

High Level Design (HLD)

Climate Change Analysis and Forecasting

Document Version Control

Date	Version	Description	Author
20/10/2022	1	Initial HLD	Anirban Debnath
25/10/2022	2	First Update	Anirban Debnath
29/10/2022	3	Final Update	Anirban Debnath

Contents

Table of Contents

Abstract.....	4
Introduction.....	5
Why this High-Level Design Document?	5
The HLD will:.....	5
Scope.....	5
General Description	6
Product Perspective :.....	6
Problem Statement :	6
Proposed Solution :	6
Requirements.....	7
Data	7
Software and Tools	9
Hardware.....	10
Research and Methodology	11
Data Collection and EDA	11
Data Collection	11
Exploratory Data Analysis.....	11
Data Modelling and Validation.....	19
Web Application Development.....	24
Deployment.....	29
Conclusion	31

Abstract

Climate change is undoubtedly one of the biggest problems in the 21st century. Artificial Intelligence methods have recently contributed in the advancement of accurate prediction tools for the estimation and assessment of extreme environmental events and investigation of the climate change time series. The recent advancement in Artificial Intelligence including the novel machine learning and deep learning algorithms as well as soft computing applications have greatly empowered prediction methods. Through this project, we have explore, analyze the global Climatic trend and pattern on temperature component and forecast the future temperature trends using a state of art time series deep learning model. After the research, exploration and analysis on the historical data and modelling, we build and deploy an end to end web solution on the frontend to view and explore historical data as well as future forecasts generated through the deep learning model.

Introduction

Why this High-Level Design Document?

The purpose of this High-Level Design (HLD) Document is to add the necessary detail to the current project description to represent a suitable model for coding. This document is also intended to help detect contradictions prior to coding, and can be used as a reference manual for how the modules interact at a high level.

The HLD will:

- Present all of the design aspects and define them in detail
- Describe the user interface being implemented
- Describe the hardware and software interfaces
- Describe the performance requirements
- Include design features and the architecture of the project
- List and describe the non-functional attributes like:
 - Security
 - Reliability
 - Maintainability
 - Portability
 - Reusability
 - Application compatibility
 - Resource utilization
 - Serviceability

Scope

The HLD documentation presents the structure of the system, such as the database architecture, application architecture (layers), application flow (Navigation), and technology architecture. The HLD uses non-technical to mildly-technical terms which should be understandable to the administrators of the system.

General Description

Product Perspective :

This Climate Change Analysis and Forecasting Application is a Web Application powered by Streamlit, Plotly and a Deep Learning based Time Series Forecasting Model to analyze, visualize and forecast average temperatures per month.

Problem Statement :

Will analyze the change in temperatures across globe from the 17th century till now and build a multivariate deep learning-based time series model to forecast the U.S. Average temperature. Predictive models attempt at forecasting future value based on historical data. The main objective here is -

1. Analyse the changes in climate across the globe.
2. Alert if any unusual climate change happen.
3. Maintain a database to store each and every data.

Proposed Solution :

The solution here proposed is a Multi Paged Web Application through which user can input or set Country, State, Time or Year Range and based on that data (may historical or future) of average temperatures per month, statistical description like mean, median, max, min, count etc. This data also downloadable on a button click. An interactive Trend Plot to view trend based on moving average , an interactive seasonal bar plot to view monthly average temperatures and an interactive autocorrelation plot to show correlation upto 100 previous data. Except this all user also can send a feedback or message to development team.

Requirements

Data :

	A	B	C	D	E	F	G	H	I
1	dt	LandAvera	LandAvera	LandMaxT	LandMaxT	LandMinTe	LandMinTe	LandAndO	LandAndO
2	1850-01-0	0.749	1.105	8.242	1.738	-3.206	2.822	12.833	0.367
3	1850-02-0	3.071	1.275	9.97	3.007	-2.291	1.623	13.588	0.414
4	1850-03-0	4.954	0.955	10.347	2.401	-1.905	1.41	14.043	0.341
5	1850-04-0	7.217	0.665	12.934	1.004	1.018	1.329	14.667	0.267
6	1850-05-0	10.004	0.617	15.655	2.406	3.811	1.347	15.507	0.249
7	1850-06-0	13.15	0.614	18.946	2.817	7.106	0.857	16.353	0.245
8	1850-07-0	14.492	0.614	19.233	2.84	8.014	0.786	16.783	0.238
9	1850-08-0	14.039	0.802	18.477	2.079	7.406	1.086	16.718	0.28
10	1850-09-0	11.505	0.675	15.846	2.692	4.533	1.798	15.886	0.254

For this project we used 3 csv file and 1 json. This are –

- 1) GlobalTemperatures.csv
- 2) GlobalLandTemperaturesByCountry.csv
- 3) GlobalLandTemperaturesByState.csv
- 4) Countries.geo.json

The GlobalTemperatures.csv contains :

- Date: starts in 1750 for average land temperature and 1850 for max and min land temperatures and global ocean and land temperatures
- LandAverageTemperature: global average land temperature in celsius
- LandAverageTemperatureUncertainty: the 95% confidence interval around the average
- LandMaxTemperature: global average maximum land temperature in celsius
- LandMaxTemperatureUncertainty: the 95% confidence interval around the maximum land temperature
- LandMinTemperature: global average minimum land temperature in celsius
- LandMinTemperatureUncertainty: the 95% confidence interval around the minimum land temperature

- LandAndOceanAverageTemperature: global average land and ocean temperature in celsius
- LandAndOceanAverageTemperatureUncertainty: the 95% confidence interval around the global average land and ocean temperature

We just take Date and LandAverageTemperature for our analysis.

The GlobalTemperaturesByCountry.csv contains :

- Date : Same as Date column of GlobalTemperatures.csv.
- AverageTemperature : Same as LandAverageTemperature column of GlobalTemperatures.csv.
- Country : Name of countries from which the data contains.

The GlobalTemperaturesByCountry.csv contains :

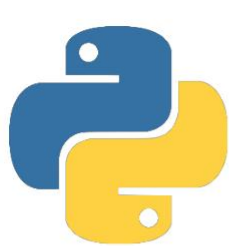
- Date : Same as Date column of GlobalTemperatures.csv.
- AverageTemperature : Same as LandAverageTemperature column of GlobalTemperatures.csv.
- Country : Name of countries from which the data contains.
- State : Name of state from which the data contains.

Countries.geo.json : This is geo json file contain the longitude and latitude of all countries. This is used to generate choropleth maps using plotly.

Software and Tools :

Here is the list of software and tools used in this project implementation.

