International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Disaster Management (Data Analysis)

[1]Prof. Anurag Singh, [2]Ansari Mohammed Moin, [3]Shaikh Umair, [4]Mohammed Ali, [5]Shaikh Shaziya

[1]Professor of Pravin Patil Polytechnic, Department of Information Technology, Bhayandar, Thane, India - 401107
[2,3,4,5]Dept. Information Technology, Bhayandar, Thane, India - 401107

ABSTRACT-

This paper presents a comprehensive overview of our project Disaster Management (Data Analysis) that based on today's Machine Learning. Machine Learning (ML) is a field of artificial intelligence that empowers computers to learn and make predictions or decisions without explicit programming, relying on patterns and data-driven insights. The paper begins with the brief introduction of our Machine Learning. Next, the paper highlights the problem statement of Machine learning in various fields.

Keywords- Disaster Management, Machine Learning, Data Analysis and Data Driver Insights

I. Introduction

The Disaster Management (Data Analysis) is a Machine Learning Model that is based on Prophet Model of Machine Learning and Predicts Earthquake. What does Machine Learning mean? Machine Learning is a field of Artificial Intelligence that empowers computers to learn and make predictions or decisions without explicit programming, relying on patterns and data-driven insights. This Machine Learning model is limited and only available for Tokyo City of Japan. We have created a ML model that predicts up the Earthquake and gives us the significant outcomes with that prediction. It includes Time, Latitude and Longitude, Magnitude, Intensity of Earthquake etc. We will then store up that output and display it in a Dashboard powered up Power bi that gives us various Analytical Insights such as Bar Graphs, Map etc.

II. Problem Statement

This project introduces a simulated disaster management system that employs machine learning for accurate earthquake prediction in Tokyo. Recognizing the model's limitations, its applicability is confined to Tokyo. The core objective is to showcase the developed machine learning model and its associated user-friendly dashboard. The motivation stems from the inadequacies of traditional earthquake prediction methods, aiming to provide a more precise and engaging solution. Overcoming existing limitations, the project emphasizes not only accuracy but also the accessibility of earthquake predictions to the wider community.

III. Literature Survey

In recent years, machine learning (ML) has gained widespread applications across various domains, transforming industries with its predictive capabilities. The inception of ML can be traced back to the mid-20th century, with significant advancements leading to its current prominence. The term "machine learning" was officially coined by Arthur Samuel in 1959, marking a pivotal moment in the field.

One of the seminal works in ML is the development of the perceptron by Frank Rosenblatt in 1957, laying the foundation for neural network-based learning. In the early 2000s, the availability of large datasets and computational power contributed to the resurgence of neural networks, particularly deep learning, propelling ML into new frontiers.

A crucial turning point came with the ImageNet competition in 2012, where deep learning models outperformed traditional computer vision techniques, showcasing the potential of deep neural networks for complex tasks. This triumph catalyzed the integration of ML into diverse sectors, from healthcare to finance, revolutionizing decision-making processes.

IV. Methodology

The machine learning (ML) methodology involves several key steps to develop an effective earthquake prediction model and a user-friendly dashboard. The process encompasses data collection, preprocessing, model training, evaluation, and deployment.

