count
1	2021-03-25	113929	1935
2	2021-03-26	114015	1935
3	2021-03-27	114183	1935
4	2021-03-28	114294	1935
5	2021-03-29	114398	1938
6	2021-03-30	114482	1941
7	2021-03-31	114570	1943
8	2021-04-01	114683	1943
9	2021-04-02	114852	1950
10	2021-04-03	114947	1950

Figure 5.3.2.3 Tabular column for cases

5.4 Twitter Analysis

Facts that are stated by WHO are compared with the tweets and are shown on the web application.

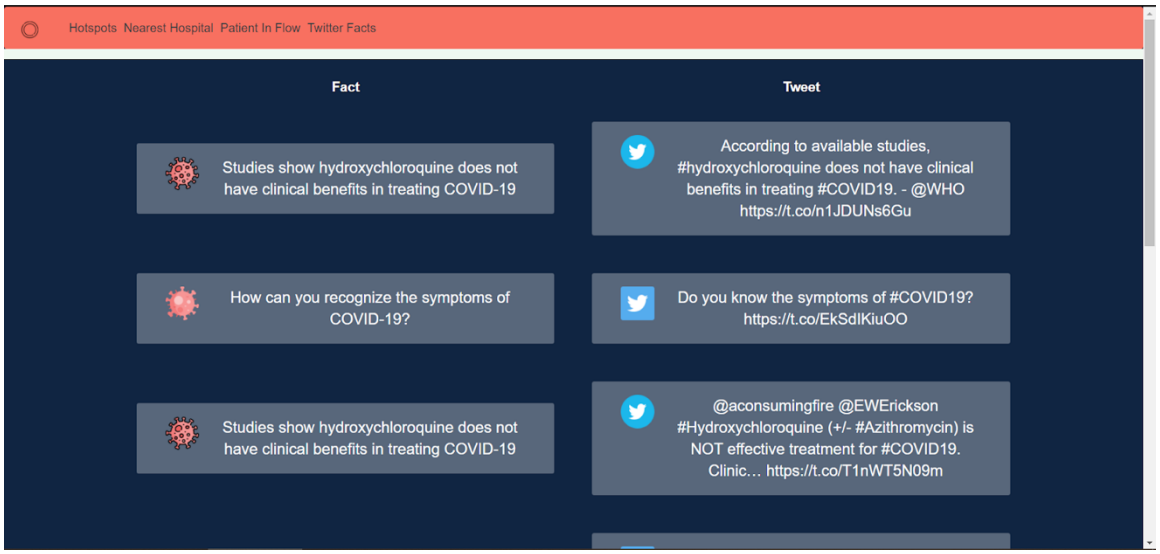


Figure 5.4.1 Showing Tweets and Facts

5.4.1 APIs and Libraries

Tweepy: It is a Python library which helps to access the Twitter data. This API helps in getting the data from Twitter with various parameters and return responses and methods.

Ref: https://docs.tweepy.org/

```
import tweepy

auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
auth.set_access_token(access_token, access_token_secret)

api = tweepy.API(auth)

public_tweets = api.home_timeline()
for tweet in public_tweets:
    print(tweet.text)
```

```
auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
```

```
auth = tweepy.OAuthHandler(consumer_key, consumer_secret,
callback_url)
```

NLTK: Natural language Toolkit is used for working on human language. It is used for applying statistical natural language processing. It is used for preprocessing of text data. It performs tokenization, stemming, lemmatization, semantic reasoning and many more.

```
def clean_text(text):
    '''Make text lowercase, remove text in square brackets,remove links,remove punctuation
    and remove words containing numbers.'''

    text = re.sub('https?://\S+|www\.\S+', '', text)
    text = re.sub(r"(@[A-Za-z0-9]+)|([^0-9A-Za-z \t])|(\w+:\/\/\S+)|^rt|http.+?", "", text)
    text = text.lower()
    #text = re.sub('[^a-zA-Z]', ' ', text)

    text = re.sub('\[.*?\]', '', text)
    text = re.sub('<.*?>+', '', text)
    text = re.sub('[%s]' % re.escape(string.punctuation), '', text)
    text = re.sub('\n', '', text)
    text = re.sub('\w*\d\w*', '', text)
    text = text.split()
    text = [wordnet.lemmatize(word) for word in text if not word in set(stopwords.words('english'))]
    text = ' '.join(text)
    return text
```

Stemming and Lemmatization are almost similar. Stemming removes or stems the word which often gives incorrect meaning and spelling errors. Lemmatization considers the context of the word and gives a meaningful word which is called Lemma. Lemmatization is used for getting better contextual meaning.

