CLIMATE CHANGE PREDICTION TEMPRATURE DETECTION

## A PROJECT REPORT

***Submitted by***

# V.SRIVATSAN M.RAGHAVENDRAN

**S.DEEPAK**

***in partial fulfillment for the award of the degree of***

# BACHELOR OF ENGINEERING

**IN**

**COMPUTER SCIENCE AND ENGINEERING**



**TAGORE ENGINEERING COLLEGE,KANCHEEPURAM**

# ANNA UNIVERSITY : CHENNAI 600 025

## JUNE 2022

**ANNA UNIVERSITY : CHENNAI 600 025**

# BONAFIDE CERTIFICATE

Certified that this project report **“CLIMATE CHANGE PREDICTION AND TEMPRATURE DETECTION”** is the bonafide work of “V.SRIVATSAN, M.RAGHAVENDRAN, S.DEEPAK**”** who carried out the project work under my supervision.

### SIGNATURE SIGNATURE

**DR.S.SURENDRAN.M.E.P.H.D. DR. B. JaiShanthi , M.E,P.H.D**

### HEAD OF THE DEPARTMENT, SUPERVISOR

**Professor Assistant Professor,**

### Department of CSE Department of CSE

**Tagore engineering college, Tagore engineering college,**

### Rathinamangalam, Rathinamangalam,

**Vandalur, Chennai-600127 Vandalur, Chennai-600127**

### Submitted for Project Report viva voice held on .

**INTERNAL EXAMINAR EXTERNAL EXAMINAR**

# ACKNOWLEDGEMENT

My heartfelt thanks go to **Prof. M. Mala., M. A, M. Phil**, Chairperson of Tagore Engineering College, Rathinamangalam , for having provided us with all necessary infrastructure and other facilities, for their support to complete this project successfully.

We extend our sincere gratitude to. **Dr. R. Ramesh M.E., Ph. D, FIE, MIETE, MISTE**., Principal Tagore Engineering College for this degree of encouragement and moral support during the course of this project.

We are extremely happy for expressing my heart full gratitude to my department **HOD Dr. S. SURENDRAN, M.E., Ph.D**. for his valuable suggestions which helped us to complete the project successfully.

Our sincere gratitude to my supervisor **DR.B.JAI SHANTHI. B.E,M.E,P.H.D** for extending all possible help for this project work.

We thank to our project coordinator **DR.B.JAI SHANTHI. B.E,M.E,P.H.D** for her valuable suggestions which helped us to complete the project successfully.

Our sincere thanks to all teaching and non-teaching staff who have rendered help during various stage of our work.

# ABSTRACT

Temperature changes in last few years have been increasing tremendously and it is expected to increase more in the future. Therefore, it is a tedious process to analyze larger forms of climate, temperature changes in the data and perform predictive analysis of the same using traditional methods. Our project aims to forecast temperature and seasonality changes using predictive analysis with the help of various machine learning techniques such as Linear Regression, Time series forecasting using ARIMA & SARIMAX. The proposed system serves as a tool which takes in the changes in temperature from large amount of data as input and predicts the future temperature with min, max, and average temperature in an efficient manner and approximately produces the output. Predictive analytic models capture relationships among various factors in a data set to assess risk with a particular set of conditions to assign a score or a weight. These patterns of score/weight found in historical data can be used for predicting the future temperature and seasonality changes.

# TABLE OF CONTENTS

|  |  |  |  |
| --- | --- | --- | --- |
| **CHAPTER NO.** | **TITLE** | **PAGE NO.** | |
|  | **ABSTRACT** | **iii** | |
|  | **LIST OF TABLE** | **xvi** | |
|  | **LIST OF FIGURES** | **xvii** | |
|  | **LIST OF SYMBOLS** | **xxvii** | |
|  |  |  | |
| 1. | **INTODUCTION**  1.1GENERAL |  | |
| 2. | **LITERATURE STUDY** |  | |
| 3. | **PROBLEM DEFINITION**  EXISTING SYSTEM  PROPOSED SYSTEM |  | |
| 4. | **SYSTEM REQUIREMENTS**    4.1 HARDWARE REQUIREMENTS  4.2 SOFTWARE REQUIREMENTS | |  |
| 5. | **SYSTEM ARCHITECTURE** |  | |
| 6. | **SYSTEM DESIGN** |  | |
| 7. | **MODULES DESCRIPTION** |  | |
| 8. | **SYSTEM TESTING** |  | |
| 9. | **CONCLUSION AND FUTURE ENHANCEMENT** |  | |
|  | **APPENDIX 1** |  | |
|  | **SOURCE CODE** |  | |
|  | **SCREEN SHOTS** |  | |
|  | **REFERENCES** |  | |

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **FIGURE NO.** | **NAME** | **PAGE**  **NO.** |
| **1** | **Temperature change by year month and week** |  |
| **2** | **Features of Python** |  |
| **3** | **Temperature box plot** |  |
| **4** | **Temperature change** |  |
| **5** | **Temperature distribution curve based on density change** |  |
| **6** | **Temperature distribution curve** |  |
| **7** | **Temperature forecast changes** |  |
| **8** | **Temperature graph using ADF Test Statistics** |  |
| **9** | **Time of execution graph** |  |
| **10** | **Correlation and Autocorrelation** |  |

# LIST OF ABBREVIATIONS

**CHAPTER 1 INTRODUCTION**

Climate change is a crucial challenge in the recent era. It affects many factors of the environmental ecosystems such as, soil erosion, bio-diversity, and changes in sea-water level

.Weather forecasting mitigates the economic crisis and promotes better public health to maintain the quality of life. Safety and well-being of human is highly implacable by weather changes. It is also useful in the agricultural domain as it is an essential part of planning the farming operations. Farmers can make optimal decisions for crops using prediction of weather whether to undertake or withhold the sowing operation. The consequences of unseasonal changes in weather and their potential negative effects on host plants and pests are very well known. Unseasonably high temperatures may lead to lower plant productivity and more pests on farm. Industries such as energy consumption and food security can also benefit from weather forecasting. As the key problem of weather forecasting, air temperature prediction has manifold benefits for the environment, industry and agriculture. The impact of temperature on morbidity and mortality can be assessed at both the seasonal and daily level. Extreme temperature changes Due to harsh environment, arises the lack of access to safe water and food, it can also cause Heat-aggravated and respiratory illness. Prediction of the energy consumption, soil surface temperature and solar radiation is related to ambient air temperature forecasting. Air temperature forecasting is useful in understanding the probability of storm, wildfires, drought and flood occurrence in an area. . This leads to an incomplete understanding of the atmospheric processes, so it restricts weather prediction up to a 10 day period, because beyond that weather forecasts are significantly unreliable. But Machine learning is relatively robust to most atmospheric disturbances as compared to traditional methods. Another advantage of machine learning is that it is not dependent on the physical laws of atmospheric processes.

## 1.1 Temperature prediction:

Temperature prediction is an infamously sophisticated and resource consuming task. Temperature changes are caused by many factors. Parameterization of those features is a difficult task to achieve due to their dynamic nature. Recent development in the field of artificial intelligence can help provide less computationally expensive solutions. We can approximate the forecasts using several black box methods without a need of extensive mathematical calculations by analyzing historical temperature data. The purpose of this thesis

was to analyze current research on weather forecasting and compare machine learning techniques in the field. Various inputs to these machine learning models were also tested to determine the usefulness of each, as measured by their contribution to lowering the difference between predicted values and the ultimate observed ground truth. Machine Learning algorithms have been widely used for complicated data. Pattern analysis and recognition of temperature data can be simplified with use of machine learning algorithms. Air temperature data is classified as part of time series statistics up. Hence, use of Linear regression, Auto correlation algorithms to estimate the future value of temperature seems a plausible solution. The purpose of this thesis was to analyze current research on weather forecasting and compare machine learning techniques in the field. Various inputs to these machine learning models were also tested to determine the usefulness of each, as measured by their contribution to lowering the difference between predicted values and the ultimate observed ground truth. It was found that including weather forecast data in the prediction models resulted in a 7.6% reduction in mean absolute error (MAE) for one-hour predictions when compared to using historical observations alone, and a 40.2% reduction in MAE for 24-hour predictions. Results from several machine learning techniques were compared, with Random Forests achieving the lowest error rate. In addition, inclusion of weather forecasts from nearby areas resulted in a 4.6% lower mean absolute error (MAE) in one-hour predictions and a 20.9% reduction in 24-hour predictions when averaged across the five cities studied.

Chapter 3 analyzes the effect of incorporating the historical biases of weather predictions

in order to improve them, a widely-used method known as postprocessing. Due to the existence of persistent, statistically-significant time- and location-based biases, nearly all current numerical weather forecasting systems apply some form of postprocessing to their raw input data (Marzban, 2003). While linear regression Model Output Statistics (MOS) is commonly used in practice, there is a minimal number of current studies comparing its results to those achievable by more advanced models. This thesis compared several different machine learning techniques and found that an ensemble model stacking a Random Forest with an artificial neural network (ANN) was found to reduce prediction error over MOS on seven of the eight weather variables studied (air temperature, cloud cover, visibility, wind speed, wind direction, dew point temperature, air pressure, and humidity). The inclusion of additional forecasted weather variables from areas immediately surrounding the target location was not found to have an impact on prediction error, a contrast to the solar radiation prediction results found in Chapter 2.

**2.1 Related Work**

# CHAPTER-2 LITERATURE REVIEW

In our experiment, we came across various machine and deep learning methodologies. An early paper of this domain by B. Ustaoglu [9], tests three different kind of ANN based methods:(1) feed-forward back propagation (FFBP), (2) radial basis function (RBF) and, (3) generalized regression neural network (GRNN). It compares the answers with traditionally used multiple linear regression (MLR),Auto-correlation, Time series forecasting using ARIMA AND SARIMAX model they obtained notable improvements over MLR outputs. A paper written in 2015 by Z. Karevan [10] describes a black box idea: use of Machine learning methods such as k-NN and Elastic Net for the process of feature selection then trains model using Support Vector Machine Regression with Least Square loss function to predict minimum and maximum temperature. After 3 years, R. Isabelle used Recurrent Convolutional Neural Networks for weather forecasting and visualization [11] where they propose use of convolution filters +LSTMs. Their results were found substantially better in comparison with popular methods. Another approach was used by P. Hewage [4], where they used multiple features like temperature, pressure, wind, humidity, precipitation and moisture to predict future value of thesame feature. This was executed by implementing several machine learning and deep learning algorithms like TCN [13], LSTM with multi-input multi- output and multi-input single-output methods . S.Kendzierski used a novel approach by implementing Jordan Pi-Sigma Neural Network (JPSN) for time series data, introduced by N. Husaini [14]. In this paper they combined two methodology: Jordan Neural Network, Pi- Sigma Neural Network to predict the temperature. The MSE of the model is remarkably low, but model does not satisfy the criteria suggested by A. Kumar [3]—the performance of the model can be acceptable if: NMSE ≤ 0.5. Satellite images can also be used as model inputs; when combined with estimates for clear-sky solar radiation, these forecasts can be quite effective (Miller et al., 2013; Linares- Rodriguez et al., 2013). An alternative to these satellite-based approaches are methods using Numerical Weather Prediction (NWP) for regional solar radiation prediction (Lara-Fanego et al, 2012; Perez et al., 2013). Ruiz-Arias, Quesada-Ruiz, Fernández, and Gueymard (2015) suggest several advantages for NWP methodologies over satellite-based imagery, including the fact that NWP models comprehensively simulate the entire atmospheric system, including wind, temperature, humidity, and other variables. For forecast horizons extending beyond 6 hours, NWP models

tend to outperform those without forecasts (Cheung et al., 2015). The results found in these previous works indicate that present- and forward-looking approaches such as satellite images and forecasts of multiple weather variables are important components to predicting solar radiation, as opposed to using historical solar radiation observations alone. This paper will attempt to further quantity their contribution.

**Table 2.2** :Detail-Metrics Of Related Work

|  |  |  |  |
| --- | --- | --- | --- |
| **Reference** | **Input variables** | **Algorithm** | **Error Metrics** |
| [9] | temperature | ANN | RMSE = 1 |
| [10] | wind, snow, rain,fog, MIN, AVG and  MAX temperature,wind speed | k-NN + Elastic Net + LS-SVM | MAE = 1.15 |
| [11] | temperature,  pressure, wind directions (2D) | CNN+LSTM(RCNN) | MAE = 0.88 |
| [4] | temperature,pressure,  wind,humidity, | TCN, MIMO-LSTM,  MISO-LSTM | MSE = 3.4 |
| [15] | temprature | JPSM | MSE=0.00642 |

## PRELIMINIRIES:

Accurate prediction of temperature requires knowledge of various parameters: longitude, latitude, sea level, pressure, wind, precipitation; and their internal correlations. Ecosystem of some parameters complete within small areas and for some-other, it occupies large areas. As consequences, it becomes difficult to define a region to accurately estimate those parameters. Due to this difficulty, parameterization becomes a vital part of the process. Parameterization is a complex and instantiate process. In the sense of computational liability, it is very resource consuming procedure. In different regions, temperature patterns may vary. However, it is generally repetitive with respect to time. Hence, with help of recent developments in the field of machine learning and deep learning, we can eliminate use of parameterization. In this thesis, we have used black box methods described in subsection 3.1 and 3.2 to forecast temperature using past temperature data points. Our aim is to create an accurate procedure which analyzes the patterns of past temperature data to predict future results.