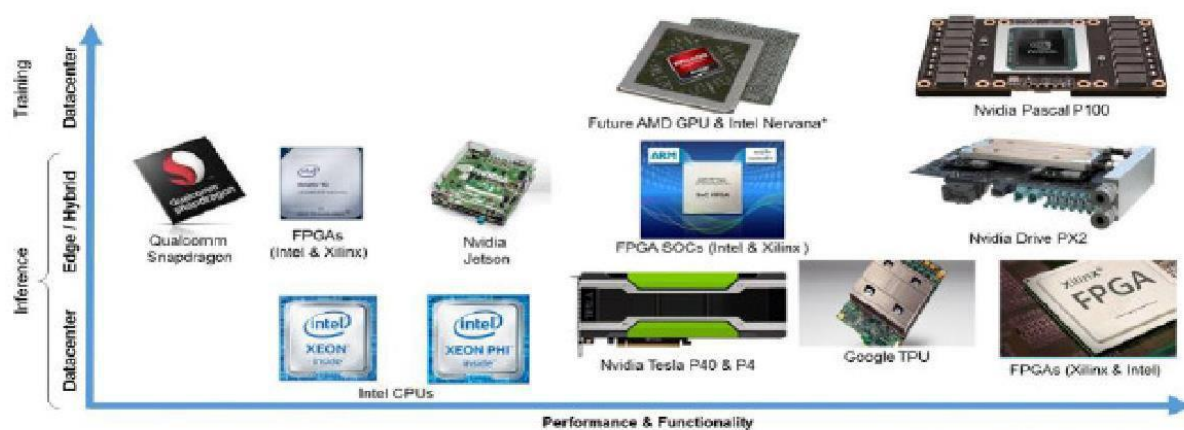
- Python : used as the primary programming and scripting language.
- Jupyter Notebook : For python scripting, data analysis and research.
- Spyder : For application backend Programming.
- Pandas : used for data frame manipulation.
- NumPy : For array manipulation.
- Plotly Express : for interactive plots.
- Neural Prophet : For Deep Learning modelling and forecasting.
- Streamlit : For frontend development.
- Html / CSS : For frontend improvement.
- Git : For project version control.



Hardware :

Deep learning is a very CPU intensive program-esque thing to be running, so be prepared to shell out a lot of money for a good enough system. Here are some system requirements to adhere to -

- Quad core Intel Core i7 Skylake or higher (Dual core is not the best for this kind of work, but manageable)
- 16GB of RAM (8GB is okay but not for the performance you may want and or expect)
- M.2 PCIe or regular PCIe SSD with at least 256GB of storage, though 512GB is best for performance. The faster you can load and save your applications, the better the system will perform. (SATA III will get in the way of the system's performance)
- Premium graphics cards, so things with GTX 980 or 980Ms would be the best for a laptop, and 1080s or 1070s would be the best for the desktop setup. (try not to sacrifice too much here. While a 980TI or a 970m may be cheaper, this is also a critical part of the system, and you'll see a performance drop otherwise.



Research and Methodology

The whole project methodology can be divide into 3 major part –

- Data Collection and EDA
- Data Modelling and Validation
- Web Application Development
- Deployment

Data Collection and EDA :

Data Collection : Data used in this project already discussed before. The datasets presented here have been divided into three categories: Output data, Source data, and Intermediate data. The Berkeley Earth averaging process generates a variety of Output data including a set of gridded temperature fields, regional averages, and bias-corrected station data. Source data consists of the raw temperature reports that form the foundation of our averaging system. Source observations are provided as originally reported and will contain many quality control and redundancy issues. Intermediate data is constructed from the source data by merging redundant records, identifying a variety of quality control problems, and creating monthly averages from daily reports when necessary. The definitive repository for Source and Intermediate data is located in the SVN, which is built nightly. This data was collected from <http://berkeleyearth.org/data/>

Exploratory Data Analysis : Analyze the change in temperatures across globe from the 17th century till now.

This Exploratory Data Analysis contains,

- 1) Load and Show Dataset
- 2) Missing Values Imputation
- 3) Lag Plot and Analysis
- 4) ACF or Autocorrelation plots and Analysis
- 5) Trend Plot and Analysis
- 6) Seasonal Plot and Analysis
- 7) Chloropleth Map of Average Temperature by Countries and Analysis

Here is all about the EDA :

Load and Show Dataset

Loading Dataset.....

Dataset Loaded Successfully.

First 5 Records of Global Land Average Temperature (1750 - 2015

) AverageTemperature

dt

1750-01-01 3.034

1750-02-01 3.083

1750-03-01 5.626

1750-04-01 8.490

1750-05-01 11.573

Last 5 Records of Global Land Average Temperature (1750 - 2015)

AverageTemperature

dt

2015-08-01 14.755

2015-09-01 12.999

2015-10-01 10.801

2015-11-01 7.433

AverageTemperature**dt****2015-12-01 5.518****Missing Values Imputation :****Strategy :** Impute missing Values with Seasonal (Monthly) mean

Number of Missing Values before imputation : 12

Imputing missing Values.....

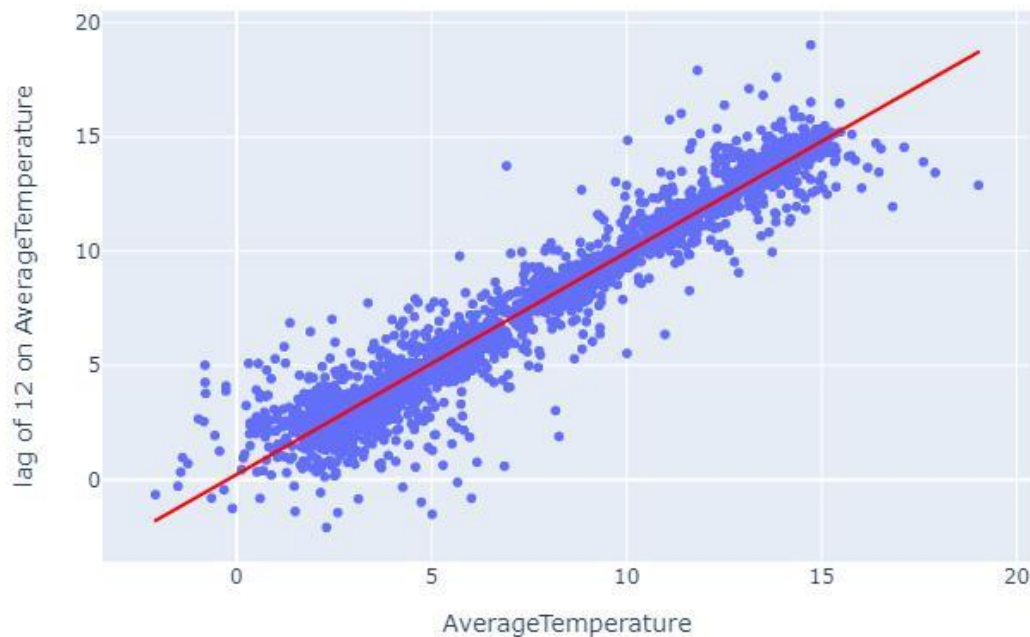
Number of Missing Values after imputation : 0

Lag Plot :

A lag plot is a special type of scatter plot in which the X-axis represents the dataset with some time units behind or ahead as compared to the Y-axis. The difference between these time units is called lag or lagged and it is represented by k.

Here is the lag plot on Global Land Average Temperature with K = 12. This show a Scatter plot with the Same Month Average Temperature of Previous Year

Lag Plot of lag 12



As per the above plot, Average temperature of a Month is highly Correlated with Average temperature of same Month of Previous year. That means if Average temperature of Previous Year July Month was around 15 degree celcius, then This Year Average temperature of July Month would be around 15 degree celcius.

The above plot only describe the Lag correlation for $K = 12$. Now Let visualize an Autocorrelation Plot to identify correlation for upto 100 lags.

ACF or Autocorrelation plots :

Autocorrelation plots are a commonly used tool for checking randomness in a data set. This randomness is ascertained by computing autocorrelations for data values at varying time lags.