Ref: https://www.nltk.org/

5.4.2 BERT

BERT: Bidirectional Encoder Representations from Transformers is a Neural Network based model, which is mainly used for Natural Language Processing. It computes the vector space representations based on the text provided into deep learning models. The Transformer Encoders in the BERT convert each token of the text into vectors before and after the processing.

Ref: https://github.com/google-research/bert

Chapter 6. Testing and Verification

6.1 API Unit Testing

6.1.1. Mocha Testing

Mocha is one of the well-known and easy testing frameworks for node.js and to implement test suites in the backend. The tests run serially providing accurate reports of the results and exceptions raised while testing.

6.2 Model Testing

6.2.1 Predictive analysis

The datasets available are divided into training and testing sets in a least possible ratio. The training part of the dataset is used for model training and the testing part for validation of the model.

The deployment model is tested based on the weekly data entered into the warehouse to generate latest predictions.

6.2.2 Clustering

The model will be tested based on new data as every week there will be a new set of data entering the system and the model will run on the latest data to update or add new clusters.

Chapter 7. Performance and Benchmarks

7.1 Model Evaluation

- Silhouette Coefficient

  Silhouette refers to a method of interpretation and validation of consistency within clusters of data. The technique provides a succinct graphical representation of how well each object has been classified.

  Two distance scores that are used to calculate the silhouette coefficient:

  a: The average distance between one data point and all other points in the same cluster

  b: The average distance between one data point and all other points in the next nearest cluster.

$$s = \frac{(b - a)}{max(b - a)}$$

Figure 7.1.1 Silhouette Formula

- Root Mean Square Error

  It is used to find the spread of regression values around the average of the values. The residuals are calculated first with the difference between actual values and the predicted values.

- Confusion matrix

  This would be useful to display the correct and incorrect values predicted by the classification model in respect to the actual values. Sensitivity and specificity can also be estimated using these values.

- ROC curve

  It is a graphical representation of performance of a model in classification based on True positive rate and False Positive rate.

  - Relative Absolute Error

  Relative Absolute Error (RBA) measures the performance of the model in predicting the model. This can be achieved with values having different units as well.

- Cosine Similarity

  It is a metric to generally measure how similar two documents or sentences are. The similarity is irrespective of the two document sizes. Mathematically it is a measure of the angle between the two documents. The smaller the angle is, the greater the similarity between the documents.

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}},$$

Figure 7.1.2 Cosine Similarity Formula

7.2 Load Benchmarking

7.2.1 Load testing using JMeter

Performance tests can be conducted with a large number of API requests, measuring the load handling capacity of the application. JMeter provides API test capabilities with variable load and query parameters.

Chapter 8. Deployment, Operations, Maintenance

Describe any deployment strategies, operational needs, and maintenance required for your project.

8.1 VPC and Subnets

A Virtual Private Cloud is used to set up a virtual topology of IP addresses that can be specifically tailored to the requirements. Usage of a VPC provides isolation from the public network, by configuring the subnets and network policies.

The VPC created here operates with the IP ranges in the CIDR block 192.168.0.0/16. These IP addresses fall in the private set of IPs recommended, according to the industry standards.

The VPC has two subnets dividing the above IP ranges. A subnet is a regional resource that divides the parent network into sub-networks. Two subnets, private and public are used for hosting the frontend and backend servers respectively. The IP ranges for these are 192.168.2.0/20 and 192.168.1.0/20. This gives ample network bandwidth to scale the compute engines.

8.2 Load Balancers

Google Cloud provides server-side load balancing which can be used to distribute the incoming traffic among the available servers. It can also be used to detect and remove unhealthy nodes that fail a predefined health check.

Out of the available load balancers, the External HTTP (S) is the most suitable for this project, with the traffic majorly being of the HTTPS protocol. The HTTP 80 port is used for distributing the incoming requests.