## DATA ACQUISITION:

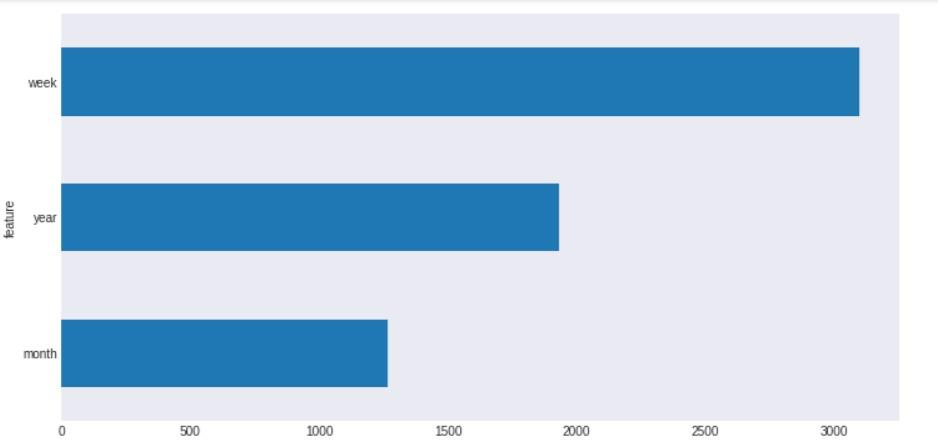
The Georgia Automated Environmental Monitoring Network (GAEMN) was developed in 1990 to develop a climatic data base used for various agricultural, ecological, water

management, and other environmental-based research (Hoogenboom, Verma, & Threadgill,

1990). A variety of weather- and environment-related data including air and soil temperature, barometric pressure, wind speed and direction, rainfall, and solar radiation is collected in 15- minute intervals across the state of Georgia. This GAEMN data was analyzed for the time period of 2003 to 2013 for five cities in Georgia. The data included 43 observed fields, although only a handful were proven relevant for solar radiation prediction. In addition to these historical data observations, this paper makes use of weather forecasts provided by The National Oceanic and Atmospheric Administration's (NOAA). This data is disseminated in the GRIB file format, a compact binary format commonly used to store historical and forecasted weather data due to its self-description and flexibility (World Meteorological Organization, 2003). Each GRIB file describes a particular geographical region for a single date, and internally splits this region into a grid of cells of a consistent size. For each cell, attribute values are listed describing weather attributes in the cell at that time, or, in the case of weather forecasts, at a specified time in the future. As a final note, for the purposes of this paper a "forecast" will always reference a NOAA weather attribute forecasted variable as obtained through GRIB data or an interpolation, whereas a "prediction" will reference a prediction generated for a future temperature change value.

## DATA VISUALIZATION:

As temperature change measures the amount of change received based on the earth surface, its value is highest during the early afternoon hours and drops to near-zero values between sunset and sunrise.



**Fig 1:Temperature change by year month and week**

The second significant source of seasonality in solar irradiance corresponds to the time of year. Due to the variation of tilt of the earth's axis during summer and winter months, solar radiation varies significantly throughout the year, peaking in June.

### 2.4: . Uncertainties and Methods in Climate Change Prediction:

If we had a perfect climate system model, a perfect observing system and a perfect knowledge of the statistical behavior of the external forcings, this added uncertainty would be eliminated and the uncertainty in the prediction would essentially reflect the stochastic and non-linear nature of the climate change problem. It is thus important to understand and assess the contributions of different sources of uncertainty in a climate change prediction, so that “bottleneck steps” in the prediction process can be identified and improved.

# CHAPTER-3 PROBLEM DEFINITION

## Problem Statement

Efforts to understand the influence of historical climate change, at global and regional levels, have been increasing over the past decade. In particular, the estimates of air temperatures have been considered as a key factor in climate impact studies on agricultural, ecological, environmental, and industrial sectors. Mitigating climate change is one of the biggest challenges of humankind. Despite the complexity of predicting the effects of climate change on earth, there is a scientific consensus about its negative impacts. Among them, the affectation of ecosystems, decrease of biodiversity, soil erosion, extreme changes in temperature, sea level rise, and global warming have been identified. Likewise, impacts on the economy, human health, food security and energy consumption are expected. Specifically, air temperature forecasting has been a crucial climatic factor required for many different applications in areas such as agriculture, industry, energy, environment, tourism, etc. Some of these applications include short-term load forecasting for power utilities, air conditioning and solar energy systems development, adaptive temperature control in greenhouses, prediction and assessment of natural hazards, and prediction of cooling and energy consumption in residential buildings. Therefore, there is a need to accurately predict temperature values because, in combination with the analysis of additional features in the subject of interest, they would help to establish a planning horizon for infrastructure upgrades, insurance, energy policy, and business development. [source of information: mdpi]

## Motivation:

In the motivated attention framework, we define political motivation as political orientation, and we predict that liberals and conservatives attend to the same climate change evidence (i.e., a graph of global temperature) in ways that are consistent with their political norms. Attention is measured by eye gaze dwell time on the graph. We define perception of climate evidence as the estimation of global temperature from the graph, and actions to mitigate climate change as the likelihood to sign climate petitions or donate to an environmental organization. To seek evidence for this framework, we conducted three experiments to examine how people with different political orientations perceive the same global temperature graph and whether the perceptual differences can be explained by different

attentional priorities (Experiment 1), how these attentional biases alter actions to mitigate climate change, and how drawing attention to motivationally consistent evidence influences climate actions

## 3.1.2 Experiment 1:

This experiment examines how political motivation alters the perception of climate change evidence. We predict that people with different political orientations perceive the same temperature graph differently when the graph is framed as global temperature, but not when the graph is under a neutral frame (i.e., when the evidence is not motivationally relevant). We further examine whether the perceptual differences can be explained by different attentional allocations on the graph. We tracked visual attention using an eyetracker in the lab while participants were viewing the graph. We predict that liberals and conservatives focus on different parts of the graph consistent with their political motivations to guide their temperature estimation.

## 3.1.3 Product Scope:

We have implemented and machine learning model through which it will display the change in climate, seasonality, temperature among state and city accurately. This product is designed for PC, laptops and all devices which are compatible with python. The users of this product can use the services of this model when they are connected to the web as data is retrieved and stored in excel as data and we have process the data and produce the output in terms as graph.

**Existing System**

Weather forecasting is the application of science and technology to predict the state of the atmosphere for a given location. Weather forecasts are made by collecting quantitative data about the current state of the atmosphere and using scientific understanding of atmospheric processes to project how the atmosphere will evolve. There are a variety of end users to weather forecasts. Weather warnings are important forecasts because they are used to protect life and property. In ancient times, forecasting was mostly based on weather pattern observation. Over the years, the study of weather patterns has resulted in various techniques for rainfall forecasting. Present rainfall forecasting embodies a combination of computer models, interpretation, and an acquaintance of weather patterns. The following technique was used for existing weather prediction.

### Use of a barometer:

Measurements of barometric pressure and the pressure tendency have been used in forecasting since the late 19th century. The larger the change in pressure, the larger the change in weather can be expected. If the pressure drop is rapid, a low pressure system is approaching, and there is a greater chance of rain [TF05].

### Looking at the sky:

Along with pressure tendency, the condition of the sky is one of the most important parameters used to forecast weather in mountainous areas. Thickening of cloud cover or the invasion of a higher cloud deck is an indication of rain in the near future. At night, high thin clouds can lead to halos around the moon, which indicates the approach of a warm front and its associated rain. Morning fog portends fair conditions, as rainy conditions are preceded by wind or clouds which prevent fog formation [KMQ10].

### Nowcasting:

The forecasting of the weather within the next six hours is often referred to as nowcasting. In this time range, it is possible to forecast smaller features such as individual showers and thunderstorms with reasonable accuracy, as well as other features too small to be resolved by a computer model. A human, given the latest radar, satellite and observational data will be able to make a better analysis of the small scale features present and so will be able to make a more accurate forecast for the following few hours [RR03]. Analog technique The analog technique is a complex way of making a forecast, requiring the forecaster to remember a previous weather event which is expected to be mimicked by an upcoming event. It remains a

useful method of observing rainfall in places such as oceans, as well as the forecasting of precipitation amounts and distribution in the future. A similar technique is used in medium range forecasting, which is known as

teleconnections, when systems in other locations are used to help pin down the location of another system within the surrounding regime[Dj75].

### Numerical Weather Prediction model :

Numerical Weather Prediction (NWP) is the science of predicting the weather using models of the atmosphere and computational techniques. Current weather conditions are used at the input of the mathematical models of the atmosphere to predict the weather. This model usually provides surrounding point around the wind farm with a spatial resolution of a few kilometers. NWP uses the power of computers to make a forecast. A forecaster examines how the features predicted by the computer will interact to produce the day's weather. The NWP method is flawed in that the equations used by the models to simulate the atmosphere are not precise [Ry02]. A number of weather forecasting agencies operate modeling centers where supercomputers are used to run NWP models that span the entire globe. These include the National Center for Environmental Prediction (NCEP) in the United States, the United Kingdom Meteorological Office (UKMO), and the European Centre for Medium-range Weather Forecasts (ECMWF). Although costly, a global approach to NWP is essential, especially for long-range forecasting. For this reason, achieving accurate forecasts requires an accurate analysis from which to get the model started. This involves a computer-based process called data assimilation, in which the most recent weather observations from around the world are combined with model forecasts to create a global analysis of current conditions. This becomes the starting point for the next run of the NWP model, and is the computer equivalent of the manual analysis cycle that forecasters carry out on an on-going basis. Global models play a key role in modern weather forecasting, and meteorologists at Met Service routinely use the NCEP, UKMO and ECMWF models to assist with day-to-day production of forecasts and weather warnings. These models give insight into the behavior of weather systems on a large scale, without much emphasis on local detail [FS78].

### Ensemble Forecasting:

To predict the weather forecast meteorologists have developed atmospheric models that approximate the atmosphere by using ensemble forecasting to describe how atmospheric temperature, pressure and moisture will change over time. The equations are programmed into a computer and the data on the present atmospheric conditions are fed into the computer. The computer solves the equations to determine how the different atmospheric variables will

change over the next few minutes. The computer repeats this procedure again and again using the output from one cycle as the input for the next cycle. For some desired time in the future, the computer prints its calculated information. It then analyzes the data, drawing the lines for the projected position of the various pressure systems. A forecaster uses the prognostic chart as a guide to predicting the weather. There are many atmospheric models that represent the atmosphere, with each one interpreting the atmosphere in a slightly different way. Weather forecasts made for 12 and 24 hours are typically accurate. Forecasts made for two or three days are usually good. Beyond above five days, forecast accuracy falls off rapidly. Weather information can also come from remote sensing, particularly radar and satellites.

### Radar:

Radar stands for Radio Detection and Ranging. In radar, a transmitter sends out radio waves. The radio waves bounce off the nearest object and then return to a receiver. Weather radar can sense many characteristics of precipitation, its location, motion, intensity, and the likelihood of future precipitation. Most weather radar is Doppler radar, which can also track how fast the precipitation falls. Radar can outline the structure of a storm and in doing so estimates the possibility that it will produce severe weather condition [NCW12].

### Weather satellites:

Weather satellites have been increasingly important sources of weather data since the first one was launched in 1952. Weather satellites are the best way to monitor large scale systems, like storms. Satellites can also monitor the spread of ash from a volcanic eruption, smoke from fires, and pollution.

### Weather Maps:

Weather maps simply and graphically depict meteorological conditions in the atmosphere. Weather maps may display only one feature of the atmosphere or multiple features. They can depict information from computer models or from human observations. Weather maps are found in newspapers, on television, and on the Internet. On a weather map, each weather station will have important meteorological conditions plotted. These conditions may include temperature, current weather, dew point, cloud cover, sea level air pressure, wind speed and direction. On a weather map, meteorologists use many different symbols. These symbols give them a quick and easy way to put information onto the map. The empirical approach is based upon the occurrence of analogues and is often referred to by meteorologists as analogue forecasting. This approach normally is useful for predicting local-scale weather if recorded cases are plentiful. Dynamical approach is based upon equation of the atmosphere and is often referred to as computer modeling.

**Proposed System**

The main objective of this study is to analyze temperature and seasonality change in chennai (temperature and seasonality changes like winter, spring, summer) for approx. 10-20 years on the basis data recorded at different place across Tamil Nadu and we predict the temperature changes from 2020-2024.Camparing to existing work we can predict the temperature change more accurately.

The specific objectives are:

1. To predict the temperature and seasonality changes for next 10 years.
2. To make a graphical interface based on the prediction to make it easy understand. Machine Algorithms: There are so many technologies to predict data like SVM, linear regression, lasso, elastic net, light gm regressor, Auto Regressive Model, Time series analysis using Arima and Sarimax etc. We have tried many of the algorithm to get the highest accuracy. All the methods have different working procedure. The working procedure of some of those predicting techniques are discussed below:

### Linear Regression:

Linear Regression is a method which gives a relationship between a dependent variable or scalar variable and an independent variable or explanatory variable. In this method the relationships are model using linear predictor function. Here the data is trained by this method. Linear predictor function is used to make an object of that function and used it for prediction. After creation of the object, the data is forecasted for future.