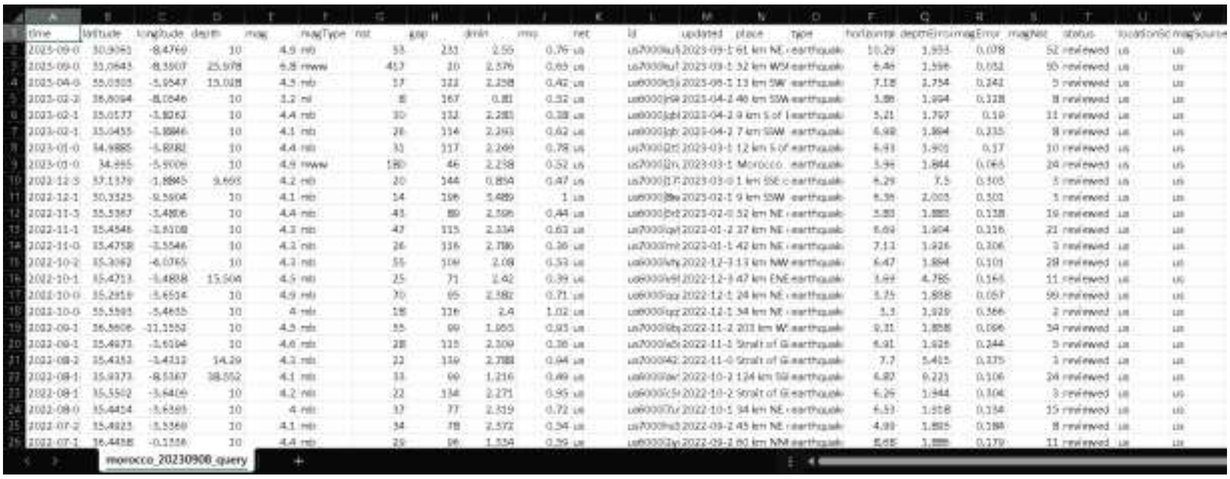
1. **Data Collection:**
   - Gather historical earthquake data, including date, time, location, magnitude, and depth.
   - Collect relevant geographical and geological features that may impact earthquake occurrence.
2. **Data Preprocessing:**
   - Clean and preprocess the collected data, handling missing values and outliers.
   - Convert time stamps to a usable format and extract additional features for analysis.
3. **Model Training:**
   - Utilize a machine learning algorithm, such as Prophet or other time-series models, to train the earthquake prediction model.
   - Split the dataset into training and testing sets to evaluate model performance.
4. **Evaluation:**
   - Assess the model's accuracy using metrics like Root Mean Squared Error (RMSE) and Mean Absolute Percentage Error (MAPE).
   - Validate predictions against historical earthquake data to ensure reliability.
5. **Dashboard Development:**
   - Design an interactive dashboard to visualize real-time earthquake predictions.
   - Incorporate geographical maps, intuitive charts, and user-friendly interfaces for enhanced accessibility.
6. **Integration and Deployment:**
   - Integrate the trained ML model with the dashboard for seamless interaction.
   - Deploy the combined system, allowing users to access earthquake predictions and relevant information.
7. **User Testing and Feedback:**
   - Conduct user testing to gather feedback on the dashboard's usability and effectiveness.
   - Iterate on the design based on user suggestions to enhance the overall user experience.
8. **Documentation:**
   - Document the entire ML methodology, including data sources, preprocessing steps, model training details, and dashboard development.
   - Provide clear instructions for users to understand and navigate the dashboard effectively.

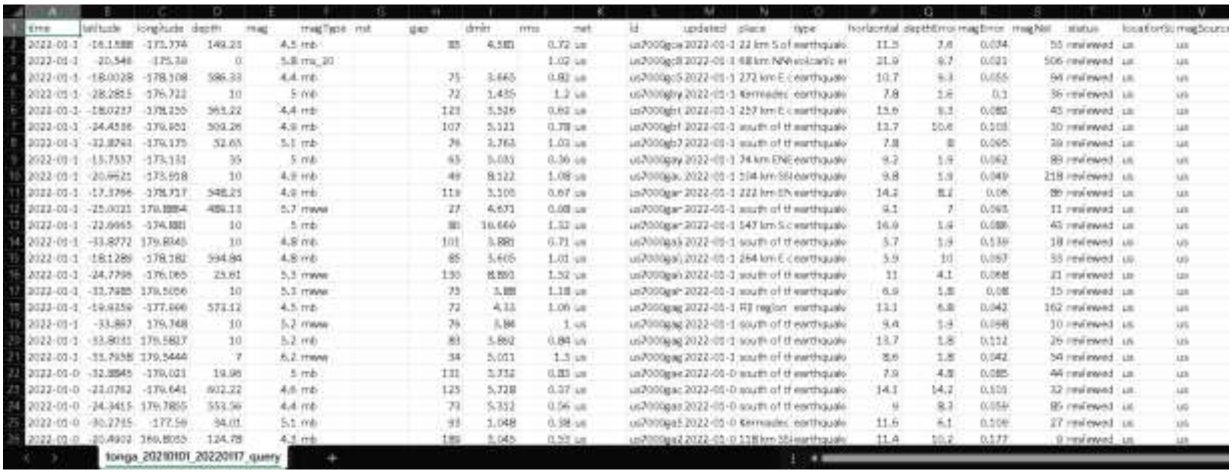
V. Working of Project

Our Project is divided into two parts. The first part is of Machine Learning Model i.e. based on prophet model and the second part is our GUI i.e. Dash Board powered up by Power bi. For now, we have only viewed the high-level components we will discuss the whole back-end process along with the Front end.

First - Research the earthquakes in Tokyo, Gathering up the Datasets, Analyzing them with the insights they provide, Extracting Classes and Features from dataset for our ML Model. We have trained our model on three dataset namely

- morocco_20230908_query.csv
- japan-20190101_20211203_query.csv
- tonga_20210101_20220117_query.csv





Second - Clearing up data, dealing out with blank spaces because we didn't explicitly collect that data we used some one else data that was uploaded on Kaggle for public use.