The balancing policy can be either based on the requests or the utilization of the resources (Rate and Utilization). Here, the utilization based balancing policy is being used, for its complex analysis of the workloads on each of the servers.

8.3 Auto Scaling

Auto scaling enables the feature to add or delete the instance from a virtual machine if it is not managed or supported within the region or zone. This helps to reduce cost by automatically adjusting the instance based on the traffic and measuring the load. Auto scaling can be done based on the following parameters.

- CPU Utilization - It is the simplest form to scale a managed instance based on a target utilization set for that group of instances.
- HTTP load balancing serving capacity - It works based on an external load balancer to determine the target capacity of the balancing instance.
- Cloud monitoring metrics - Managing the instances based on monitoring metrics is achieved either by scaling using per-instance metrics or using per-group metrics.

Out of these parameters, CPU utilization is used to auto scale where it adds more VM instances if the average CPU utilization exceeds the target utilization and removes the instances if it is vice versa. This best suits the project as it can run the filtering of data.

8.4 Compute Engines

Compute Engines are the Virtual Machines that are provided by the Google Cloud Platform. Out of the multiple instance types with varied CPU and storage capacities, the F1 micro type is being used from the free-tier. The F1 micro instance will be sufficient to cater to the Node.js backend servers and also the React Frontend servers. If the need be, an easy upgrade can be performed to the everyday E2 machines. This gives a varied range of available computing resources.

8.5 Network Routing and Firewalls

Network from the internet has to be routed through multiple levels before it reaches the deployed servers. The outermost firewall controls the data flow of the VPC, with the default as all incoming traffic blocked and all outgoing traffic allowed. This firewall table can be modified to include the trusted IP addresses or be kept open to the internet.

A Route is the defined network path that is followed from the point of ingress to the destination server node. A route for each subnet is automatically created at the time of creating the subnets. The default route has the priority as 1000 and the broadest IP range possible with 0.0.0.0/0.

Chapter 9. Summary, Conclusions, and Recommendations

9.1 Summary

The web application lists the hotspots on a map which help to understand which regions are more affected by the pandemic and these regions could be avoided for travelling or other purposes. Nearest hospitals assist users to know the closest hospital near them and also near any specified location. Predictive analysis module predicts the number of cases for a community which might be insightful for hospitals in managing the healthcare resources. The correlation between tweets and fact from WHO helps users to understand if the trending tweets are valid or invalid.

9.2 Conclusions

COVID-19 has made a global disruption in every aspect of our social fabric. The pandemic has taken a toll on the healthcare system, increasing the need for acute care that has overtaxed some hospitals. The lack of public health information systems impacted the balancing and distribution of the critical resources from areas of surplus to undersupply.

A predictive analysis-based web application that can support the hospitals with the surges in patient inflow, assist the public users in finding treatment and predict the hotspots based on the existing and historical admittance and infections data would be one of the project's goals. Clustering hotspot regions based on the environmental, socio-economic similarities, and accurate social media data would help identify the patterns for applying control policies.

9.3 Recommendations for Further Research

There are a number ways research community and public would benefit from further analysis on coronavirus pandemic, including some of the applications here:

- The clustered hotspots can be utilized to notify the governing authorities if the density of hotspot goes beyond a threshold and the authorities can then implement policies to control the spread.
- The prediction of increase in cases could be integrated with the hospital data ingestion pipelines to know the need of hospital equipment and resources such as bed availability .
- The location data from nearest hospital applications can be integrated with hospital records and a real time record of covid/non-covid treatment specific resources could be made available to the public for faster and convenient access to treatments.
- The correlation between tweets and facts could help any social media platform to raise a red flag when misleading information is spreading and the public needs to be made aware of the facts from trusted sources.

Glossary

EC2	Elastic Compute
S3	Simple Storage Service
SQL	Structured Query Language
DB	Database

References

1. Maroko, A.R., Nash, D. & Pavilonis, B.T. COVID-19 and Inequity: a Comparative Spatial Analysis of New York City and Chicago Hot Spots. *J Urban Health* 97, 461–470 (2020). https://doi-org.libaccess.sjlibrary.org/10.1007/s11524-020-00468-0

   Description: The goal of this ecological cross-sectional study is to examine the demographic and economic nature of spatial hot and cold spots of SARS-CoV-2 rates in New York City and Chicago.