### Support Vector Regression:

Support Vector Regression is one part of Support Vector Machine. SVR follows the same principle which is followed by SVM. For support vector regression, the prediction method is difficult comparative to other methods. The algorithm is more complicated. Among all the technologies more accuracy is observed in Linear Regression. For that in this paper Linear Regression is used. The complexity on Linear Regression is also comparatively much lesser than the other technologies.

### Time Series Forecasting using ARIMA and SARIMAX:

ARIMA, short for ‘Auto Regressive Integrated Moving Average’ is actually a class of models that ‘explains’ a given time series based on its own past values, that is, its own lags and the lagged forecast errors, so that equation can be used to forecast future values.

Any ‘non-seasonal’ time series that exhibits patterns and is not a random white noise can be modeled with ARIMA models.

An ARIMA model is characterized by 3 terms: p, d, q where,

* + p is the order of the AR term
  + q is the order of the MA term
  + d is the number of differencing required to make the time series stationary

If a time series, has seasonal patterns, then you need to add seasonal terms and it becomes SARIMA, short for ‘Seasonal ARIMA’.

### Elastic Net:

Elastic Net first emerged as a result of critique on lasso, whose variable selection can be too dependent on data and thus unstable. The solution is to combine the penalties of ridge regression and lasso to get the best of both worlds. Elastic Net aims at minimizing the following loss function: where α (alpha) is the mixing parameter between ridge (α = 0) and lasso (α = 1). There are some modules which are required to develop the Global Warming Prediction System. Those modules are briefly explained below:

### Data Collection:

In this module the raw is collected data from different data set. Then the data set is changed as per need. This raw data cannot be predicted directly. So, it is needed to clean and pre- process.

### Data Pre-processing :

In this module the data is cleaned. After cleaning of the data, the data is grouped as per requirement. This grouping of data is known as data clustering. Then check if there is any missing value in the data set or not. It there is some missing value then changes it by any default value. After that if any data need to change its format, it is done. Total process before the prediction is known is data pre-processing. After that the data is used for the prediction and forecasting step.

### Data Prediction and forecasting:

In this step, the pre-processed data is taken for the prediction. This prediction can be done in any process which are mentioned above. But the Linear Regression algorithm scores more prediction accuracy than the other algorithm. So, in this project the linear regression method

is used for the prediction. For that, the pre-processed data is splitted for the train and test purpose. Then a predictive object is created to predict the test value which is trained by the trained value. Then the object is used to forecast data for next few years.

### Visualization:

In this step, the predicted and forecasted data is used to provide a graphical interface separately. At first the predicted data is plotted in a graph separately with the help of matplot library. Then the forecasted data of temperature is plotted in graph with proper scale. Then the greenhouse gases forecasted data are plotted in a single graph with proper scale.

### Auto Regressive Model:

**Autocorrelation and Partial Autocorrelation**

* + Identification of an AR model is often best done with the PACF.
    - For an AR model, the theoretical PACF “shuts off” past the order of the model. The phrase “shuts off” means that in theory the partial autocorrelations are equal to 0 beyond that point. Put another way, the number of non-zero partial autocorrelations gives the order of the AR model. By the “order of the model” we mean the most extreme lag of x that is used as a predictor.
    - Look for sudden drop
  + Identification of an MA model is often best done with the ACF rather than the PACF.
    - For an MA model, the theoretical PACF does not shut off, but instead tapers toward 0 in some manner. A clearer pattern for an MA model is in the ACF. The ACF will have non-zero autocorrelations only at lags involved in the model.
    - Look for exponential drop link code

### Parameters for ARIMA

1. p - Autoregressive (AR) Model Lags - Use PACF
2. d - No. of times differencing performed
3. q - Moving Average (MA) Lags - Use ACF

## CHAPTER - 4 SYSTEM REQUIREMENTS SPECIFICATION

**4.1 SOFTWARE REQUIREMENTS:**

**Technology :** Python

**Project Technologies :** Machine learning models trained and tested successfully

**Python Version :** Python 3.10

**Libraries Used :** Numpy , Pandas, Matplotlib, Seaborn, Sklearn

**SDLC Model Used :** Agile Methodology

## 4.2 HARDWARE REQUIREMENTS:

**Processor :** i5 10th gen/ Ryzen 5 5300 h

**RAM :** 8gb

**Hard disk :** 1 TB

**SSD :** 512 gb

**FDD :** 1.44MB

**Display :** 15.6 inches

**Mouse :** 3 Button scroll

**CD Drive** : 52 X

**Keyboard :** 108 keys

## 4.3 Technology Used:

**4.3.1 Introduction to python:**

## Introduction:

**What is Python??**

Python is a high-level object-oriented programming language that was created by Guido van Rossum. It is also called general-purpose programming language as it is used in almost every domain we can think of as mentioned below:

* Web Development
* Software Development
* Game Development
* AI & ML
* Data Analytics

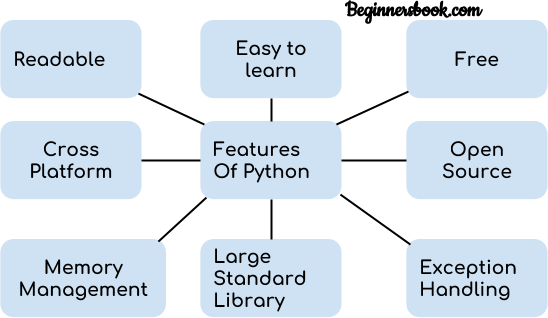
## Why Python Programming?

Every Programming language serves some purpose or use-case according to a domain. for eg, Javascript is the most popular language amongst web developers as it gives the developer the power to handle applications via different frameworks like react, vue, angular which are used to build beautiful User Interfaces. Similarly, they have pros and cons at the same time. so if we consider python it is general-purpose which means it is widely used in every domain the reason is it’s very simple to understand, scalable because of which the speed of development is so fast. Now you get the idea why besides learning python it doesn’t require any programming background so that’s why it’s popular amongst developers as well. Python has simpler syntax similar to the English language and also the syntax allows developers to write programs with fewer lines of code. Since it is open-source there are many libraries available that make developers’ jobs easy ultimately results in high productivity. They can easily focus on business logic and Its demanding skills in the digital era where information is available in large data sets.

## 4.3.3 Applications of Python Programming:

1. **Web Development:** Python offers different frameworks for web development like Django, Pyramid, Flask. This framework is known for security, flexibility, scalability.
2. **Game Development:** PySoy and PyGame are two python libraries that are used for game development
3. **Artificial Intelligence and Machine Learning:** There is a large number of open-source libraries which can be used while developing AI/ML applications.
4. **Desktop GUI:** Desktop GUI offers many toolkits and frameworks using which we can build desktop applications.PyQt, PyGtk, PyGUI are some of the GUI frameworks.

## Features of Python programming language:



**Fig 2 : Features of python**

1. **Readable:** Python is a very readable language.
2. **Easy to Learn:** Learning python is easy as this is a expressive and high level programming language, which means it is easy to understand the language and thus easy to learn.
3. **Cross platform:** Python is available and can run on various operating systems such as Mac, Windows, Linux, Unix etc. This makes it a cross platform and portable language.
4. **Open Source:** Python is a open source programming language.
5. **Large standard library:** Python comes with a large standard library that has some handy codes and functions which we can use while writing code in Python.
6. **Free:** Python is free to download and use. This means you can download it for free and use it in your application. See: [Open Source Python License.](https://docs.python.org/3/license.html) Python is an example of a FLOSS (Free/Libre Open Source Software), which means you can freely distribute copies of this software, read its source code and modify it.
7. **Supports exception handling:** If you are new, you may wonder what is an exception? An exception is an event that can occur during program exception and can disrupt the normal flow of program. Python supports exception handling which means we can write less error prone code and can test various scenarios that can cause an exception later on.
8. **Advanced features:** Supports generators and list comprehensions. We will cover these features later.
9. **Automatic memory management:** Python supports automatic memory management which means the memory is cleared and freed automatically. You do not have to bother clearing the memory.

## Python standard library Used:

1. **Tensor Flow:** This library was developed by Google in collaboration with the Brain Team. It is an open-source library used for high-level computations. It is also used in machine learning and deep learning algorithms. It contains a large number of tensor operations. Researchers also use this Python library to solve complex computations in Mathematics and Physics.
2. **Matplotlib:** This library is responsible for plotting numerical data. And that’s why it is used in data analysis. It is also an open-source library and plots high-defined figures like pie charts, histograms, scatterplots, graphs, etc.
3. **Pandas:** Pandas are an important library for data scientists. It is an open-source machine learning library that provides flexible high-level data structures and a variety of analysis tools. It eases data analysis, data manipulation, and cleaning of data. Pandas support operations like Sorting, Re-indexing, Iteration, Concatenation, Conversion of data, Visualizations, Aggregations, etc.
4. **Numpy:** The name “Numpy” stands for “Numerical Python”. It is the commonly used library. It is a popular machine learning library that supports large matrices and multi- dimensional data. It consists of in-built mathematical functions for easy computations. Even libraries like TensorFlow use Numpy internally to perform several operations on tensors. Array Interface is one of the key features of this library.
5. **SciPy:** The name “SciPy” stands for “Scientific Python”. It is an open-source library used for high-level scientific computations. This library is built over an extension of Numpy. It works with Numpy to handle complex computations. While Numpy allows sorting and indexing of array data, the numerical data code is stored in SciPy. It is also widely used by application developers and engineers.
6. **Scikit-learn:** It is a famous Python library to work with complex data. Scikit-learn is an open-source library that supports machine learning. It supports variously supervised and unsupervised algorithms like linear regression, classification, clustering, etc. This library works in association with Numpy and SciPy.

### Google Colab

Google have released Colaboratory: **a web IDE for python**, to enable Machine Learning with storage on the cloud — this internal tool had a pretty quiet public release in late 2017, and is set to make a huge difference in the world of machine learning, artificial intelligence and data science work. Colab is a free notebook environment that runs entirely in the cloud. It lets you and your team members edit documents, the way you work with Google Docs. Colab supports many popular machine learning libraries which can be easily loaded in your notebook. Google is quite aggressive in AI research. Over many years, Google developed AI framework called TensorFlow and a development tool called Colaboratory. Today TensorFlow is open-sourced and since 2017, Google made Colaboratory free for public use. Colaboratory is now known as Google Colab or simply Colab. Another attractive feature that Google offers to the developers is the use of GPU. Colab supports GPU and it is totally free. The reasons for making it free for public could be to make its software a standard in the academics for teaching machine learning and data science. It may also have a long-term perspective of building a customer base for Google Cloud APIs which are sold per-use basis. Irrespective of the reasons, the introduction of Colab has eased the learning and development of machine learning applications. To be precise, Colab is a free Jupyter notebook environment that runs entirely in the cloud. Most importantly, it does not require a setup and the notebooks that you create can be simultaneously edited by your team members - just the way you edit documents in Google Docs. Colab supports many popular machine learning libraries which can be easily loaded in your notebook. As a programmer, you can perform the following using Google Colab- Write and execute code in Python, document your code that supports mathematical equations, Create/Upload/Share notebooks, Import/Save notebooks from/to Google Drive, Import/Publish notebooks from GitHub, import external datasets e.g., from Kaggle, Integrate PyTorch, TensorFlow, Keras, OpenCV and Free Cloud service with free GPU.