Third – For now we will view the code in our ML model in python we will be going through all of our ML code

Importing Libraries:

```
import pandas as pd
from datetime import datetime, timedelta
from prophet import Prophet
import joblib
from sklearn.metrics import mean_squared_error
import random
```

Data Loading:

```
# Load historical earthquake data for Tokyo
tokyo_data = pd.read_csv('japan-20190101_20211203_query.csv')

# Load historical earthquake data for Tonga
tonga_data = pd.read_csv('tonga_20210101_20220117_query.csv')

# Load historical earthquake data for Morocco
morocco_data = pd.read_csv('morocco_20230908_query.csv')
```

Data Combination:

```
# Combine the datasets
combined_data = pd.concat([tokyo_data, tonga_data, morocco_data], ignore_index=True)
```

Data Preparation:

```
# Prepare the historical data for the Prophet model
combined_data['ds'] = pd.to_datetime(combined_data['time']).dt.tz_localize(None)  # Remove timezone
combined_data['y'] = combined_data['mag'].astype(float)
```

Model Initializing and Training:

```
# Initialize and fit the Prophet model
prophet_model = Prophet()
prophet_model.fit(combined_data)
```

Model Saving:

```
# Save the Prophet model using joblib
joblib.dump(prophet_model, 'prophet_model_combined.joblib')
```

User Input and Model Loading:

```
# Get user input for prediction
user_input_date = input("Enter the date or date range (YYYY-MM-DD or YYYY-MM-DD to YYYY-MM-DD): ")

# Process user input
user_dates = pd.date_range(*user_input_date.split(' to ')) if ' to ' in user_input_date else [pd.to_datetime(user_input_dat

# Load the saved Prophet model
prophet_model_loaded = joblib.load('prophet_model_combined.joblib')
```

Prediction Loop, Result Display and Accuracy Calculation:

```
# Make predictions for user-provided dates
for user_date in user_dates:
    # Create a DataFrame with the user-provided date
    user_data = pd.DataFrame({'ds': [user_date]})

    # Make predictions for the user-provided date
    user_predictions = prophet_model_loaded.predict(user_data)

    # Display the forecasted values for the user-provided date
    print(f"\nForecasted values for {user_date} - Tokyo:")
    print(user_predictions[['ds', 'yhat', 'yhat_lower', 'yhat_upper']])

    # Generate random time within the day
    random_time = random.uniform(0, 23.99)
    forecasted_time = user_date.replace(hour=int(random_time), minute=0, second=0)

    # Get a more relevant location within Tokyo (assuming Tokyo's coordinates)
    tokyo_center = {'latitude': 35.6895, 'longitude': 139.6917}
    random_location = {
        'latitude': tokyo_center['latitude'] + random.uniform(-0.1, 0.1),
        'longitude': tokyo_center['longitude'] + random.uniform(-0.1, 0.1),
    }
```

Result Storage and CSV Saving:

```
# Create a DataFrame from the list of forecast results
forecast_results_df = pd.DataFrame(forecast_results)

# Save forecasted results to a CSV file
forecast_results_df.to_csv('latitude_forecast_results.csv', index=False)
```

Overall Accuracy Statistics:

```python
# Display overall accuracy statistics if at least one prediction was made
if total_rmse > 0:
    average_rmse = total_rmse / len(user_dates)
    average_mape = total_mape / len(user_dates)
    print(f"\nOverall Accuracy Statistics - Tokyo:")
    print(f"Average RMSE: {average_rmse}")
    print(f"Average MAPE: {average_mape}%")
```

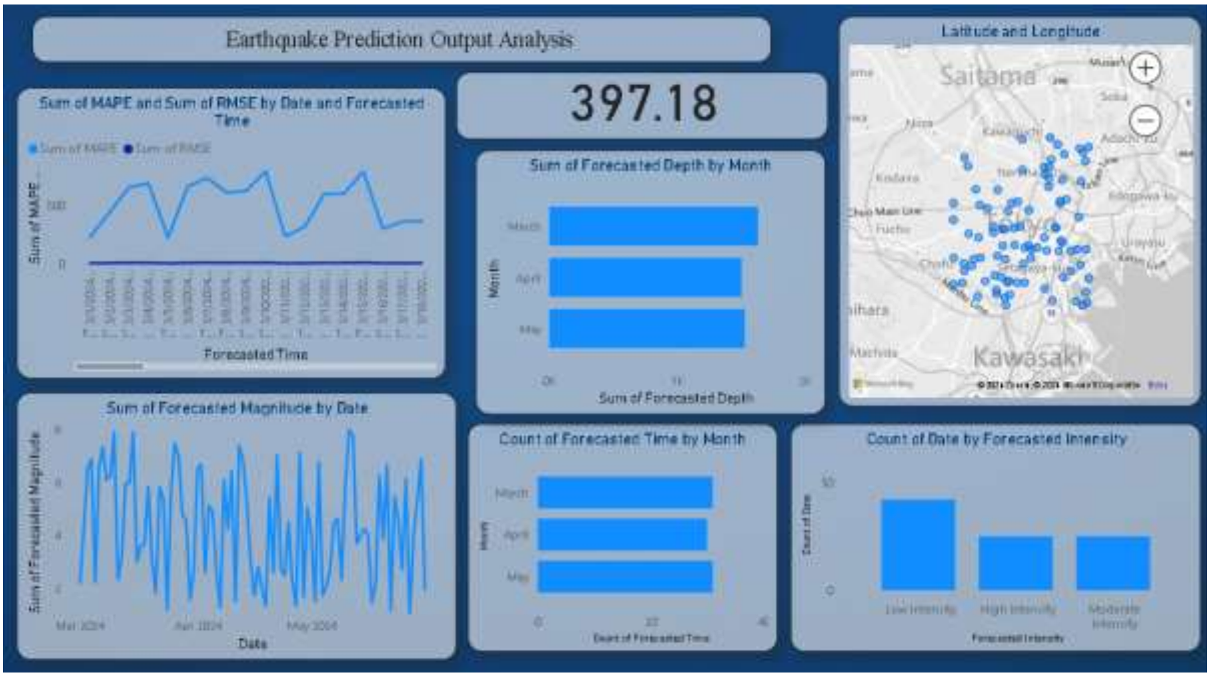
The output of our code:

```
Enter the date or date range (YYYY-MM-DD or YYYY-MM-DD to YYYY-MM-DD): 2024-03-01 to 2024-05-31

Forecasted values for 2024-03-01 00:00:00 - Tokyo:
          ds      yhat  yhat_lower  yhat_upper
0 2024-03-01  4.413818    4.310939    5.241304
Forecasted Time: 2024-03-01 04:00:00
Forecasted Location: Latitude -55.2461929288217, Longitude 167.38467627553194
Forecasted Magnitude: 4.413818324374
Forecasted Depth: 82.30543902609145
Forecasted Intensity: Rare Medium Intensity

Tokyo - RMSE: 4.413818324374
Tokyo - MAPE: 441.381832437399996%
Tokyo - Percentage Difference with Lower Interval: 2.330853683754047 4%
Tokyo - Percentage Difference with Upper Interval: 18.74761589296159%
```