2. Smith CD, Mennis J. Incorporating Geographic Information Science and Technology in Response to the COVID-19 Pandemic. Prev Chronic Dis 2020;17:200246. DOI: http://dx.doi.org/10.5888/pcd17.200246

   Description: The paper talk about applications of GIS&T include developing spatial data infrastructures for surveillance and data sharing, incorporating mobility data in infectious disease forecasting, using geospatial technologies.

3. I. M. K. Karo, K. MaulanaAdhinugraha and A. F. Huda, "A cluster validity for spatial clustering based on davies bouldin index and Polygon Dissimilarity function," 2017 Second International Conference on Informatics and Computing (ICIC), Jayapura, 2017, pp. 1-6, doi: 10.1109/IAC.2017.8280572.

Description: Main subject of this paper is a cluster validity for spatial region clustering by using modified of Davies Bouldin index with Polygon Dissimilarity function.

4. Chetashri Bhadane and Ketan Shah. 2020. Clustering Algorithms for Spatial Data Mining. In Proceedings of the 2020 3rd International Conference on Geoinformatics and Data Analysis (ICGDA 2020). Association for Computing Machinery, New York, NY, USA, 5–9. DOI:https://doi.org/10.1145/3397056.3397068

   Description: This paper gives an introduction to the clustering techniques used in the GPS based mobility datasets to find clusters.

5. Jose Cadena, Achla Marathe, and Anil Vullikanti. 2020. Finding Spatial Clusters Susceptible to Epidemic Outbreaks due to Undervaccination. In Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 1786–1788.

   Description: This paper aims to find the possible geographical hotspots in case of an epidemic based on the prior vaccination records.

6. Jiandong Tu, Chongcheng Chen, Hongyu Huang, and Xiaozhu Wu. Proceedings. 2005 IEEE International Geoscience and Remote Sensing Symposium, 2005.

IGARSS '05. Vol. 2. 2005. 2005 IEEE International Geoscience and Remote Sensing Symposium, 2005. IGARSS '05. Web.

Description: This paper attempts to provide a demonstration of visual hierarchical clustering based on the graph partitioning algorithm VSG-CLUST.

7. L. Luo, X. Xu, J. Li and W. Shen, "Short-term forecasting of hospital discharge volume based on time series analysis," 2017 IEEE 19th International Conference on e-Health Networking, Applications and Services (Healthcom), Dalian, 2017, pp. 1-6, doi: 10.1109/HealthCom.2017.8210801.

   Description: This paper discusses the machine learning models and techniques for predicting the bed resources in order to support the management of the hospital for decision making on admits.

8. J. M. Keller, A. R. Buck, A. Zare and M. Popescu, "A human geospatial predictive analytics framework with application to finding medically underserved areas," 2014 IEEE Symposium on Computational Intelligence in Big Data (CIBD), Orlando, FL, 2014, pp. 1-6, doi: 10.1109/CIBD.2014.7011525.

   Description: This paper is a study about geographic data across various locations to improve medical conditions in underserved areas.

9. G. Bordogna, L. Frigerio, A. Cuzzocrea and G. Psaila, "Clustering Geo-tagged Tweets for Advanced Big Data Analytics," 2016 IEEE International Congress on

Big Data (BigData Congress), San Francisco, CA, 2016, pp. 42-51, doi: 10.1109/BigDataCongress.2016.78.

Description: This paper is about performing analysis on locations tagged to the tweets and to suggest trips accordingly.

10. https://medium.com/predict/three-popular-clustering-methods-and-when-to-use-each-4227c80ba2b6
11. https://towardsdatascience.com/a-quick-start-of-time-series-forecasting-with-a-practical-example-using-fb-prophet-31c4447a2274
12. Taylor SJ, Letham B. 2017. Forecasting at scale. PeerJ Preprints 5:e3190v2 https://doi.org/10.7287/peerj.preprints.3190v2
13. https://airflow.apache.org/
14. https://aws.amazon.com/
15. https://azure.microsoft.com/en-us/
16. https://cloud.google.com/deep-learning-vm
17. https://spark.apache.org/docs/latest/api/python/index.html