### CHAPTER - 5 SYSTEM ARCHITECTURE

The System architecture shows how the data mining, data visualization concept is applied on the Temperature detection . Whenever people get confused about what the weather and upcoming seasonality changes is we have implemented a system through which we can predict the upcoming temperature and seasonality changes with an approximation as results. The people who wants to know about the current temperature and season they can see this model which is being implemented. The temperature change is predicted from 1992-2024 through which we can see the minimum, maximum and average temperature in an particular month in an year and also we will be comparing the temperature change across various years to see how weather is increased, decreased through some past years 20 years . The temperature data set is taken. After taking the data, it is modified as per need and we train various machine learning models and see the graphs for comparing, contrast the data. In the raw data, the temperature of each month. Those data are converted into average temperature for better and ease usage. In the temperature data the average temperature of India is given. So that we can easily predict the average temperature of India by training those data and we can calculate the accuracy and prediction of those trained data and calculated the accuracy of the classified data. We predict the temperature based on the data which is cleaned and visualized and we develop and EDA and data preprocessing model which will process the data and predict the temperature in terms of statistical analysis of data which will be data s- driven activity and data mining Weather is important for most aspects of human life. Predicting weather is very useful. Humans have attempted to make predictions about the weather, many early religions used gods to explain the weather. Only relatively recently have humans developed reasonably accurate weather predictions. We decided to collect weather data and measured the accuracy of predictions made using linear regression. The Weather prediction model designed by us would be of great use to the farmers and for normal being as well. This model basically uses historical weather data to predict the weather on a specific day of and year in the future. Initially the aim is to teach the model

with large historical data set and then use it for weather prediction. The observations include:

* Temperature - the measure of warmth or coldness
* Humidity - the amount of moisture in the atmosphere
* Precipitation - the amount of moisture (usually rain or snow) which falls on the ground
* Wind Speed - the speed at which air flows through the environment
* Wind Direction – the direction in which the wind is moving
* Pressure - the force the atmosphere applies on the environment

For the comparison, we created two models: one model was trained on smoothed input

and other one was trained on unsmooth input. Both models had similar training duration which was approximately 20 seconds and both models had NMSE ≅ 0 which suggested the models were acceptable and significant. However, from the perspective of accuracy, the model created with unsmooth inputs, gave high error rates with MSE = 3.4 and MAE = 1.44. Meanwhile, performance of the proposed model which was train on smoothed inputs provided substantially low errors. The MSE of proposed model was about 0.064 and MAE =

0.202. Proposed method seems to give better performance of accuracy on test data, as shown in the figures. Our model outperformed other models except JPSN. JPSN had MSE = 0.006462. NMSE of JPSN model is 0.771. Hence, it fails NMSE test which requires NMSE <

* 1. for model to be acceptable. Performance comparisons of different methods are shown in the figure 5.4, 5.5, and 5.6. The R2 value of proposed model was found 0.9984 and correlation = 0.9991. Same experiment was done using ARIMA method where we found high error rates with MSE = 3.52, MAE = 1.47, NMSE ≅ 0, correlation = 0.9629, and R2 = 0.9217. Comparing performance of proposed model with ARIMA performance, our method outperforms ARIMA model. Dividing up a complex system into modules and then arranging these modules hierarchically is an important part of making the world “theoretically intelligible to the human mind”. However, there are usually many possible choices of decomposition. While Earth system modelers strive, as Plato suggested, to “carve nature at its joints”, in practice, judgment is needed to find a decomposition that is fit for this purpose. Comparison of architectural patterns in software has become a standard approach for analyzing the constraints that shape such decisions. The boundaries between components in an Earth system model represent both natural boundaries in the physical world (e.g., the ocean surface) and divisions between communities of expertise (e.g., ocean science vs. atmospheric physics). The model architecture must facilitate simulation of physical processes that cross these boundaries (e.g., heat transport) as well as support collaboration between knowledge communities within the work practices of model development (e.g., to study climate feedbacks). Each major model component tends to have two distinct uses: as a stand- alone component used by a specific sub-community, and as a building block for coupled Earth system modeling. Hence, there is a tension between the need for each component to remain loosely coupled to facilitate its ongoing use as a stand-alone.

The "Climate Emergency" continues to embody a renewed worldwide focus on tackling climate change. While there is no "one solution" to the multifaceted challenges brought about by this crisis, there is an onus on every citizen, in both a personal and professional capacity, to apply their skills and actions in addressing the profound pressures on the natural world. For those involved in the design of buildings and cities, be they architects, urbanists, or citizens, there is a deep responsibility to be aware of, and design for, the impact of climate change. With 36% of global energy devoted to buildings and 8% of global emissions caused by cement alone, the architectural community is deeply entwined with the flows of materials, energy, and ideas that relate to climate change, both causes, and solutions. Addressing the often-disparate waves of information associated with the topic we have curated a list of the most important facts and figures pertaining to architecture and climate change. Using exclusively reputable, trusted sources, this collection serves as a toolkit, a starting point for members of the architectural community to learn more about how their skills can be used to fight the biggest crisis of our time.

### CHAPTER - 6

### SYSTEM DESIGN

Most people don’t see themselves as having the personal power or influence to make a compelling difference in climate change. But so many of the design decisions made every day have a climate implication; each one can help promote a low-carbon future that doesn’t rely on fossil fuels. Those who create the products and built environments of everyday life—from mechanical engineers to architects—have an important role to play by designing for climate change.

* **Understanding the Problem. Climate change is real and progressing:** [2016 was the](https://www.nytimes.com/interactive/2017/01/18/science/earth/2016-hottest-year-on-record.html?_r=0) [hottest year on record](https://www.nytimes.com/interactive/2017/01/18/science/earth/2016-hottest-year-on-record.html?_r=0), for the third consecutive year. The effects of climate change are clear in rising sea levels, more frequent and intense storms, droughts, and floods. These changes in weather patterns create resource scarcity, displaced communities, increased risk of disease, and political and economic turmoil—all profoundly impacting people’s lives. Global warming is more than an environmental issue; it’s a human issue. But it’s also a business issue. Energy is one of the largest contributors to climate change, accounting for [84 percent](https://www.epa.gov/climate-impacts/climate-impacts-energy) of greenhouse-gas emissions in the United States. And as temperatures rise, so will energy costs, the [EPA predicts.](https://www.epa.gov/climate-impacts/climate-impacts-energy) Fortunately, the [2015 Paris Agreement](http://unfccc.int/paris_agreement/items/9485.php) is an encouraging step toward ensuring those rising energy needs don’t mean catastrophe for the planet. The Agreement forged a consensus among 197 nations (despite [recent claims](http://fortune.com/2017/01/30/donald-trump-paris-agreement-climate-change-withdraw/) that U.S. President Donald Trump is preparing to withdraw) to keep average worldwide temperatures from rising more than 2 degrees Celsius, thus averting climate calamity. But to achieve that, there’s work to be done.
* **Design and Climate Change:** The most significant way designers and engineers can address the climate-change challenge is by designing for higher productivity. Increased [productivity](http://www.businessdictionary.com/definition/productivity.html) requires maximizing value (output) while minimizing costs (inputs, such as energy and material resources). In terms of architecture and construction, that could mean designing buildings that use less energy. On the manufacturing side, it could be designing products that last longer and use less virgin material. One thing is certain: Energy and resource productivity is paramount in all forms of design.
* Designers and engineers who are aiming for higher productivity—and thus addressing climate change—typically employ these five approaches:

**1. Ask Energy-Productivity Questions Up Front**. From the start, asking questions such as, “What is the energy implication of this design decision?” or “How will this choice increase energy and resource productivity?” ensures designers and engineers keep costs down while adding value. It’s important to raise these questions early—such as when choosing a building’s site or selecting a material for a new product—and often, thereby setting the precedent for continued sustainable design choices throughout the life of the project. Those seemingly small decisions add up to big impacts. For example: [Shanghai Tower](http://du.gensler.com/vol6/shanghai-tower/), China’s tallest building, saved 25 percent of its material costs because the architects focused on how to lightweight the structure while maintaining its strength. And they did that by analyzing how wind would strike the building and then shaping it into a twist so that it would cut through the wind. Those decisions were planned from the beginning of the project, not tacked on as a “green” afterthought.

1. **Model, Simulate, and Repeat.** It is now easier than ever to evaluate early design decisions by using simulation technology to quickly model and test alternatives. Electronics company [Opto22](https://www.youtube.com/watch?v=hJhW3xJ1Dew) discovered this firsthand. The company analyzed the electronic cooling for its Groov hardware interface (used to control the Bellagio Hotel fountains, among other things) with the goal to give the device a smaller form factor. The redesigned interface eliminated all moving parts, including two fans, resulting in greater energy efficiency, reduced raw-material requirements, and lowered costs for assembly and labor—a full 70 percent savings in labor costs. If designers and engineers, like Opto22, use the simulation [tools](http://sustainability.autodesk.com/) available today, they can run the early analyses that yield similar energy, material, or time savings—while also saving money. But project stakeholders must be aware of the gains so that they start to expect and even demand sustainable practices throughout the design and engineering process**.** Take the Long-Term View. It’s important that designers and engineers make long-term decisions that consider the product or project lifecycle. Take industrial-fan manufacturer Howden France, for example. The company [analyzed its fan wheels](http://damassets.autodesk.net/content/dam/autodesk/www/products/simulation-mechanical/docs/pdfs/howden-customer-story.pdf) (PDF), accounting for fatigue and adjusting their thickness and weight. Optimizing the wheels’ weight enabled Howden to reduce the fan’s inertia, leading to better lifetime performance and reducing the power required for the motor. And that led to lower operating costs—a big win for customers.

Designing for repair is also key; after all, the most sustainable product is often the one that lasts the longest. HP knew this when it released the [Elite x2 1012 G1](http://ifixit.org/blog/8135/hats-off-to-hps-repairable-tablet/) tablet. The company offers online repair documentation and readily available replacement parts so that users can repair their units themselves.

1. **Consider the Whole System.** Designers and engineers who use the process of [whole-](https://sustainabilityworkshop.autodesk.com/products/whole-systems-and-lifecycle-thinking) [systems thinking](https://sustainabilityworkshop.autodesk.com/products/whole-systems-and-lifecycle-thinking) consider the relationships among complex systems, instead of focusing on individual parts of systems. This is important because challenges such as climate change represent a set of interconnected issues that can’t be solved in isolation. By taking a big- picture view and considering the whole system, the most important opportunities often arise, and can be incorporated, early in the process. The team behind the [Urbee](https://sustainabilityworkshop.autodesk.com/project-gallery/urbee-whole-systems-thinking-cleaner-car) hybrid electric car applied whole-systems thinking to the goal of producing an affordable, fast, safe method of personal transportation that runs on a small amount of energy. By taking into account all the interdependent factors and uncovering issues they could address up front, the team was able to deliver a fuel-efficient hybrid that gets 150 miles to the gallon and weighs just 1,200 pounds.
2. **Communicate.** Designing for higher productivity will result in cost savings for customers and more value delivered. But those victories must be communicated to customers. It’s up to designers and engineers to educate their clients, suppliers, subcontractors, and colleagues about the choices that can make a real difference to the bottom line—and to climate change. Global architecture and engineering firm HOK has taken significant steps in this area by signing on to the [AIA 2030 Commitment](https://www.aia.org/resources/6616-the-2030-commitment). This national framework tracks progress toward meeting the [Architecture 2030 Challenge,](http://architecture2030.org/2030_challenges/2030-challenge/) which charges the worldwide design-build community with achieving carbon neutrality on all new buildings, developments, and major renovations by 2030. “More of our designers are engaged in energy discussions with clients, engineers, contractors, and consultants,” [said Anica Landreneau,](http://www.hok.com/about/news/2016/12/05/anica-landreneau-hok-on-track-to-achieve-carbon-neutral-design-portfolio-by-2030/) HOK’s director of sustainable consulting. “Having discussions about energy efficiency earlier and more frequently in the design process has enabled us to identify significant first-cost and operational savings for our clients.Simply put, sustainable design is good design. And responding to climate change is much more pressing today than ever before. If more designers and engineers commit to asking efficiency questions early, using simulation and analysis tools, and taking a long-term and whole-system view, the climate-change needle will move. And the best part is, they’ll be reducing emissions while increasing value for their customers.

### HOW TEMPRATURE CHANGE IS MONITERED:

Temperature monitoring system controls and regulates the temperature of a particular environment. A temperature monitoring system has become an essential part of healthcare, hospitals, clinics, food business, and other industries in recent years. With a temperature monitoring system, you can easily track, control, and regulate the products’ temperature in a specific environment. A temperature monitoring system makes sure that your temperature- dependent products stay safe as they are being transported.

### What temperature monitoring systems offer

Here are some of the critical features of a standard temperature monitoring system:

### Automatic alerts

You’ll get automatic alerts on activities like high and low temperatures. This way, you can learn when or the extent to which the quality of your products may be compromised.

### Notifications for change in temperature

A slight change in temperature will alert the monitoring system, and you’ll get a notification about the rise or drop in temperature. It will help you to contact the logistics team and inform them about the situation quickly.

### Reports

The temperature monitoring system will automatically generate reports that you can analyse trends and take new measures accordingly.

### Temperature tracking

Now you don’t have to manually call or check to maintain updated temperature readings or receive updates on the products. With a temperature monitoring system, you can easily track the temperatures fluctuations.

### Customized indicators

With a temperature monitoring system, you can easily customize indicators for minimum and maximum thresholds based on the type of product or container. You can also reset the indicators on the go to reflect changing needs. Due to lack of transportation of goods gets damaged in transport, which means that temperature monitoring systems are one way to go to avoid losses.

What should be taken into account in selecting temperature monitoring systems? This brings us to the next section.

* 1. **What to look for in temperature monitoring systems:**

There are multiple things that should be considered in temperature monitoring systems as the stakes are high. Below are six features that will make up a reliable and useful temperature monitoring system.

**Over-time temperature data**

Prevention is better than cure, right? Knowing the temperatures your products have gone through will help you to make informed decisions. From swapping out some products to opening a dispute, over-time temperature data is an objective source of facts in your products’ lifecycle.

Over-time data will assist you in being better at preventing loss, managing incidents, and more. You can make quick decisions to save your business and make real progress.

### Package-level monitoring:

Shipments that are compact and portable are often at risk. It’s essential to track the temperatures of individual packages that were outside cold rooms for longer than the rest of the shipment.

Sensors must be a part of your shipment to ensure package-level monitoring. The hardware must be attached to every other package and stay connected with the parent system so that the containers can be easily monitored.

### Reliable analytics:

Faulty sensors or broken equipment can lead to false alerts and vague reporting data. The data that your temperature monitoring system collects must be reliable. Consistency over time guarantees the best results.

If the analytics of your temperature monitoring system isn’t reliable, it can affect your business. A false temperature alarm here and a wrong alert there can take a toll on your business, so it’s essential to make sure that your system is working all the time correctly.

Data is everything when you are monitoring the temperature of products that are far away from you. A faulty system can produce the wrong information, which defeats the whole purpose of a temperature monitoring system.

Invest in a reliable temperature monitoring system so you can make the most of analytics.