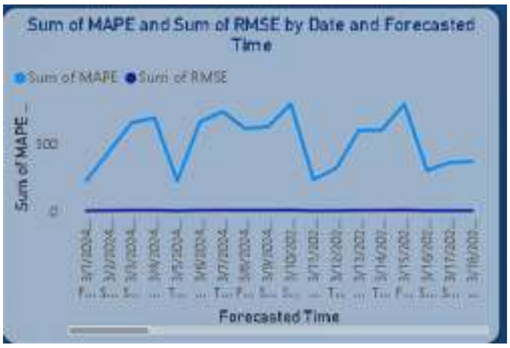
For now, we will view our dashboard:



Let’s view each of the Widgets of our dashboard

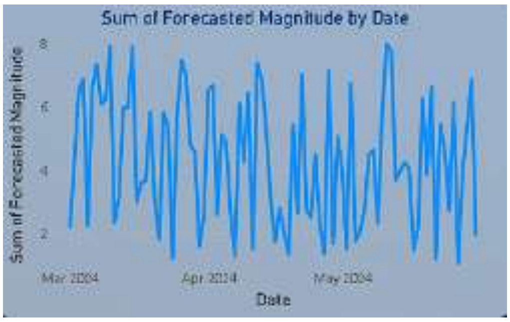
Sum of MAPE and Sum of RMSE by Date and Forecasted Time:

The "Sum of MAPE" and "Sum of RMSE" by Date and Forecasted Time refer to the total or cumulative Mean Absolute Percentage Error (MAPE) and Root Mean Squared Error (RMSE) calculated across multiple forecasted instances for different dates and times.



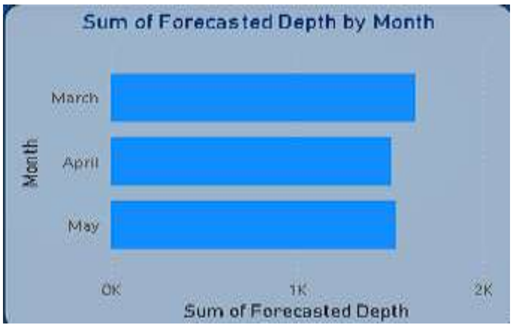
Sum of Forecasted Magnitude by Date:

The total or cumulative forecasted magnitude across multiple forecast instances for different dates. It represents the sum of the predicted magnitudes for all instances on a specific date.



Sum of Forecasted Depth by Month:

A metric that represents the cumulative forecasted depth of earthquakes for each month. It provides the total forecasted depth across all earthquake predictions made for different dates within a specific month.

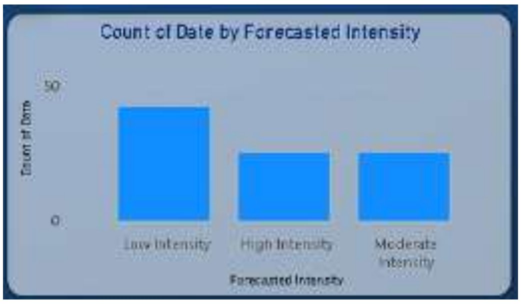


Count of Forecasted Time by Month:

A metric that indicates the number of forecasted times for which earthquake predictions were made within each month. It gives an overview of how many times the model provided predictions for different dates within a specific month.