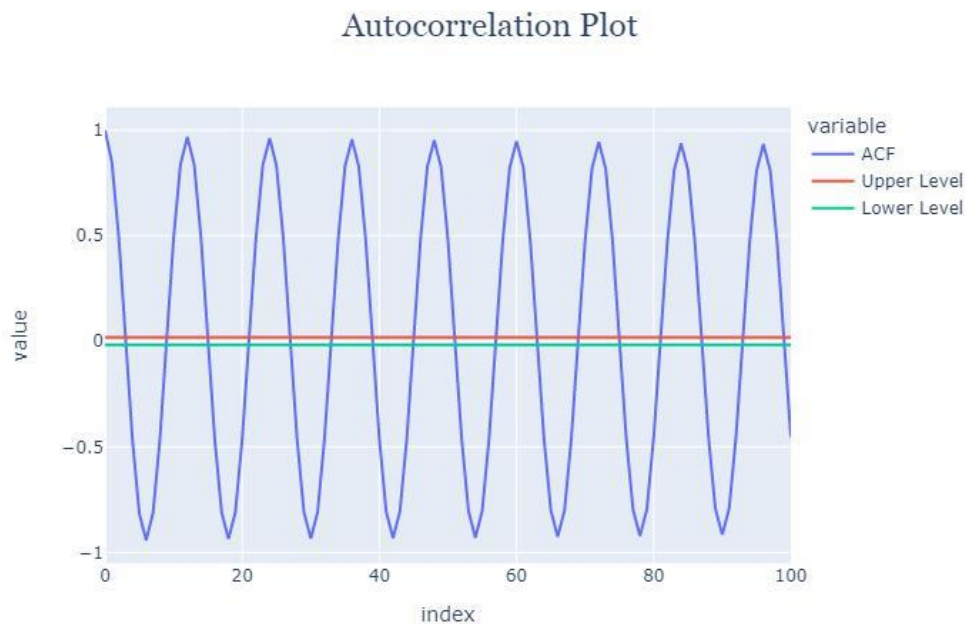
It measures a set of current values against a set of past values and finds whether they correlate.

It is the correlation of one-time series data to another time series data which has a time lag.

It varies from +1 to -1.

An autocorrelation of +1 indicates that if time series one increases in value the time series 2 also increases in proportion to the change in time series 1.

An autocorrelation of -1 indicates that if time series one increases in value the time series 2 decreases in proportion to the change in time series 1.



As per the Autocorrelation plot -

Lag with 12 has highest autocorrelation with correlation of 0.96

For every lags with multiple of 12 has very high autocorrelation (>0.9), but it decreasing with 0.005 (almost) per increase of +12 lags.

Lag with 6 has highest negative autocorrelation with correlation of -0.94

For every lags with multiple of 12 after the 6 lags has very high negative autocorrelation (<-0.9), but it decreasing with 0.004 (almost) per increase of +12 lags.

Lags with 1 and $(12i + 1)$ or $(12i - 1)$ has enough high autocorrelation with correlation around 0.8

Lags with 1 and $(12i + 1)$ or $(12i - 1)$ after the 6 lags has enough high negative autocorrelation with correlation around -0.8

Trend Plot :

A trend Graph is a graph that is used to show the trends data over a period of time. It describes a functional representation of two variables (x , y). In which the x is the time-dependent variable whereas y is the collected data.

Moving Averages : In statistics, a moving average is a calculation to analyze data points by creating a series of averages of different subsets of the full data set. It is also called a moving mean or rolling mean and is a type of finite impulse response filter.

Here is the Trend Plot using Moving Average of 10 Years :

Trend Plot with Moving Average of 10 Years



As per the above Trend plot -

from 1750 to now, the trend looks upward

Hence, Global Warming and Temperature Rising happens and

From 1800 to 1813, Temperature Falls exponentially from 8.55 to 7.19 Celcius. Then

From 1813 to 1970 Temperature Rises from 7.19 to 8.61 Celcius. Then

From 1970 to 2010 Temperature Rises exponentially from 8.61 to 9.58

Celcius just within this 40 years, which is almost equals to 1 degree celcius as per every 10 Years Interval Average.

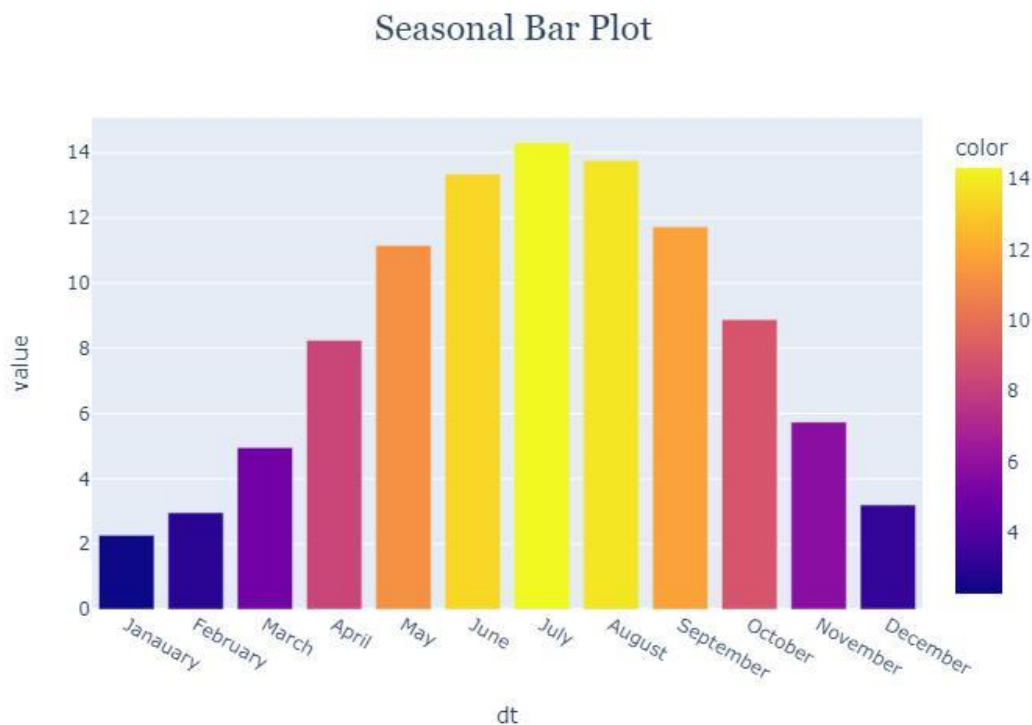
Seasonal Plot :

What Is Seasonality?

Seasonality is a characteristic of a time series in which the data experiences regular and predictable changes that recur every calendar year. Any predictable fluctuation or pattern that recurs or repeats over a one-year period is said to be seasonal.

A **seasonal plot** is similar to a time plot except that the data are plotted against the individual “seasons” in which the data were observed. This plot visualize the seasonality of the given Time Series.