### User-friendliness:

A temperature monitoring system must be easy to understand. An average user should be able to monitor, analyse, and make the most of the temperature monitoring system without scratching their head.

Your temperature monitoring solution must be simple, relevant, and customizable. Furthermore, it must also be easy to set up, to troubleshoot or debug, and continually fine- tune as your relationship to the technology strengthens. If your system is hard to configure, it will waste your both time and money.

Look for a user-friendly temperature monitoring system that is easy to understand and can be configured by any non-tech-savvy folk.

### Responds to incidents:

There are plenty of things that can go wrong in a temperature monitoring system. And most of the risks can be covered by alerts, responses, and plans. But what about exceptions?

Your temperature monitoring system must account for external help whenever needed, like professionally trained people to handle coolant spills and spoilage support.

All-in-all, an excellent temperature monitoring system should be able to cope up with complications at any time of the day.

**Important parts of a temperature monitoring system**

Commonly, a temperature monitoring system consists of five parts. Below, we’ll cover all the six parts briefly.

### Temperature sensor or probe

Temperature is one of the most common measurements across all industries. From pharmaceutical to environmental monitoring, the temperature is monitored in pretty much every industry.

A temperature monitoring system is able to capture temperatures through a sensor. Since every type of sensor has its own limits and capabilities, you must select one according to your needs.

The three most common temperature sensors you’ll find in a temperature monitoring system are **thermocouples**, **RTD sensors,** and **thermistors**.

**Thermocouples** are used in pretty much every low to mid budget temperature monitoring system. They are the least expensive sensors you’ll find in a temperature monitoring system. If your business doesn’t require extremely high-temperature accuracy, thermocouples are, hands down, the most cost-efficient and straightforward sensors.

Next up, **RTD sensors** provide more accurate readings than the thermocouples but with the expense of a narrow temperature range. RTD sensors are ideal for freezers and refrigerators as their temperature ranges are limited.

Lastly, **thermistors** are just like RTD sensors; their resistance changes with temperature. But in thermistors, the change in resistance is non-linear. This means that they can only be fit in a temperature monitoring system that supports the non-linear resistance curve.

### Thermal buffer

Thermal buffers are materials and liquids that are attached with temperature sensors to decrease their response time. Lower response time means that the sensor will be able to pick up minor changes in temperature in a short period of time.

Thermal buffers help a temperature monitoring system to get an accurate temperature of the products or environment. Some of the common thermal buffers equipped in a temperature monitoring system are vials full of glass beads, nylon blocks, and glycol bottles.

### TMD or temperature measurement device

Every modern-day system needs a digitized heart, right? The same goes for a temperature monitoring system. A temperature measurement device (TMD) does the following tasks:

* Connects to sensors for digitizing the temperature value
* Records temperature data
* Temporarily stores data before transmission to the storage destination

There are three types of temperature measurement devices: standalone, networked, and wireless. A **standalone TMD** doesn’t require any other device to record or process temperatures. It’s an all-in-one device that controls everything. A **networked TMD** is connected with a PC, server, or cloud. It can connect to a device via LAN or Wi-Fi to send and receive data. Lastly, a **wireless TMD** includes a wider wireless range, more robust data update rate, and ease of use. It’s ideal to invest in a wireless TMD if you have multiple fixed points that need data to be collected in real time. Fixed points like stationary freezers and refrigerators are actually one of the few cases where real-time alerts really come in handy, as there often is someone to fix the situation as soon as issues occur.

## CHAPTER - 7 MODULES DESCRIPTION

In our project at first we import various libraries so that we can use it to extract the data then we use various classifications of machine learning models such as:

* Linear regression
* LGBMRegressor
* Time Series Forecasting using ARIMA and SARIMAX
* Auto regression
* Autocorrelation
* Partial Autocorrelation
* Prophet
* TBATS

### 7.1 Description of Machine learning models:

* **Linear Regression:**

Linear Regression fits a linear model with coefficients w = (w1, …, wp) to minimize the residual sum of squares between the observed targets in the dataset, and the targets predicted by the linear approximation.

**Parameters:**

* + **fit\_intercept*bool, default=True***
  + Whether to calculate the intercept for this model. If set to False, no intercept will be used in calculations (i.e. data is expected to be centered).
  + **normalize*bool, default=False***
  + This parameter is ignored when fit\_intercept is set to False. If True, the regressors X will be normalized before regression by subtracting the mean and dividing by the l2-norm. If you wish to standardize, please

use [StandardScaler](https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html" \l "sklearn.preprocessing.StandardScaler) before calling fit on an estimator with normalize=False.

* + **copy\_X*bool, default=True***
  + If True, X will be copied; else, it may be overwritten.
  + **n\_jobs*int, default=None***
  + The number of jobs to use for the computation. This will only provide speedup in case of sufficiently large problems, that is if firstly n\_targets > 1 and secondly X is sparse or if positive is set to True. None means 1 unless in

a [joblib.parallel\_backend](https://joblib.readthedocs.io/en/latest/parallel.html" \l "joblib.parallel_backend) context. -1 means using all processors. See [Glossary](https://scikit-learn.org/stable/glossary.html" \l "term-n_jobs) for more details.

* + **positive*bool, default=False***
  + When set to True, forces the coefficients to be positive. This option is only supported for dense arrays.

**Attributes:**

**coef\_*array of shape (n\_features, ) or (n\_targets, n\_features)***

Estimated coefficients for the linear regression problem. If multiple targets are passed during the fit (y 2D), this is a 2D array of shape (n\_targets, n\_features), while if only one target is passed, this is a 1D array of length n\_features.

**rank\_*int***

Rank of matrix X. Only available when X is dense.

**singular\_*array of shape (min(X, y),)***

Singular values of X. Only available when X is dense.

**intercept\_*float or array of shape (n\_targets,)***

Independent term in the linear model. Set to 0.0 if fit\_intercept = False.

## Linear Regression Model Representation :

* [Linear regression](https://en.wikipedia.org/wiki/Linear_regression) is an attractive model because the representation is so simple.

The representation is a linear equation that combines a specific set of input values (x) the solution to which is the predicted output for that set of input values (y). As such, both the input values (x) and the output value are numeric.

* The linear equation assigns one scale factor to each input value or column, called a coefficient and represented by the capital Greek letter Beta (B).
* One additional coefficient is also added, giving the line an additional degree of freedom (e.g. moving up and down on a two-dimensional plot) and is often called the intercept or the bias coefficient.

For example, in a simple regression problem (a single x and a single y), the form of the model would be:

**y = B0 + B1\*x**

* In higher dimensions when we have more than one input (x), the line is called a plane or a hyper-plane.
* The representation therefore is the form of the equation and the specific values used for the coefficients (e.g. B0 and B1 in the above example).
* When a coefficient becomes zero, it effectively removes the influence of the input variable on the model and therefore from the prediction made from the model (0 \* x = 0).
* This becomes relevant if you look at regularization methods that change the learning algorithm to reduce the complexity of regression models by putting pressure on the absolute size of the coefficients, driving some to zero.
* **LightGBM Regressor:**

init (boosting\_type='gbdt', num\_leaves=31, max\_depth=-1, learning\_rate=0.1, n\_estimators=100, subsample\_for\_bin=200000, objective=None, class\_ weight=None, min\_split\_gain=0.0, min\_child\_weight=0.001, min\_child\_samples=20, subsa mple=1.0, subsample\_freq=0, colsample\_bytree=1.0, reg\_alpha=0.0, reg\_lambda=0.0, rando m\_state=None, n\_jobs=- 1, importance\_type='split', \*\*kwargs)

**Construct a gradient boosting model.**

**Parameters**

• boosting\_type (str, optional (default='gbdt')) – ‘gbdt’, traditional Gradient Boosting Decision Tree. ‘dart’, Dropouts meet Multiple Additive Regression Trees. ‘goss’, Gradient-based One-Side Sampling. ‘rf’, Random Forest.

• num\_leaves (int, optional (default=31)) – Maximum tree leaves for base learners.

* + max\_depth (int, optional (default=-1)) – Maximum tree depth for base learners, <=0 means no limit.
  + learning\_rate (float, optional (default=0.1)) – Boosting learning rate. You can use using the parameter of fit method to shrink/adapt learning rate in training callback. Note, that this will ignore argument in training.

• n\_estimators (int, optional (default=100)) – Number of boosted trees to fit.

• subsample\_for\_bin (int, optional (default=200000)) – Number of samples for constructing bins.

• objective (str, callable or None, optional (default=None)) – Specify the learning task and the corresponding learning objective or a custom objective function to be used (see note below). Default: ‘regression’ for LGBMRegressor, ‘binary’ or ‘multiclass’ for LGBMClassifier, ‘lambdarank’ for LGBMRanker.

• class\_weight (dict, 'balanced' or None, optional (default=None)) – Weights associated with classes in the form {class\_label: weight}. Use this parameter only for multi-class classification task; for binary classification task you may

use or parameters. Note, that the usage of all these

parameters will result in poor estimates of the individual class probabilities. You may want to consider performing probability calibration (https://scikit- learn.org/stable/modules/calibration.html) of your model. The ‘balanced’ mode uses the values of y to automatically adjust weights inversely proportional to class frequencies in the input data as n\_samples / (n\_classes \* np.bincount(y)). If None,

all classes are supposed to have weight one. Note, that these weights will be

multiplied with if is specified.

(passed through the fit method)

• min\_split\_gain (float, optional (default=0.)) – Minimum loss reduction required to

make a further partition on a leaf node of the tree.

• min\_child\_weight (float, optional (default=1e-3)) – Minimum sum of instance weight (Hessian) needed in a child (leaf).

• min\_child\_samples (int, optional (default=20)) – Minimum number of data needed in a child (leaf).

• subsample (float, optional (default=1.)) – Subsample ratio of the training instance.

subsample\_freq (int, optional (default=0)) – Frequency of subsample, <=0 means no enable.

* colsample\_bytree (float, optional (default=1.)) – Subsample ratio of columns when constructing each tree.
* reg\_alpha (float, optional (default=0.)) – L1 regularization term on weights.
* reg\_lambda (float, optional (default=0.)) – L2 regularization term on weights.

• random\_state (int, RandomState object or None, optional (default=None)) – Random number seed. If int, this number is used to seed the C++ code. If RandomState object (numpy), a random integer is picked based on its state to seed the C++ code. If None, default seeds in C++ code are used.

• n\_jobs (int, optional (default=-1)) – Number of parallel threads to use for training (can be changed at prediction time).

• importance\_type (str, optional (default='split')) – The type of feature importance to be filled into feature\_importances\_. If ‘split’, result contains numbers of times the feature is used in a model. If ‘gain’, result contains total gains of splits which use the feature.

• How it differs from other tree based algorithm?

• Light GBM grows tree vertically while other algorithm grows trees horizontally meaning that Light GBM grows tree leaf-wise while other algorithm grows level-wise. It will choose the leaf with max delta loss to grow. When growing the same leaf, Leaf-wise algorithm can reduce more loss than a level-wise algorithm.

* **Time Series Forecasting using ARIMA and SARIMAX**
  + ARIMA, short for ‘Auto Regressive Integrated Moving Average’ is actually a class of models that ‘explains’ a given time series based on its own past values, that is, its own lags and the lagged forecast errors, so that equation can be used to forecast future values.
  + Any ‘non-seasonal’ time series that exhibits patterns and is not a random white noise can be modeled with ARIMA models.
  + An ARIMA model is characterized by 3 terms: p, d, q where,

p is the order of the AR term q is the order of the MA term

d is the number of differencing required to make the time series stationary

If a time series, has seasonal patterns, then you need to add seasonal terms and it becomes SARIMA, short for ‘Seasonal ARIMA’.

### AR: Autoregressive

This is the easiest part. **Autoregressive means that we regress the target variable on its own past values.** That is, we use lagged values of the target variable as our X variables:

**Y = B0 + B1\*Y\_log1 + B2\*Y\_log2 + ... + Bn\*Y\_logn**

That’s pretty straightforward. All this equation is saying is that the currently observed value of Y is some linear function of its past ***n*** values (where ***n*** is a parameter we choose; and B0, B1, etc. are the regression betas that we fit when we train our model). The previous equation is commonly called an AR(***n***) model where ***n*** denotes the number of lags. We can easily make this a forecast of the future by changing around the notation a little bit:

**Y\_forward1 = B0 + B1\*Y + B2\*Y\_log1 + B3\*Y\_log3 + ... Bn\*Y\_log(n-1)**

Now we are predicting the future value (1 time step ahead) using the current value and its past logs.

### I: Integrated

Integrated denotes that we apply a differencing step to the data.That is, instead of running a regression like the following:

**Y\_forward1 = B0 + B1\*Y + B2\*Y\_lag1 + ...**

We do this:

**Y\_forward1 - Y = B0 + B1\*(Y - Y\_lag1) + B2\*(Y\_lag1 - Y\_lag2) + ...**

What the second equation is saying is that the future change in Y is a linear function of the past changes in Y. Why bother with differencing? The reason is that differences are generally much more stationary than the raw undifferenced values. When we do time series modeling, we like our Y variables to be mean variance stationary. This means that the main statistical properties of a model do not vary depending on when the sample was taken. Models built on stationary data are generally more robust.