Count of Date by Forecasted Intensity:

A metric that indicates the number of dates for which earthquake predictions were made, categorized by the forecasted intensity levels. It gives you an overview of how many times the model predicted earthquakes with different intensity levels on various dates.

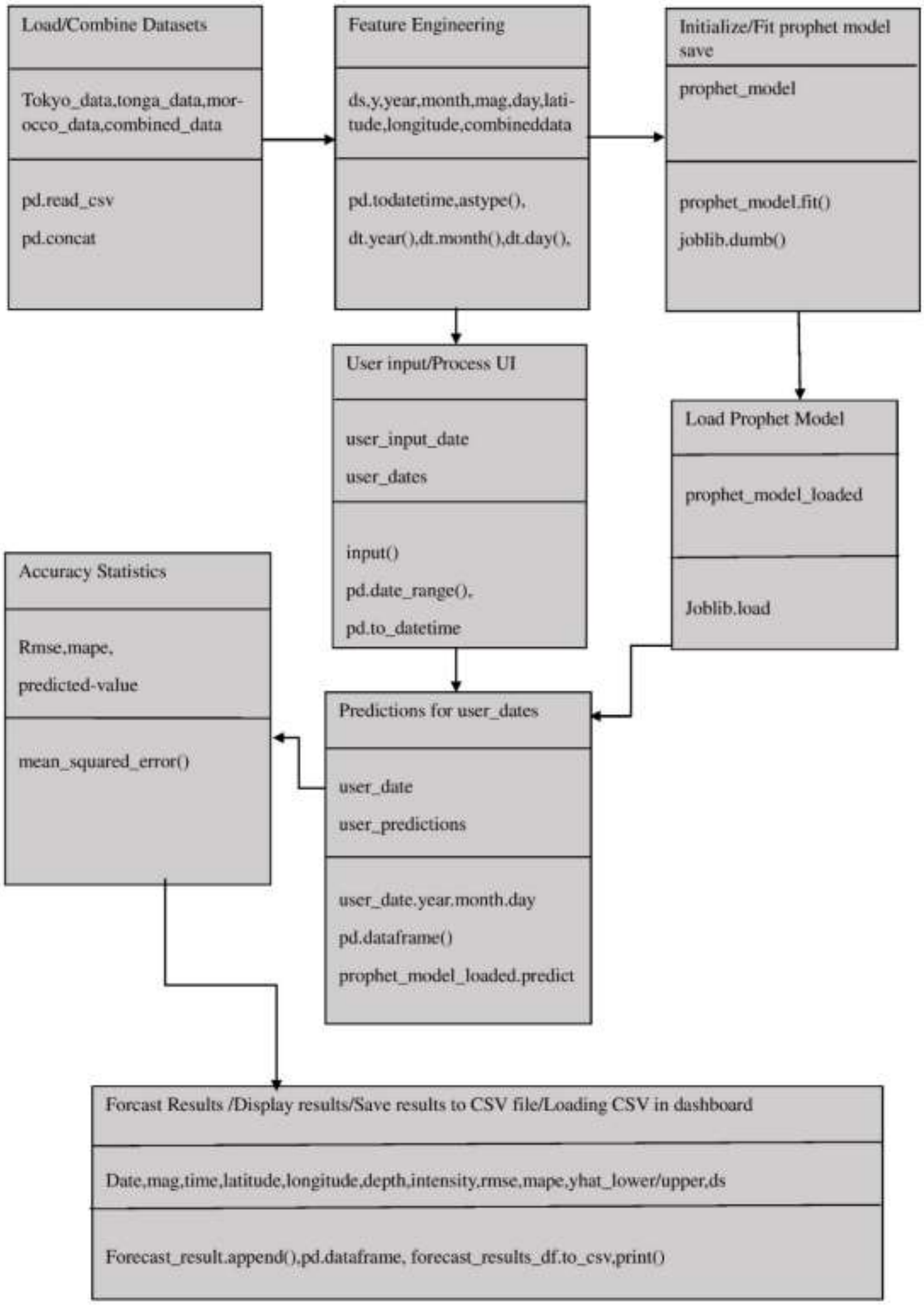


Latitude and Longitude:

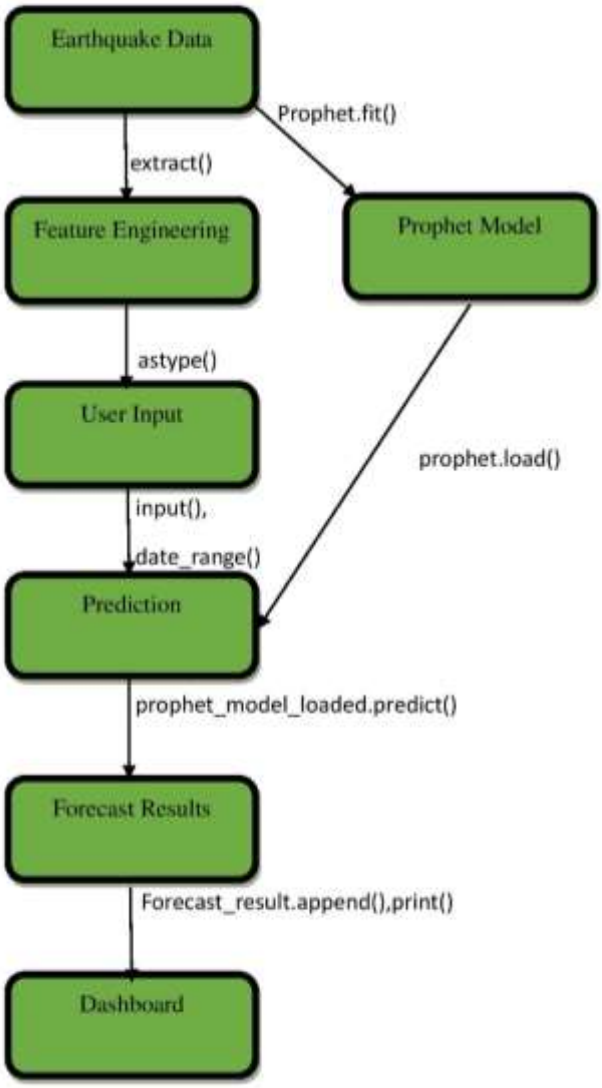
The latitude and longitude values are geographic coordinates that specify the position of a location on the Earth's surface. They are expressed in degrees, minutes, and seconds and uniquely identify a point on the Earth. In this case latitude and longitude represents Tokyo and surrounding area of Tokyo.