Here is the Seasonal Bar Plot to visualize the Average Temperature Per Month -



As per the above Seasonal Bar Plot -

July is the hottest month followed by August, June. This months are belongs to Global Summer Season and

January is the Coldest followed by February and December. This months are belongs to Global Winter Season

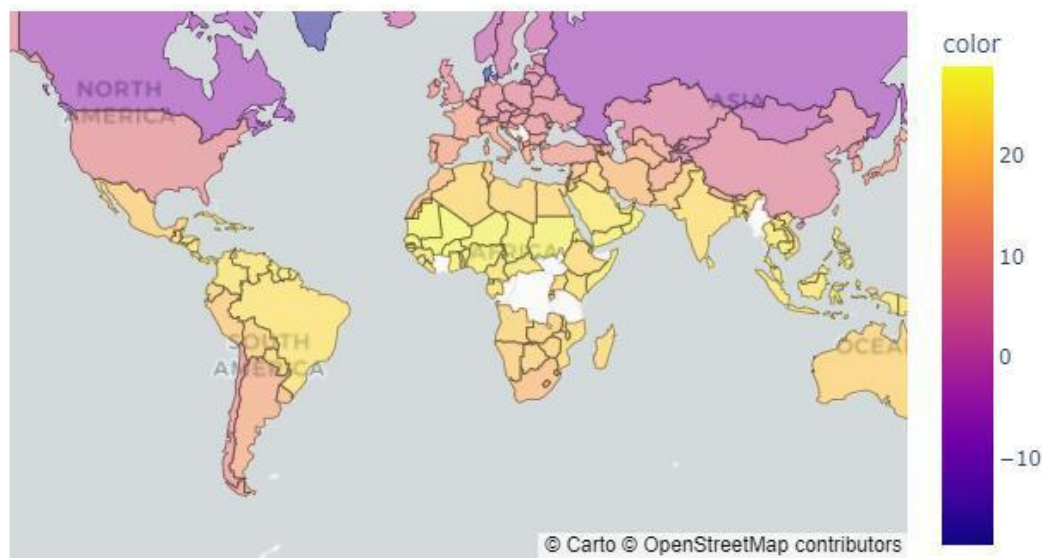
March, April, May after Winter and September, October, November after Summer are generally falls under the category of not very hot, not very cold.

Choropleth Map :

Choropleth Maps display divided geographical areas or regions that are coloured, shaded or patterned in relation to a data variable. This provides a way to visualise values over a geographical area, which can show variation or patterns across the displayed location.

Here is the **Chloropleth Map of Average Temperature by Countries** -

Chloropleth Map of Average Temperature by Countries



As per the above Chloropleth Map,

Countries near to the Equator are hottest Most Countries (about more than 20 degree celcius Yearly Average Temperature). India and South Asian Countries, Brazil and Middle of Both America, Arabia, North African Countries falls under this categories.

Countries near to the Pole are Coldest Most Countries (about less than 0 degree celcius Yearly Average Temperature). Russia, Canada and Greenland falls under this category.

USA, China, Japan, Europe, Korea, Kazakhstan like countries are falls under the category of not very hot, not very cold. Yearly Average Temperature ranges in this countries generally from 5 to 12 degree celcius.

Data Modelling and Validation :

Select, train and Validate a best Time Series Model to forecast Average Average Temperatures per month for next years.

This part contains,

- 1) Import All Required Libraries
- 2) Load and Show Dataset
- 3) Missing Values Imputation
- 4) Data Stationarity Check
- 5) Time Series Modeling : Neural Prophet
- 7) Preparing dataset for NeuralProphet Training
- 8) Splitting Dataset for training and validation
- 9) Model Training
- 10) Model Training History and Visualization
- 11) Model performance analysis on Validation Data
- 12) Demo forecasting for next 2 years

Load and Initialize Data

Loading Dataset.....

Dataset Loaded Successfully.

AverageTemperature

dt

1750-01-01 3.034

1750-02-01 3.083

1750-03-01 5.626

1750-04-01 8.490

1750-05-01 11.573

3192 rows × 1 columns

Missing Values Imputation :

Strategy : Impute missing Values with Seasonal (Monthly) mean

Number of Missing Values before imputation : 12

Imputing missing Values.....

Number of Missing Values after imputation : 0

Data Stationarity Check

Strategy : Augmented Dickey Fuller test (ADF Test)

P Value : 0.00447

Stationary

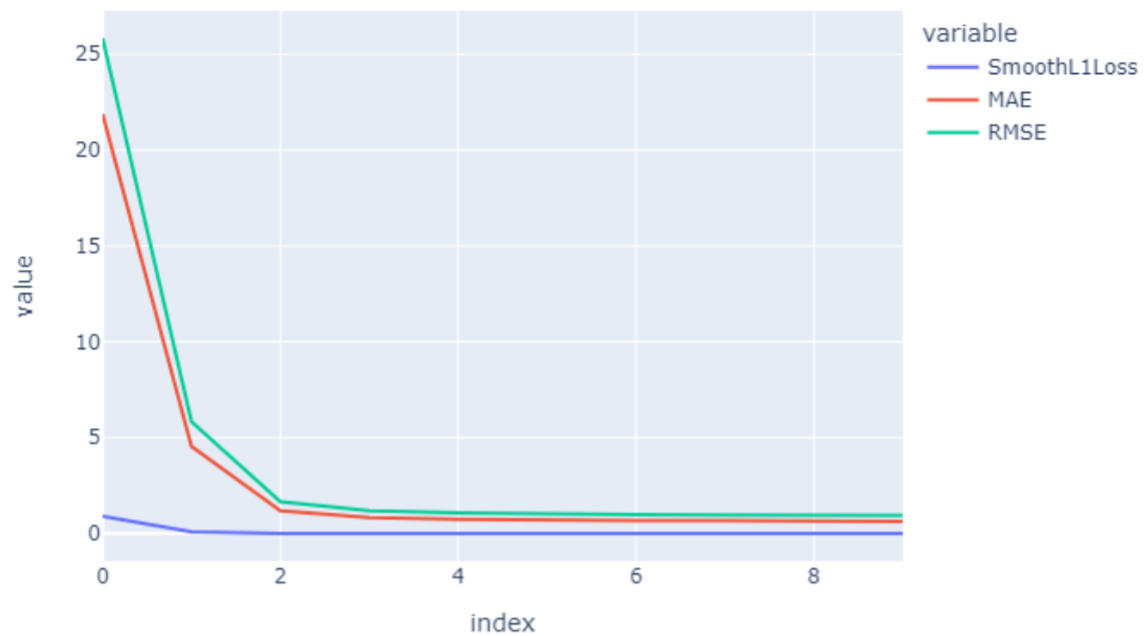
As per the ADF test, dataset is already stationary and it is ready fit a Time Series Model.

Time Series Modeling : Neural Prophet**What is Neural Prophet ?**

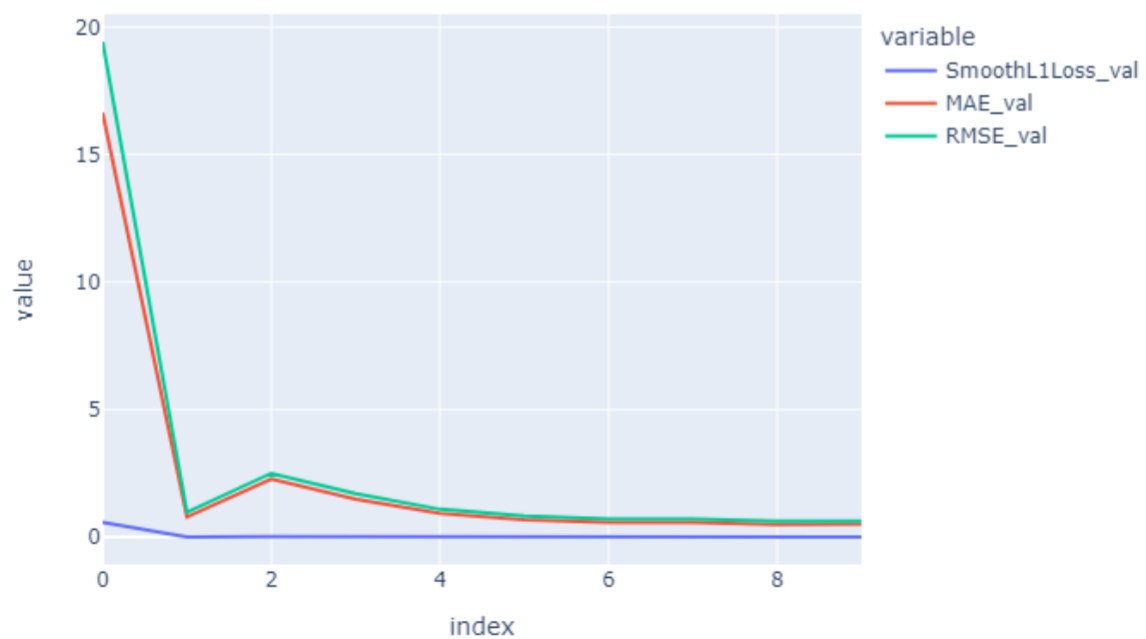
NeuralProphet is an upgraded version of Prophet that is built using PyTorch and uses deep learning models such as AR-Net for time-series forecasting. The main benefit of using NeuralProphet is that it features a simple API inspired by Prophet, but gives you access to more sophisticated deep learning models for time-series forecasting.