Take real GDP (real means that it’s been adjusted for inflation) for instance. It’s obvious from the plot that the raw GDP data is not stationary. It’s rising making the mean GDP in the first half of the plot much lower than the mean in the second half.

### Autoregressive Integrated Moving Average Model:

* + - An [ARIMA model](https://en.wikipedia.org/wiki/Autoregressive_integrated_moving_average) is a class of statistical models for analyzing and forecasting time series data.
    - It explicitly caters to a suite of standard structures in time series data, and as such provides a simple yet powerful method for making skillful time series forecasts.
    - ARIMA is an acronym that stands for AutoRegressive Integrated Moving Average. It is a generalization of the simpler AutoRegressive Moving Average and adds the notion of integration.
    - This acronym is descriptive, capturing the key aspects of the model itself. Briefly, they are:
    - **AR**: *Autoregression*. A model that uses the dependent relationship between an observation and some number of lagged observations.
    - **I**: *Integrated*. The use of differencing of raw observations (e.g. subtracting an observation from an observation at the previous time step) in order to make the time series stationary.
    - **MA**: *Moving Average*. A model that uses the dependency between an observation and a residual error from a moving average model applied to lagged observations.
    - Each of these components are explicitly specified in the model as a parameter. A standard notation is used of ARIMA(p,d,q) where the parameters are substituted with integer values to quickly indicate the specific ARIMA model being used.
    - The parameters of the ARIMA model are defined as follows:
    - **p**: The number of lag observations included in the model, also called the lag order.
    - **d**: The number of times that the raw observations are differenced, also called the degree of differencing.
    - **q**: The size of the moving average window, also called the order of moving average.
    - A linear regression model is constructed including the specified number and type of terms, and the data is prepared by a degree of differencing in order to make it stationary, i.e. to remove trend and seasonal structures that negatively affect the regression model.
    - A value of 0 can be used for a parameter, which indicates to not use that element of the model. This way, the ARIMA model can be configured to perform the function.

## Auto Regressive Model:

### Autocorrelation and Partial Autocorrelation

* + - Identification of an AR model is often best done with the PACF.
* For an AR model, the theoretical PACF “shuts off” past the order of the model. The phrase “shuts off” means that in theory the partial autocorrelations are equal to 0 beyond that point. Put another way, the number of non-zero partial autocorrelations gives the order of the AR model. By the “order of the model” we mean the most extreme lag of x that is used as a predictor.
* Look for sudden drop
  + - Identification of an MA model is often best done with the ACF rather than the PACF.
* For an MA model, the theoretical PACF does not shut off, but instead tapers toward 0 in some manner. A clearer pattern for an MA model is in the ACF. The ACF will have non-zero autocorrelations only at lags involved in the model.
* Look for exponential drop

### Parameters for ARIMA

1. p - Autoregressive (AR) Model Lags - Use PACF
2. d - No. of times differencing performed
3. q - Moving Average (MA) Lags - Use ACF

## Prophet:

* + Prophet is a procedure for forecasting time series data based on an additive model where non-linear trends are fit with yearly, weekly, and daily seasonality, plus holiday effects.
  + It works best with time series that have strong seasonal effects and several seasons of historical data. Prophet is robust to missing data and shifts in the trend, and typically handles outliers well.

Prophet is open source software released by Facebook's Core Data Science team.

* Accurate and fast.
* Prophet is used in many applications across Facebook for producing reliable forecasts for planning and goal setting. We’ve found it to perform better than any other approach in the majority of cases. We fit models in Stan so that you get forecasts in just a few seconds.
* Fully automatic.
* Get a reasonable forecast on messy data with no manual effort. Prophet is robust to outliers, missing data, and dramatic changes in your time series.
* Tunable forecasts.
* The Prophet procedure includes many possibilities for users to tweak and adjust forecasts. You can use human-interpretable parameters to improve your forecast by
* adding your domain knowledge.
* Available in R or Python. We’ve implemented the Prophet procedure in R and Python, but they share the same underlying Stan code for fitting. Use whatever language you’re comfortable with to get forecasts.

### Forecasting Time Series with Multiple Seasonalities using TBATS in Python:

* + - There are two interesting time series forecasting methods called BATS and TBATS that are capable of modeling time series with multiple seasonalities.
    - The names are acronyms for key features of the models: Trigonometric seasonality, Box-Cox transformation, ARMA errors, Trend and Seasonal components.
    - TBATS model takes it roots in exponential smoothing methods and can be described by the following equations
    - Each seasonality is modeled by a trigonometric representation based on Fourier series.
    - One major advantage of this approach is that it requires only 2 seed states regardless of the length of period.
    - Another advantage is the ability to model seasonal effects of non-integer lengths.
    - For example, given a series of daily observations, one can model leap years with a season of length 365.25.
    - **BATS** differs from TBATS only in the way it models seasonal effects. In BATS we have a more traditional approach where each seasonality is modeled by:

 **https://miro.medium.com/max/1400/1*D8KY2jQRmVRLHY5f4RNqKA.png**

* This implies that BATS can only model integer period lengths. Approach taken in BATS requires *m\_i* seed states for season *i*, if this season is long the model may become intractable.

### How Does TBATS Choose The Final Model ?

* + - Under the hood TBATS will consider various alternatives and fit quite a few models. It will consider models:
    - with Box-Cox transformation and without it.
    - with and without Trend
    - with and without Trend Damping
    - with and without ARMA(p,q) process used to model residuals
    - non-seasonal model
    - various amounts of harmonics used to model seasonal effects
    - The final model will be chosen using [Akaike information criterion](https://en.wikipedia.org/wiki/Akaike_information_criterion) (AIC).
    - In particular auto ARMA is used to decide if residuals need modeling and what p and q values are suitable.
    - Existing Implementation: So far the only implementation has been available in R language, in [forecast](https://www.rdocumentation.org/packages/forecast) package.
    - **New Implementation**: We have created a new implementation of **TBATS in Python**, available at [GitHub](https://github.com/intive-DataScience/tbats). In the rest of the article we will provide the example usage and compare the performance of this implementation with the other methods.

**CHAPTER -8**

**SYSTEM TESTING**

## Test objective:

The test objective is what type of testing the team and its testers executed, and why. For example, if the test report covers [functional testing](https://www.techtarget.com/searchsoftwarequality/definition/functional-testing), regression and [performance testing,](https://www.techtarget.com/searchsoftwarequality/definition/performance-testing) the test report writer will need to describe the objective for each testing type.In most cases, regression testing is the main purpose of the test execution. The objective of regression testing can vary, but a team usually performs the practice to search for defects once developers add new feature code to an existing code base. Regression testing is done prior to any new release and varies in length of time and testing depth. If a team's regression testing includes integration, performance or other testing types, the writer should indicate the purpose of each test specifically in the report's objective section.

## Test cases, test coverage and execution details:

The next element on how to write a test report is to explain the test suite. Specifically, include what type of test was executed, where it is stored and when it was executed. The test report writer can also add the name of the QA professional who ran the test, but that's not a specific requirement. Instead, it's more important to include the how/what/why and actual test results.

## Test Model Used:

For performing testing across machine learning models we use three types of system testing.

Model testing is a technique where any software's runtime behaviour is recorded and tested under some dataset and prediction table that the model has already predicted.

Some model-based testing scenarios are used to describe numerous aspects of the Machine Learning model.

### The way to test the model:

### Test the basic logic of the model.

* + 1. Manage the performance using the concept of manual testing.
    2. Work on the accuracy of the model.
    3. Check the performance on the real data, try to use unit testing.

### Pre-train Testing

Pre-train tests: As per the name, pre-train testing is the testing technique that allows you to catch the bugs before even running the model. It checks whether there is any label missing in your training and validation dataset; and it does not require any running parameter.

The pre-train testing goal is to avoid wastage during training jobs. Problem statement of pre-train testing:

* + 1. Check leakage label in your training dataset and validation dataset.
    2. Check the single gradient to find the loss of data.
    3. Check the shape of the dataset to ensure the alignment of data.

### Post-train Testing

Post Train Testing is used to check whether it performs all the validations correctly or not. The main purpose of post-train testing is to validate the logic behind the algorithm and find out the bugs, if any.

The post-train testing deals with the job behaviour. They are basically of three types.

* + 1. Invariant tests
    2. Directional tests
    3. Minimum functional tests

### Invariant Test

Invariant Testing is the testing technique where we check how the input data is changing without affecting the entire performance of the Machine Learning model. Here each input model is paired with the prediction and maintains consistency.

Invariant testing provides a logical guarantee about the application; this is a very low testing technique. This type of testing is mainly observed in Domain-Driven Design (DDD). Invariant testing follows three basic steps:

* + 1. Identify invariants.
    2. Enforce invariants.
    3. Refactor necessary invariants.

### Directional Test

Directional testing is a type of hypothesis testing where a direction of testing is specified earlier to the testing. This testing technique is also known as a one-tailed test.

Directional testing is way more powerful than the non-directional or invariant testing technique.

Unlike invariant testing, perturbation can change the outcome of the model in the provided input.

### Minimum functional test

Functional testing is used to check whether the software or model is working according to the pre-requisite dataset or not. This uses the black box testing technique.

### Types of functional testing:

* + 1. Unit testing
    2. Smoke testing
    3. Sanity testing
    4. Usability testing
    5. Regression testing
    6. Integration testing

The minimum functional testing model works in a similar manner to a traditional unit testing technique where the data is classified into different components, and the testing is applied over those components.

### Ways to perform functional testing:

* + 1. Testing based on user requirements.
       1. Testing based on business requirements.

### Understanding the Model Development Pipeline

The pipelining concept in machine learning is used to automate the workflows. Machine Learning pipelines are iterative processes, repeated one after the another to improve the algorithm's accuracy and model, and achieve the required successful solution.

### An evaluation of the Model development pipeline includes the following steps:

* + 1. Pre-Train Test.
    2. Post-Train Test.
    3. Train model.
    4. Evaluation of model.
    5. Review and approval of dataset.

### Benefits of Model Testing:

* + 1. Easy maintenance.
    2. Less cost.
    3. Early detection.
    4. Less time-consuming.
    5. More job satisfaction.

### Issues while performing Model-Based Testing in Machine Learning

While working over any model, there are many shortcomings we have to deal with, which can be due to a design issue or implementation issues. Here are some drawbacks of the Model-Based Testing Technique:

* + 1. Deep understanding of problem statement is required.
    2. Different skill sets are required.
    3. More emphasis is placed on a learning curve.
    4. More human power is required.

**TABLE I** :TESTING GENERAL ACTIVITY DIMENTION

|  |  |
| --- | --- |
| Sub- dimension | Possible values |
| A: Test Planning | Testing cost estimation [9] |
| B: Test Case Management Test case design [1] | Test case prioritization [1]  Test case refinement [6] Test case evaluation [16] |
| C: Debugging | Fault localization [18]  Bug prioritization [18]  Fault prediction [15] |

According to the defined categories, six main dimensions and some sub-dimensions are obtainable in the classification framework. The framework’s dimensions are as follows: Dimension 1-Testing approach: According to the older testing terminology [2], we define the testing approach dimension with three possible values: black-box, white-box, and graybox [2]. Based on the black-box approach, testing can be performed using the external description of a software system such as the software specification. In a white-box approach, the internal properties of a software system like source code can be used for testing purposes. The grey- box approach is a combination of the two, which considers both internal and external properties at the same time. Dimension 2-Testing general activity: In the second dimension, the standard testing process life-cycle [13] inspired us to define the a) test planning, b) test case management, and c) debugging sub-dimensions. With respect to the testing process life- cycle, we found out that these three phases are critical and that automation can effectively assist them in order to reduce the cost and time of the whole testing process. The possible

activities that can be automated based on ML in a, b, and c are presented in Table I. In the test planning sub-dimension, testing cost estimation can help test managers to predict testing process cost and time and provide good testing plan to manage the testing process efficiently. Test case management includes several tasks such as test case prioritization, which intends to prioritize the test input space in terms of test case effectiveness; test case design, which intends to generate high quality test cases; test case refinement, which intends to map the current specification of a software system to the existing test cases in order to reuse the available test cases; and test case evaluation which intends to measure the quality of the generated test cases. In the Debugging sub-dimension, fault localization can help to find the exact location of the program that is defected. In addition, bug prioritization intends to prioritize the revealed faults based on their severities; later test engineer can focus on more critical faults accordingly. Fault prediction can assist test engineers in the debugging stage, in the sense that potential faults for a given program are predicted.

**CHAPTER -9**

**CONCLUSIONS AND FUTURE WORK**

In this thesis, we proposed a model for temperature forecasting. The objective of this study was to forecast the daily mean temperature using three machine learning models , EDA AND DATA PRE-PROCESSING and smoothing technique. Model showed acceptable and significantly high performance in terms of root mean squared error, mean squared error and mean absolute error on predicting average next day temperature of chennai. Model used the previous n day’s temperature where the optimal trade-off resolution method selected the value of n. Smoothing dataset played an important role in performance of the model. In our work we will be predicting the weather from 1992-2024 in which for some particular date and year we can tell how much temperature is there and produce the approximate results based on our evaluation criteria. Results are intrinsically a high and accurate as demonstrated as it is steady for exceptions and forecasting, so one approach to enhance the straight relapse show is by accumulation of more information using linear regression and SVM. Showing that the decision of model was efficient and effective that its expectations can be enhanced by promote accumulation of information under the proposed scheme. For future scope the same can be incorporated over apache spark for concurrent prediction of weather whereas the same

Can be compare with the results obtained from sensors.