Activity Diagram

Class Diagram

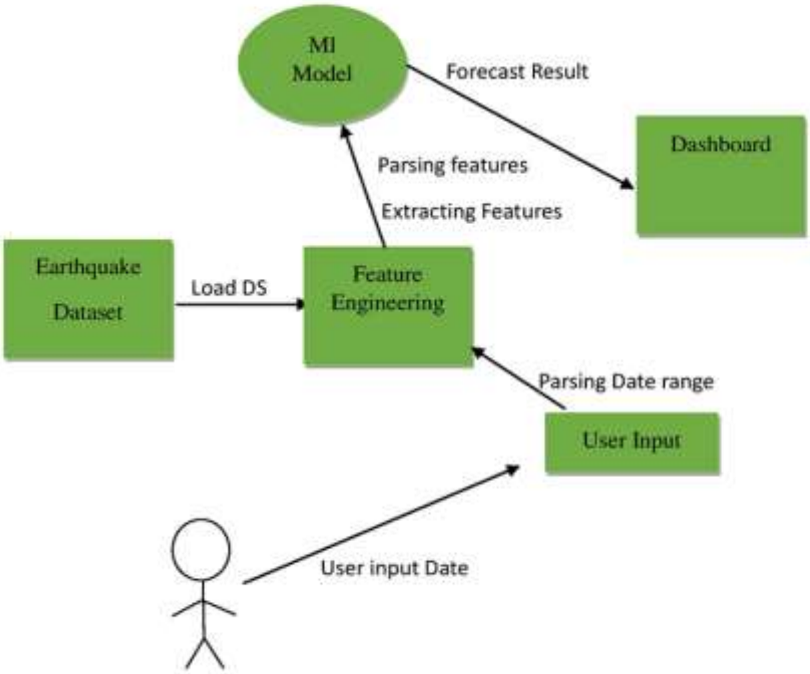


Collaboration Diagram

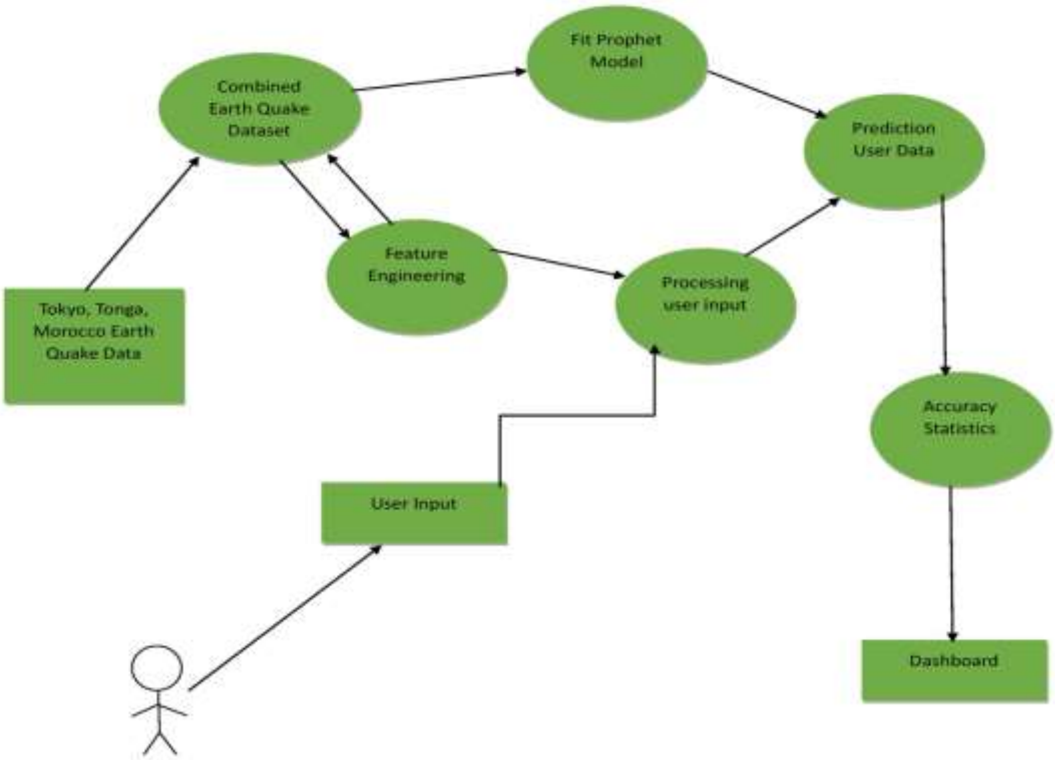


DFD Level 0

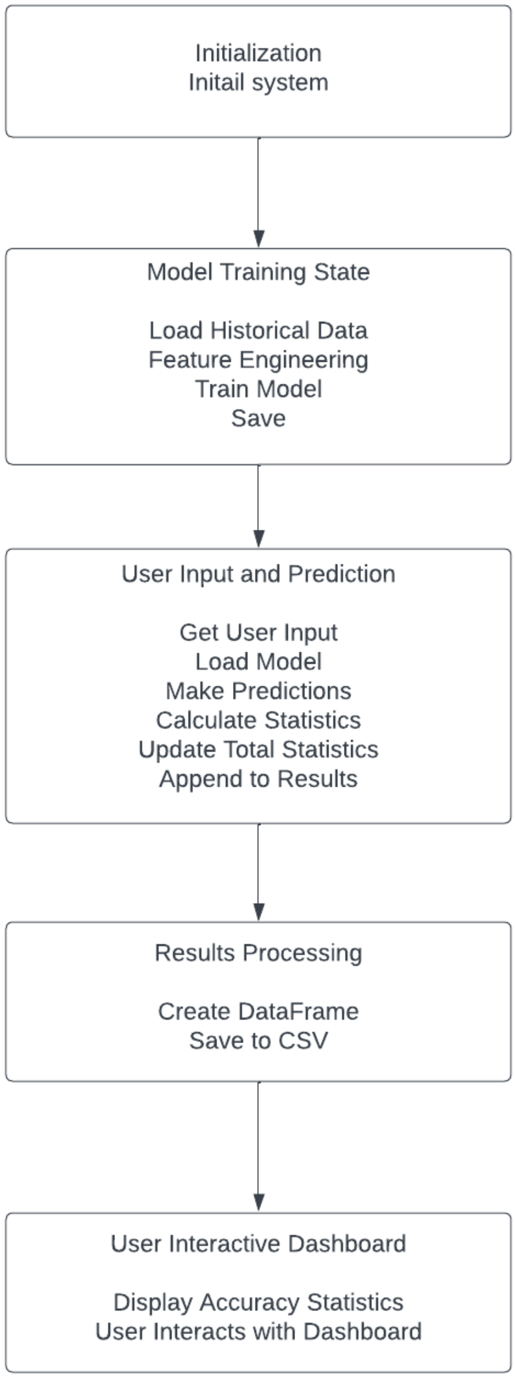
DFD level 1



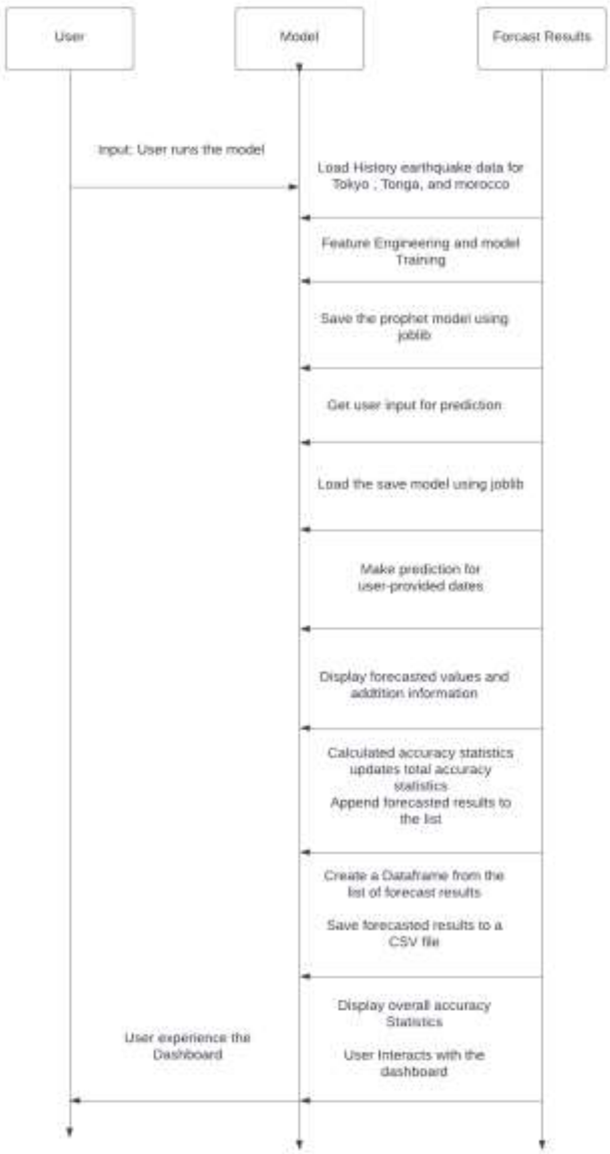
DFD Level 2



State Diagram



Sequence Diagram



VI. Existing System

Earthquake prediction remains a challenging task, and accurate short-term prediction is not yet achievable. Seismologists primarily focus on earthquake monitoring and early warning systems to mitigate the impact of seismic events.

Various institutions, such as the United States Geological Survey (USGS) and the Japan Meteorological Agency, operate seismic networks and provide real-time earthquake information to the public.

Advanced research and technologies, including machine learning models, are being explored for improved seismic hazard assessments, but widespread accurate prediction remains elusive.

VII. Limitations

Due to not having a power full hardware to train our ML model we can not consistently train our model on Multiple datasets and on more models like Prophet Model. Since we do not have real time data for training model's accuracy may vary.

The dataset and model are primarily focused on earthquake occurrences in specific regions (Tokyo, Tonga, and Morocco). Generalizing findings to global scale or other diverse regions may not be accurate, as seismic activities vary significantly worldwide.

Limited computational resources and time constraints may have hindered an exhaustive hyperparameter search.

Fine-tuning hyperparameters is crucial for optimizing model performance.

The model's performance is not extensively validated on diverse datasets or against alternative forecasting methods

VIII. Future Direction

In the future, our earthquake prediction model aims to evolve beyond its current state, exploring innovative avenues to enhance its predictive capabilities and societal impact. By Integrating real time seismic data it will enable model to respond more dynamically for earthquake patterns. Expanding the model's scope to cover a more extensive geographical footprint, encompassing diverse seismic regions worldwide. This will involve collaborating with international agencies to gather comprehensive datasets for training and validation.