Visualization of Model Training History

Loss per Epochs on Training Data



Loss per Epochs on Validation Data



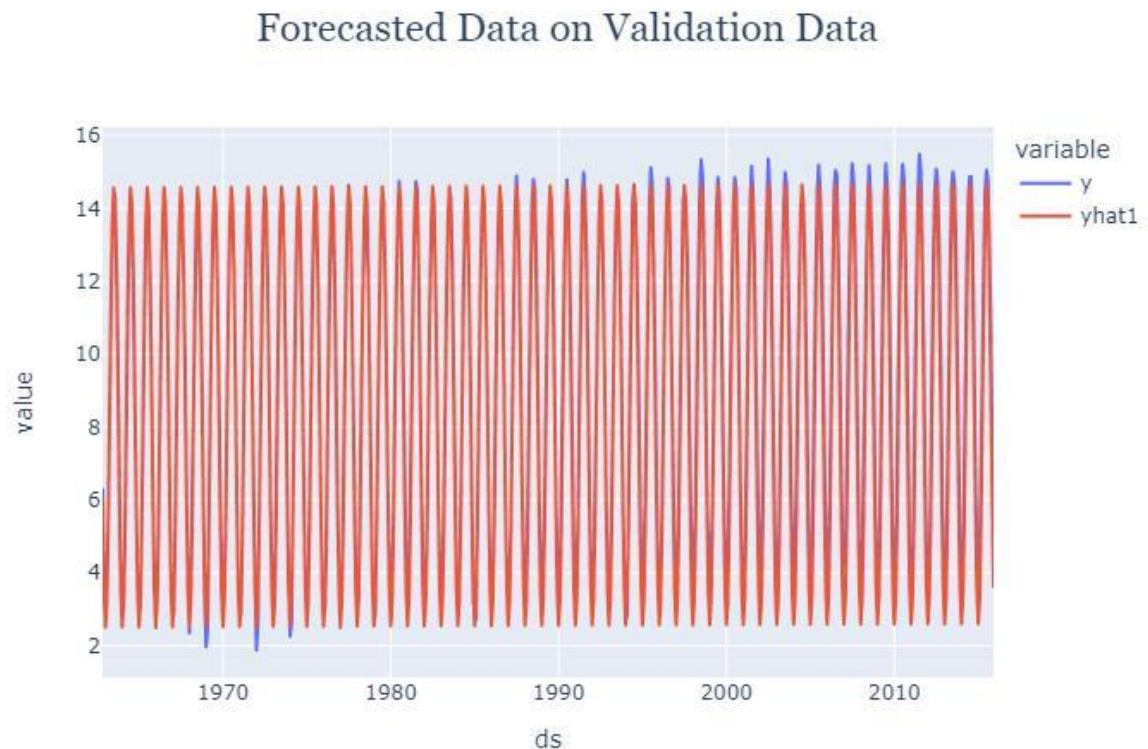
Model Performance Analysis

Testing model on validation data

Generate forecast as per dates of Validation dataset

638 rows \times 6 columns

Visualization forecasted Data on top of Real Observed Validation data



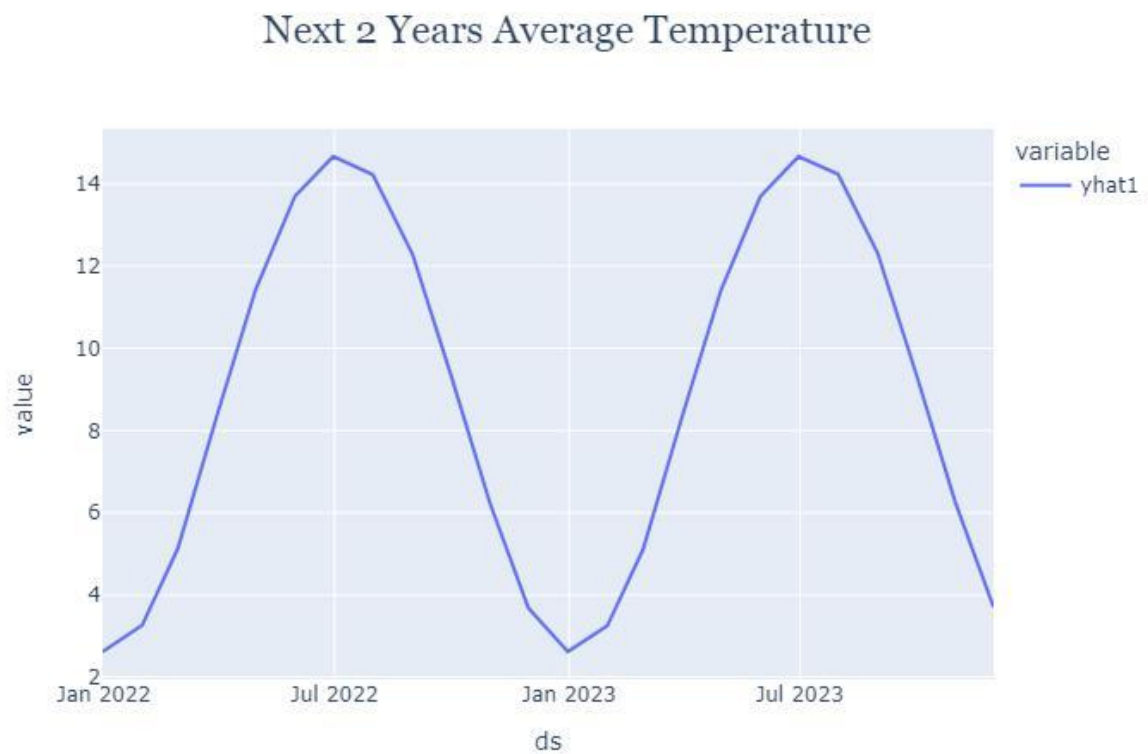
RMSE Error and R2 Score :

RMSE : 0.6221

R2 Score : 0.9776

As per the RMSE and R2 Score, the model trained perfectly and enough good for deployment.