**SOURCE CODE**

**#Import Libraries**

import pandas as pd import numpy as np

import statsmodels.api as sm import keras

import keras.utils import tensorflow as tf

from keras import utils as np\_utils import tensorflow\_datasets as tfds

ds = tfds.load('emnist', split='train', shuffle\_files=True) assert isinstance(ds, tf.data.Dataset)

print(ds)

#import fbprophet import fbprophet

!pip install prophet

**#Data Visualization** import seaborn as sns

import matplotlib.pyplot as plt

%matplotlib inline plt.style.use('seaborn-dark')

**#DateTime**

import datetime as dt

**#Models**

from sklearn.linear\_model import LinearRegression from lightgbm import LGBMRegressor

**#Sklearn**

from sklearn.metrics import mean\_squared\_error

from sklearn.model\_selection import KFold, cross\_val\_score, StratifiedKFold, train\_test\_spli t

from sklearn.preprocessing import StandardScaler

**#Time to run program**

import time

def load\_data():

'''

Function to Load the Train, Test and Submission Data

returns: train, test, submission dataframes '''

train = pd.read\_csv("/content/drive/MyDrive/dataset/train.csv")

test = pd.read\_csv("/content/drive/MyDrive/dataset/new\_test\_2.csv")

submission = pd.read\_csv("/content/drive/MyDrive/dataset/sample\_submission\_2.csv")

return train, test, submission TARGET = 'temp'

feature = ['date']

train, test, submission = load\_data() train.head()

test.head() submission.head()

#RMSE

def rmse():

y\_pred = train.iloc[10000:11322, 2] y = train.iloc[10000:11322, 0]

metric = np.sqrt(mean\_squared\_error(y, y\_pred)) print(f"RMSE of Data is: {metric}")

#Hackathon Metric

def predict(model, model\_features):

pred\_train = model.predict(X\_train[model\_features]) pred\_val = model.predict(X\_val[model\_features])

print(f"Train RMSE = {np.sqrt(mean\_squared\_error(y\_train, pred\_train))}") print(f"Test RMSE = {np.sqrt(mean\_squared\_error(y\_val, pred\_val))}")

def run\_gradient\_boosting(clf, fit\_params, train, test, features):

N\_SPLITS = 5

oofs = np.zeros(len(train)) preds = np.zeros((len(test)))

target = train[TARGET]

folds = StratifiedKFold(n\_splits = N\_SPLITS)

stratified\_target = pd.qcut(train[TARGET], 10, labels = False, duplicates='drop')

feature\_importances = pd.DataFrame()

for fold\_, (trn\_idx, val\_idx) in enumerate(folds.split(train, stratified\_target)): print(f'\n------------- Fold {fold\_ + 1} ')

### Training Set

X\_trn, y\_trn = train[features].iloc[trn\_idx], target.iloc[trn\_idx]

### Validation Set

X\_val, y\_val = train[features].iloc[val\_idx], target.iloc[val\_idx]

### Test Set

X\_test = test[features]

scaler = StandardScaler()

\_ = scaler.fit(X\_trn)

X\_trn = scaler.transform(X\_trn) X\_val = scaler.transform(X\_val) X\_test = scaler.transform(X\_test)

\_ = clf.fit(X\_trn, y\_trn, eval\_set = [(X\_val, y\_val)], \*\*fit\_params)

fold\_importance = pd.DataFrame({'fold': fold\_ + 1, 'feature': features, 'importance': clf.feat ure\_importances\_})

feature\_importances = pd.concat([feature\_importances, fold\_importance], axis=0)

### Instead of directly predicting the classes we will obtain the probability of positive class

.

preds\_val = clf.predict(X\_val) preds\_test = clf.predict(X\_test)

fold\_score = metric(y\_val, preds\_val)

print(f'\nRMSE score for validation set is {fold\_score}')

oofs[val\_idx] = preds\_val

preds += preds\_test / N\_SPLITS

oofs\_score = metric(target, oofs) print(f'\n\nRMSE for oofs is {oofs\_score}')

feature\_importances = feature\_importances.reset\_index(drop = True)

fi = feature\_importances.groupby('feature')['importance'].mean().sort\_values(ascending = Fa lse)[:20][::-1]

fi.plot(kind = 'barh', figsize=(12, 6)) return oofs, preds, fi

def metric(y\_true, y\_pred):

return np.sqrt(mean\_squared\_error(y\_true, y\_pred))

def download\_preds(preds\_test, file\_name = 'hacklive\_sub.csv'):

## 1. Setting the target column with our obtained predictions submission['prediction'] = preds\_test

## 2. Saving our predictions to a csv file submission.to\_csv(file\_name, index = False)

## 3. Downloading and submitting the csv file from google.colab import files files.download(file\_name)

#Download Submission File

def download(model, model\_features, file\_name = 'prophet.csv'):

pred\_test = model.predict(model\_features)

#Setting the target column with our obtained predictions submission['prediction'] = pred\_test

#Saving our predictions to a csv file submission.to\_csv(file\_name, index = False)

#Downloadingthe csv file files.download(file\_name)

def join\_df(train, test):

df = pd.concat([train, test], axis=0).reset\_index(drop = True) features = [c for c in df.columns if c not in [feature, TARGET]] df[TARGET] = df[TARGET].apply(lambda x: np.log1p(x))

return df, features

def split\_df\_and\_get\_features(df, train\_nrows):

train, test = df[:train\_nrows].reset\_index(drop = True), df[train\_nrows:].reset\_index(drop = True)

features = [c for c in train.columns if c not in [feature, TARGET]] return train, test, features

df, features = join\_df(train, test) df.head()

print(f"train.shape: {train.shape}") print(f"test.shape: {test.shape}") train.describe()

#Check Datatypes train.dtypes

print(f"Train Null Value Count: {train.isnull().sum()}") print(f"Test Null Value Count: {test.isnull().sum()}") #Temperature Distribution

train[TARGET].plot(kind = 'density', title = 'Temperature Distribution', fontsize=14, figsize= (10, 6))

#Log Temperature Distribution

\_ = pd.Series(np.log1p(train[TARGET])).plot(kind = 'density', title = 'Log Temperature Distri bution', fontsize=14, figsize=(10, 6))

#Temperature Boxplot

train[TARGET].plot(kind = 'box', vert=False, figsize=(12, 4), title = 'Temperature Boxplot', f ontsize=14)

#Log Temperature BoxPlot

pd.Series(np.log1p(train[TARGET])).plot(kind = 'box', vert=False, figsize=(12, 4), title = 'Lo g Temperature Boxplot', fontsize=14)

#Convert `date` column datatype to `datetime` df['date'] = pd.to\_datetime(df['date'])

print(f"Train Null Value Count: {train.isnull().sum()}") print(f"Test Null Value Count: {test.isnull().sum()}") #Make basic datetime features

# df['day\_of\_week'] = df['date'].dt.dayofweek df['year'] = df['date'].dt.year

df['month'] = df['date'].dt.month

df['week'] = df['date'].dt.isocalendar().week

#Get Train and Test sets from df

train, test, features = split\_df\_and\_get\_features(df, train.shape[0])

#Define the features

features = [c for c in df.columns if c not in [feature, TARGET]] features = features[1:]

features df.head()

train.fillna(np.mean(train['temp']), inplace=True) #Declare Features and Target from Training Dataset X = train[features]

y = train[TARGET]

#Split Training and Validation Datasets

X\_train, X\_val, y\_train, y\_val = train\_test\_split(X, y, test\_size=0.3, random\_state = 42)

X.shape, y.shape #Linear Regression

model = LinearRegression()

model.fit(X\_train[features], y\_train) predict(model, features)

import warnings warnings.filterwarnings("ignore")

model = LGBMRegressor(n\_estimators = 5000,

learning\_rate = 0.01,

colsample\_bytree = 0.76, metric = 'None',

)

fit\_params = {'verbose': 300, 'early\_stopping\_rounds': 200, 'eval\_metric': 'rmse'}

lgb\_oofs, lgb\_preds, fi = run\_gradient\_boosting(clf = model, fit\_params = fit\_params, train = train, test = test, features = features)

#Load Data

train, test, submission = load\_data() #Convert `date` column to datetime train.date = pd.to\_datetime(train.date) #Set `date` as index train.set\_index('date', inplace = True) train.head()

train.plot(figsize = (20, 10)) #Import adfuller test

from statsmodels.tsa.stattools import adfuller

#H0: It is not stationary #H1: It is stationary

def adfuller\_test(temp): result=adfuller(temp)

labels = ['ADF Test Statistic','p-value','#Lags Used','Number of Observations Used'] for value,label in zip(result,labels):

print(label+' : '+str(value) )

if result[1] <= 0.05:

print("strong evidence against the null hypothesis(Ho), reject the null hypothesis. Data h as no unit root and is stationary")

else:

print("weak evidence against null hypothesis, time series has a unit root, indicating it is non-stationary ")

adfuller\_test(train.temp)

train['Seasonal First Difference']=train['temp']- train['temp'].shift(12) #Because 1 year has 12 months

## Again test dickey fuller test adfuller\_test(train['Seasonal First Difference'].dropna())

train['Seasonal First Difference'].plot()

from statsmodels.graphics.tsaplots import plot\_pacf ,plot\_acf import warnings

warnings.filterwarnings("ignore")

fig = plt.figure(figsize = (12, 8)) ax1 = fig.add\_subplot(211)

fig = plot\_pacf(train['Seasonal First Difference'].iloc[13:],lags=40,ax=ax1) ax2 = fig.add\_subplot(212)

fig = plot\_acf(train['Seasonal First Difference'].iloc[13:],lags=40,ax=ax2)

from statsmodels.tsa.arima\_model import ARIMA

model=ARIMA(train['temp'],order=(2,0,2)) model\_fit=model.fit()

model\_fit.summary()

train['forecast']=model\_fit.predict(start=10000,end=11321,dynamic=True) train[['temp','forecast']].plot(figsize=(12,8))

import statsmodels.api as sm begin = time.time()

model=sm.tsa.statespace.SARIMAX(train['temp'],order=(2, 1, 2),seasonal\_order=(2, 1, 2, 12)

)

results=model.fit()

#End TIme

end = time.time()

print(f"\n\nTime of execution = {end - begin}")

#Forecast train['forecast']=results.predict(start=10000,end=11321,dynamic=True) train[['temp','forecast']].plot(figsize=(12,8))

rmse()

df = pd.concat([train, test])

df['forecast'] = results.predict(start = 11322, end = 14883, dynamic= True) df[['temp', 'forecast']].plot(figsize=(12, 8))

rmse()

def prophet\_rmse(y\_true, y\_pred):

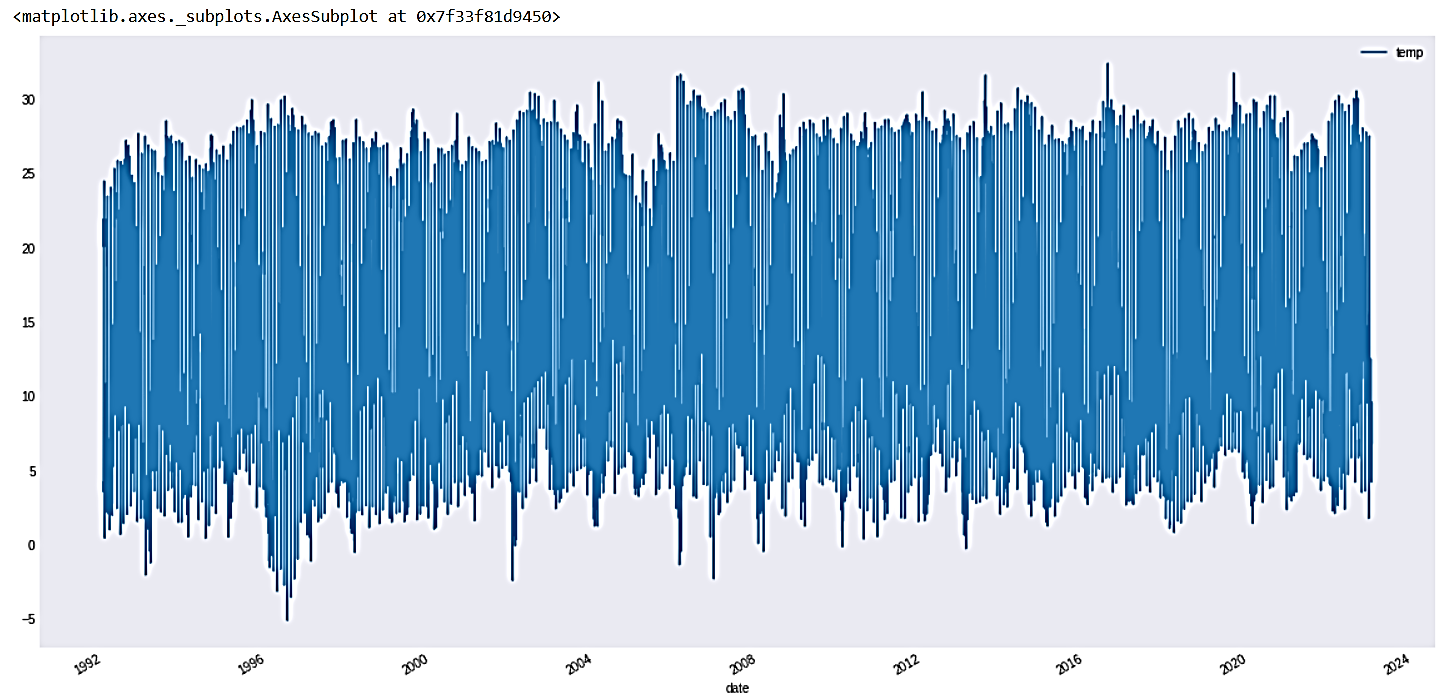
y\_true, y\_pred = np.array(y\_true), np.array(y\_pred) return np.sqrt(mean\_squared\_error(y\_true, y\_pred)) import fbprophet