Implementing a continuous improvement cycle by periodically retraining the model with updated data and fine-tuning hyperparameters. This ensures the model's adaptability to evolving seismic patterns and changing environmental conditions.

In future with Proper hardware and right choice of models we can then train our model on multiple datasets and then we will incorporate our model

Directly into the sensors where our model then will study trends ,analyzing data trends and make more accurate predictions.

IX. Advantages

- A more interactive soft Dashboard will help analysist and users to understand more consistently Earthquake Predictions.
- The model translates complex seismic data into user-friendly insights, empowering individuals and communities to make informed decisions about their safety.
- One can get a broader scope regarding Earthquake Predictions in Tokyo or around Tokyo.

Disadvantages

- Dependency on Historical Data: The model relies heavily on historical earthquake data for training. If the dataset is limited, biased, or does not represent the full spectrum of seismic events, the model's accuracy may be compromised, especially for regions with sparse historical earthquake record.
- Challenges in Generalization: The model may face challenges in generalizing predictions to diverse geographical and geological settings. Factors such as unique geological characteristics, local fault lines, and varying tectonic activities can introduce complexities that the model may struggle to capture.

X. Conclusion

Our machine learning model for earthquake prediction demonstrates promising capabilities in forecasting seismic events. Despite its current limitations, such as the focus on a single city and potential constraints on real-time data, the model showcases the potential for accurate predictions. Future enhancements, including broader geographical coverage and improved real-time data integration, are essential for advancing its effectiveness.

XI. Reference

https://www.kaggle.com/datasets/stpeteishii/earthquake-in-japan?resource=download

https://www.kaggle.com/code/stpeteishii/earthquakes-in-afghanistan-in-2023-1-r

https://www.kaggle.com/datasets/usgs/earthquake-database

https://www.kaggle.com/code/stpeteishii/m5-9-earthquake-in-tokyo-on-2021-10-07

https://ieeexplore.ieee.org/document/9768454

Prediction of Earthquake Using Machine Learning Algorithms by **Gaurav Singh Manral and Alka Chaudhary**