Forecasting for Next 2 Years and Visualizaion of Forecasted Data



Web Application Development :

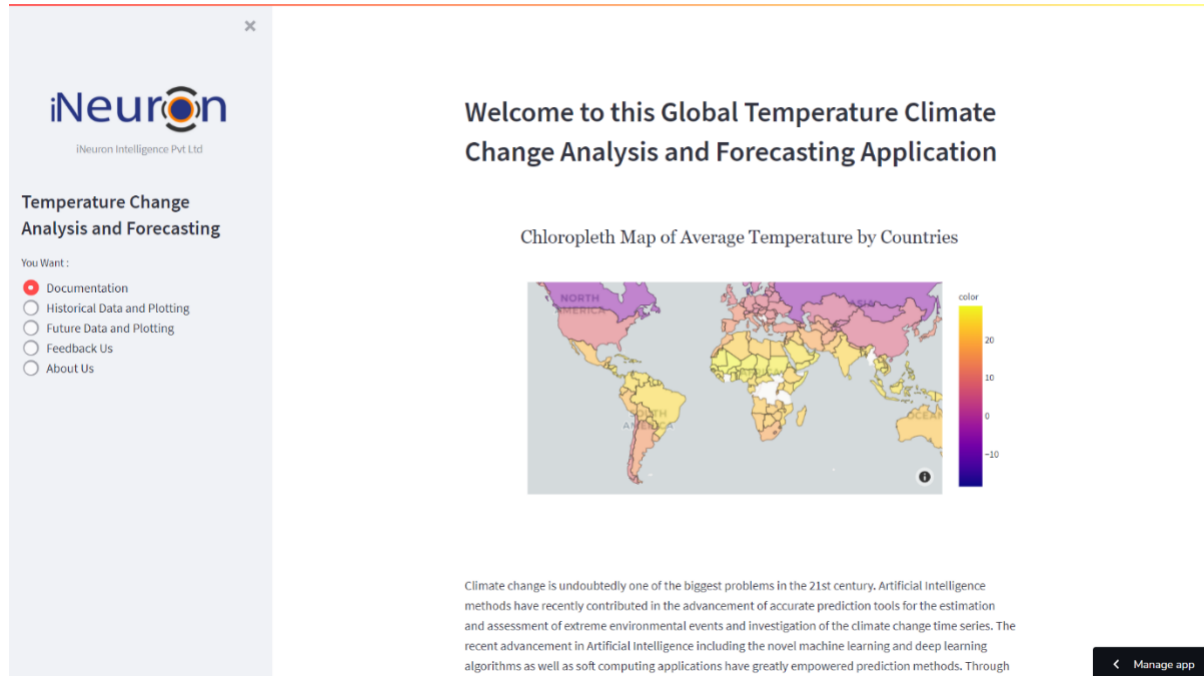
As per the proposed solution is a Multi Paged Web Application. To develop this frontend application, Streamlit is used powered by HTML / CSS. The whole coding for this frontend development is done on Spyder IDE.

Here 5 different web pages developed inside the application. This are –

- Documentation
- Historical Data and Plotting
- Future Data and Plotting
- Feedback Us
- About Us

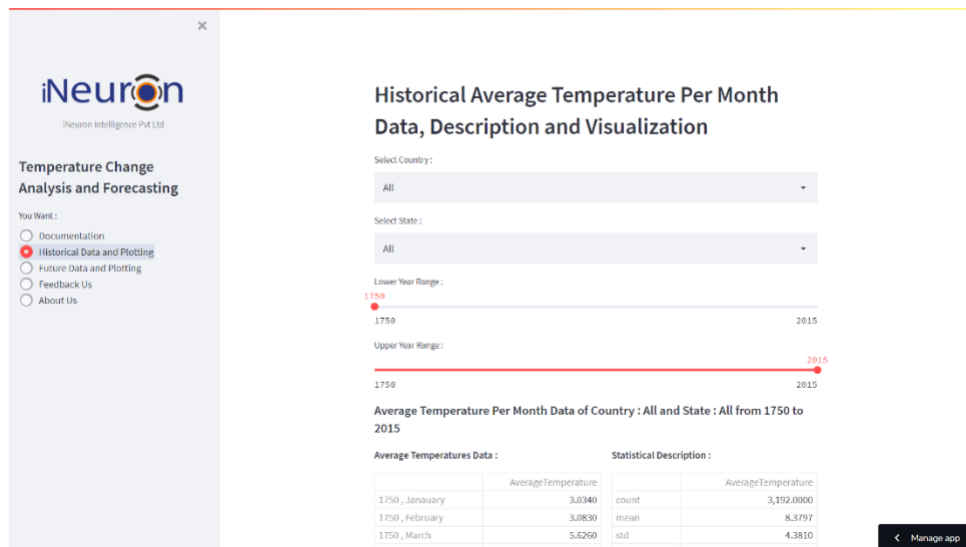
Here the details about the web pages along with screenshots –

Documentation Page :



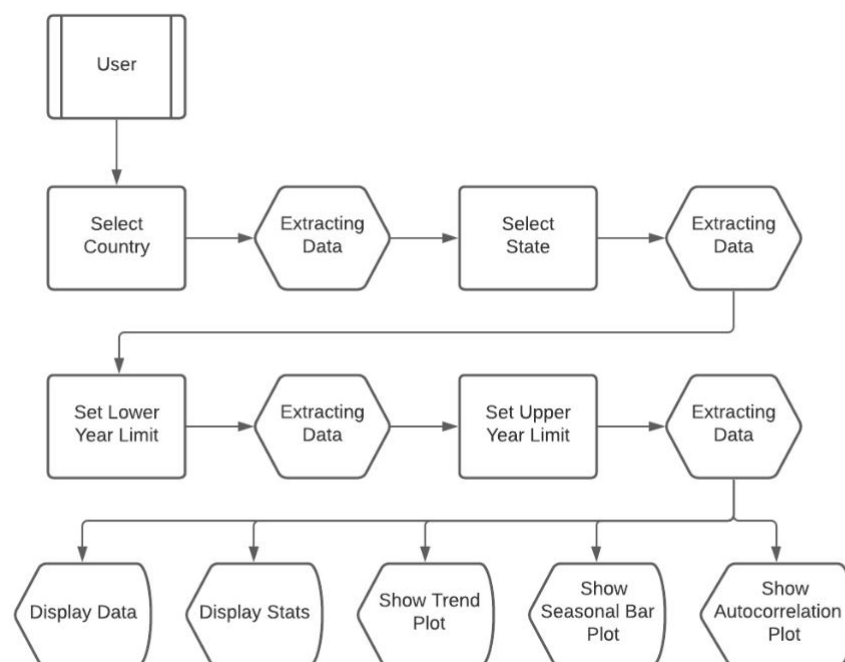
This page will be the default page when user opens the url. This page contains an abstract or a short documentation to describe what the application is and what are the pages it contains.

Historical Data and Plotting :

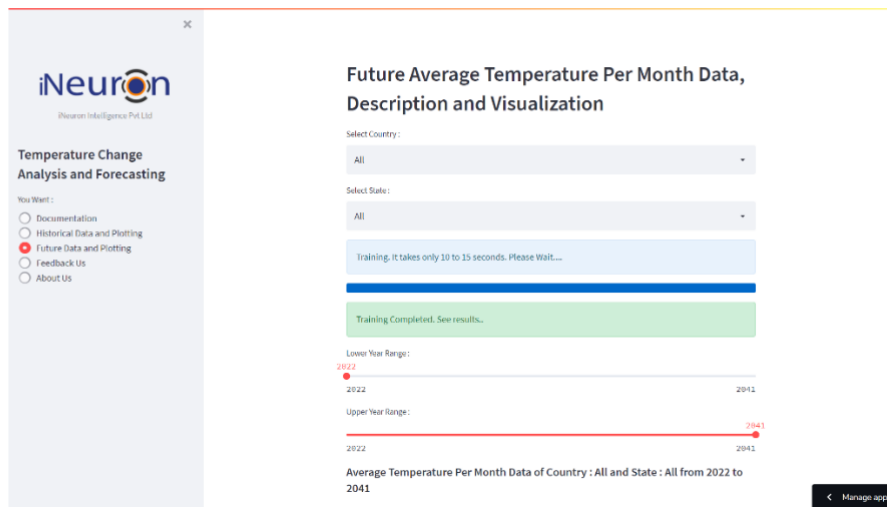


Through this page , user can –

- 1) Get Historical Average Temperatures Per Month upto past 200 years as per the choice of Country, State and Time Range selected.
- 2) Able to download the same data as filtered.
- 3) View the interactive plots to analyze and conclude the Historical Temperature trend, Seasonality and Autocorrelation on the filtered Country, State and Time Range.

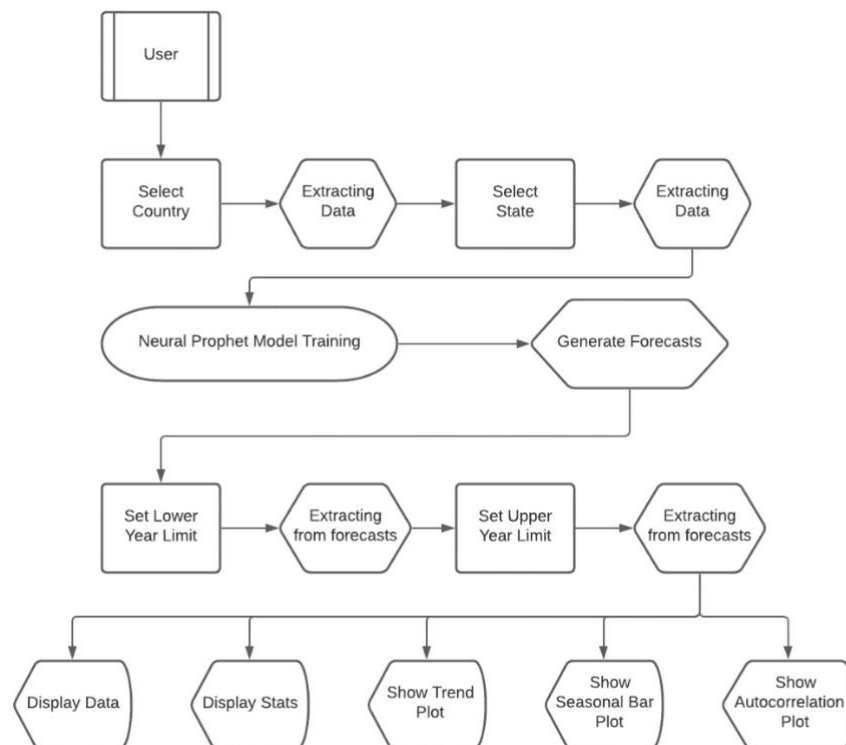


Future Data and Plotting :

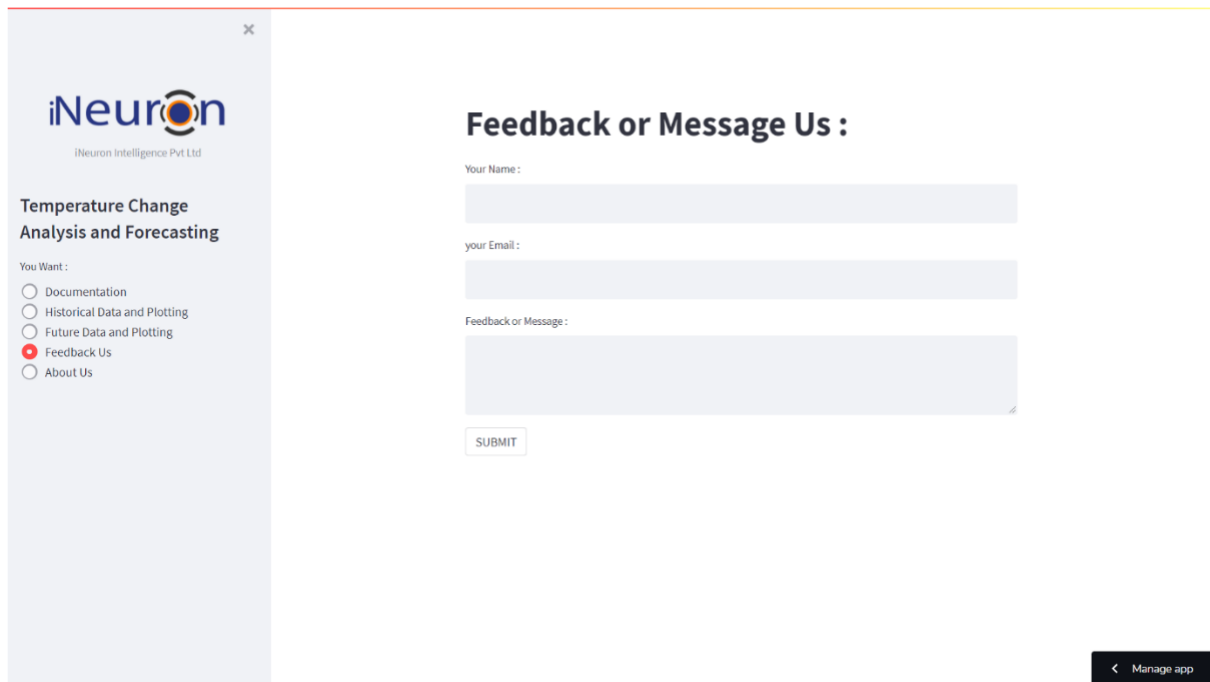


Through this page , user can –

- 1) Get Historical Average Temperatures Per Month upto past 200 years as per the choice of Country, State and Time Range selected.
- 2) Able to download the same data as filtered.
- 3) View the interactive plots to analyze and conclude the Historical Temperature trend, Seasonality and Autocorrelation on the filtered Country, State and Time Range.



Feedback Us :



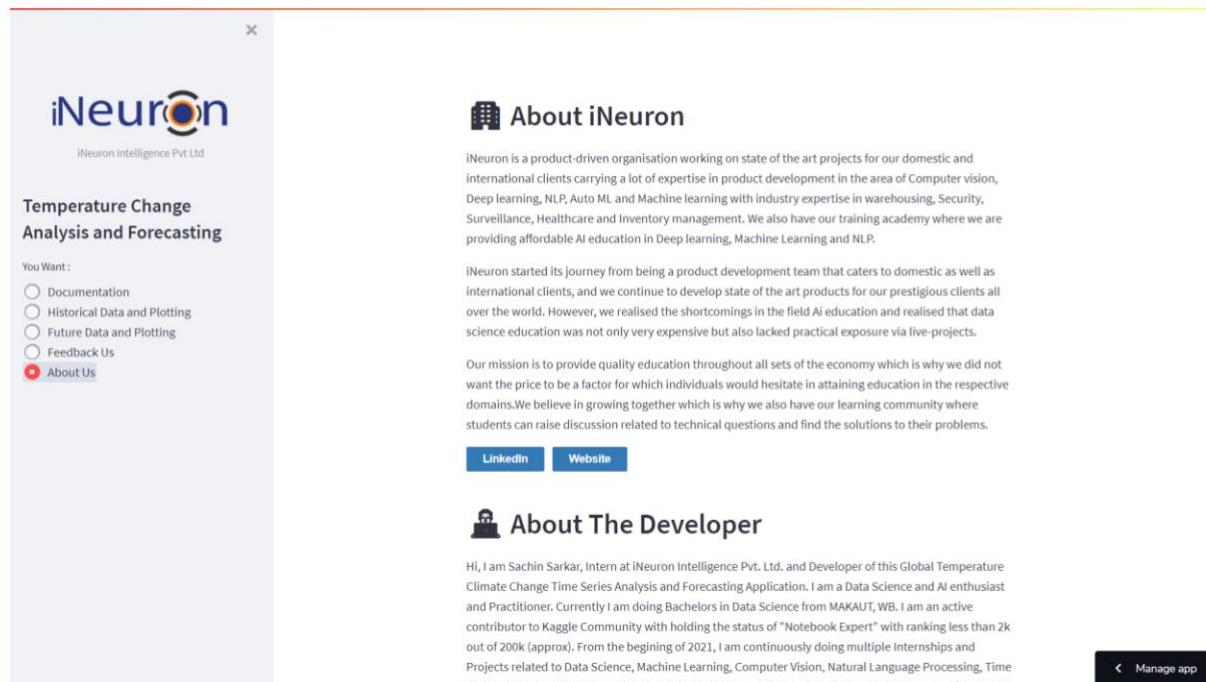
The screenshot shows a mobile application interface for iNeuron. On the left is a sidebar menu with the iNeuron logo and the text 'iNeuron Intelligence Pvt Ltd'. Below the logo, it says 'Temperature Change Analysis and Forecasting'. Under 'You Want:', there are four radio button options: 'Documentation', 'Historical Data and Plotting', 'Future Data and Plotting', 'Feedback Us' (which is selected with a red dot), and 'About Us'. The main content area has the heading 'Feedback or Message Us :'. It contains three input fields: 'Your Name :', 'your Email :', and 'Feedback or Message :'. Below these fields is a 'SUBMIT' button. In the bottom right corner of the app screen, there is a dark button with a left arrow and the text 'Manage app'.

This is for communication. User can share feedback or messages to developer through this page.

This page contains –

- A heading of the page.
- Text input to take user name.
- Text input to take user email id.
- Text Area input to take message or feedback from user .

About Us :



This page will shown about iNeuron and about developer.

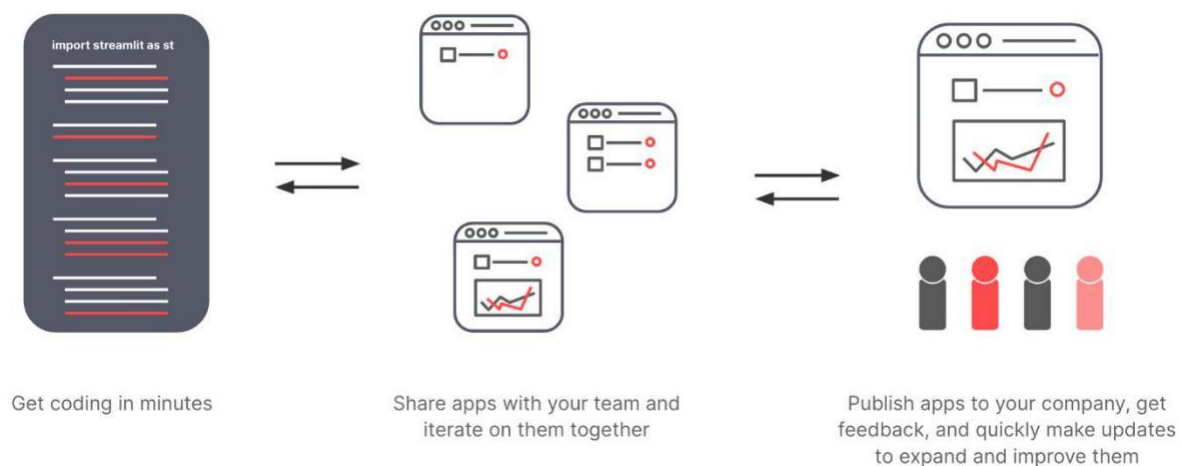
This page content –

- About iNeuron.
- About the Developer

Deployment :

Streamlit Cloud used for deployment this application.

When you work on an app in Streamlit Cloud—be it a new model, data analysis, or idea—you're just a few clicks away from securely sharing it and collaborating on it with your team.



1. Build and deploy apps in minutes

Build Streamlit apps the way you've always built them! Download the open-source library, use your favorite IDE, and take advantage of Streamlit's run-on-save rapid development flow. Done building? Use Streamlit Cloud to watch your app go *live* across the company.

Streamlit Cloud handles all the IT, DevOps, and security for you—Python dependencies, Unix package management, container orchestration, server provisioning, scaling, data security, and more—so you can get back to your data work!

Deploy an app

Apps are deployed directly from their Github repo. Enter the location of your app below.

Repository [Paste GitHub URL](#)

streamlit-apps/repo

Branch

master

Main file path

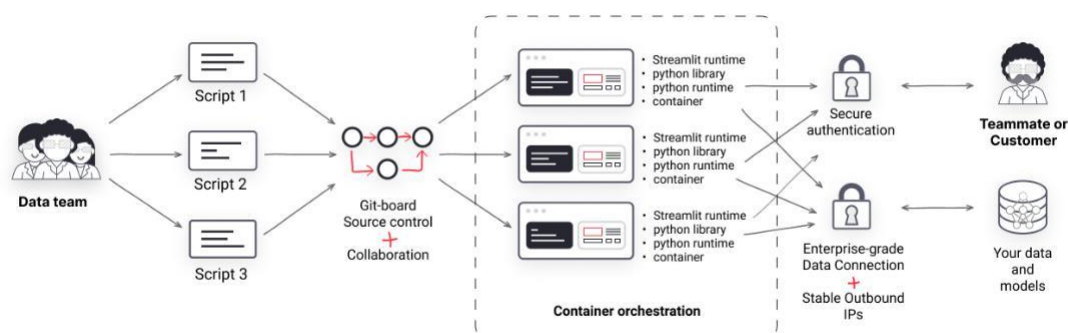
streamlit_app.py

[Deploy!](#)

2. Securely share apps

Once app is deployed, you can *securely* share it with your whole company. Send it to just one person. Or send it to teammates, customers, and other business units so they can start using it right away.

Streamlit Cloud works with your preferred SSO provider. Easily lock down your app so only certain people can see it.



3. Rapidly iterate

Your app is shared! Now you can quickly iterate on it. Streamlit Cloud continuously deploys your app from GitHub, giving you the power of modern version-controlled code development.

Like a teammate's app? Fork it and launch your own! Want to test out a new version of a production app? Branch it and deploy a dev app. Prefer an earlier version? Check the version history and roll it back. Received a user request? Make the change and watch the app automatically update.

All of this makes for an incredibly rapid prototype-to-production cycle for your whole team.

Conclusion

Undoubtedly, this application will help users to analyze past and future climatic trends and changes. As well as users can get to know the monthly seasonality and autocorrelation. Users can able to filter this insights as per there choice of country, state and time range.