from fbprophet import Prophet

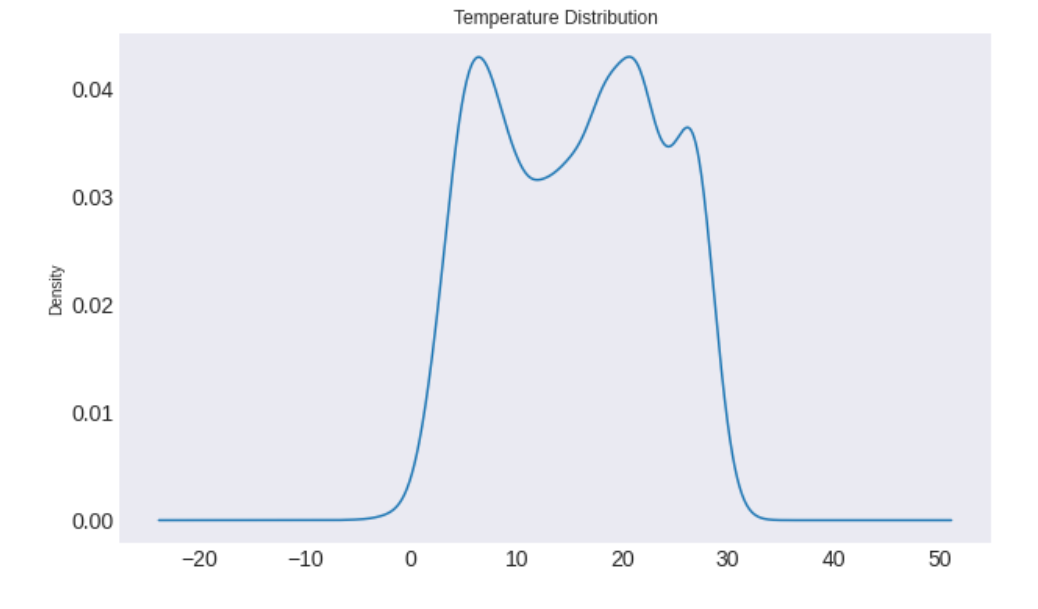
### SCREENSHOTS

### 

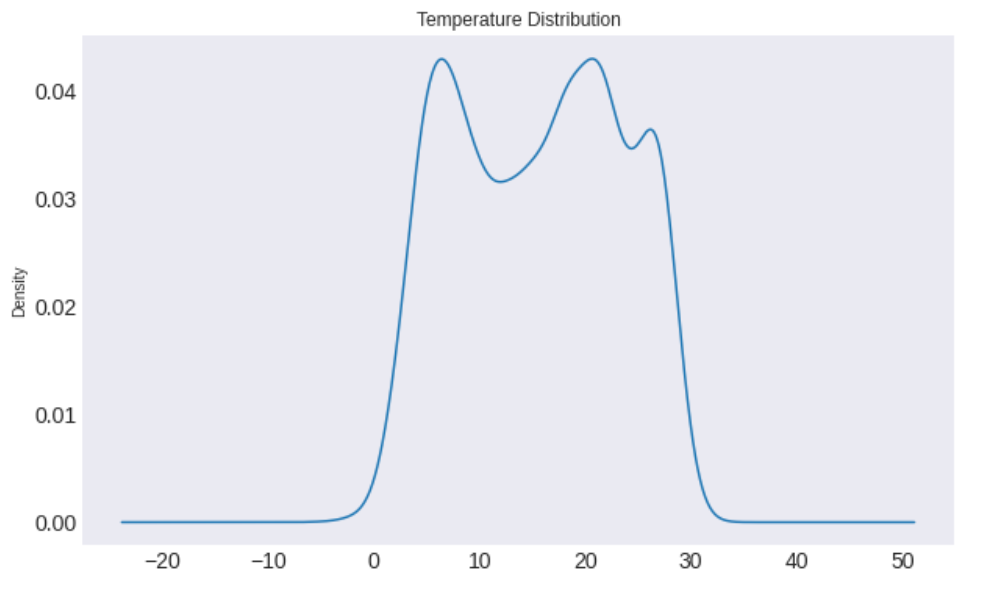
### Fig 3 Temperature Box Plot



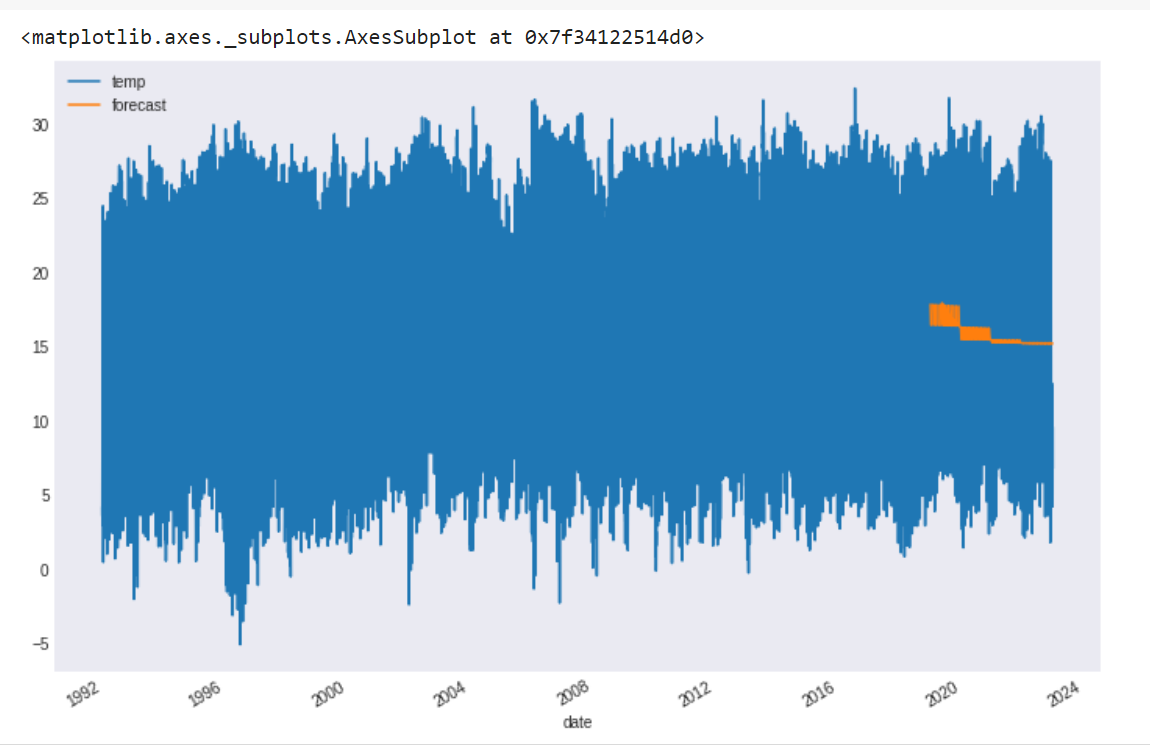
**FIG 4 Temperature Change**



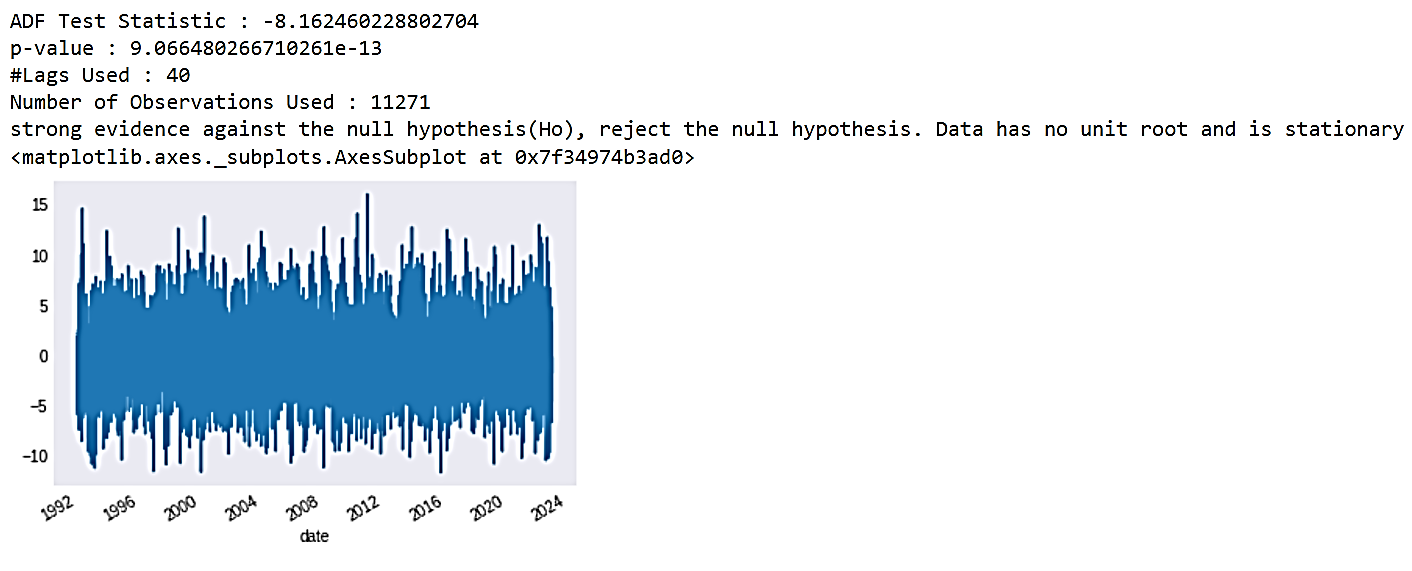
**Fig 5 Temperature distribution curve based on density change**

****

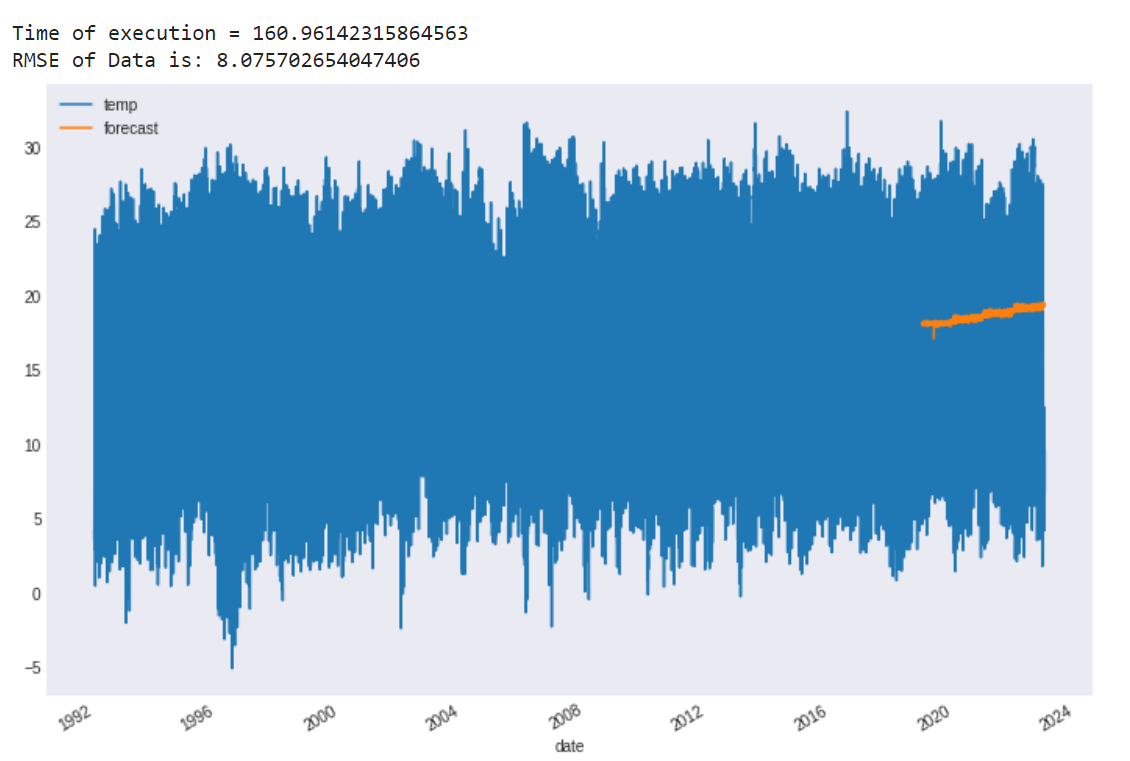
**Fig 6 Temperature distribution curve**



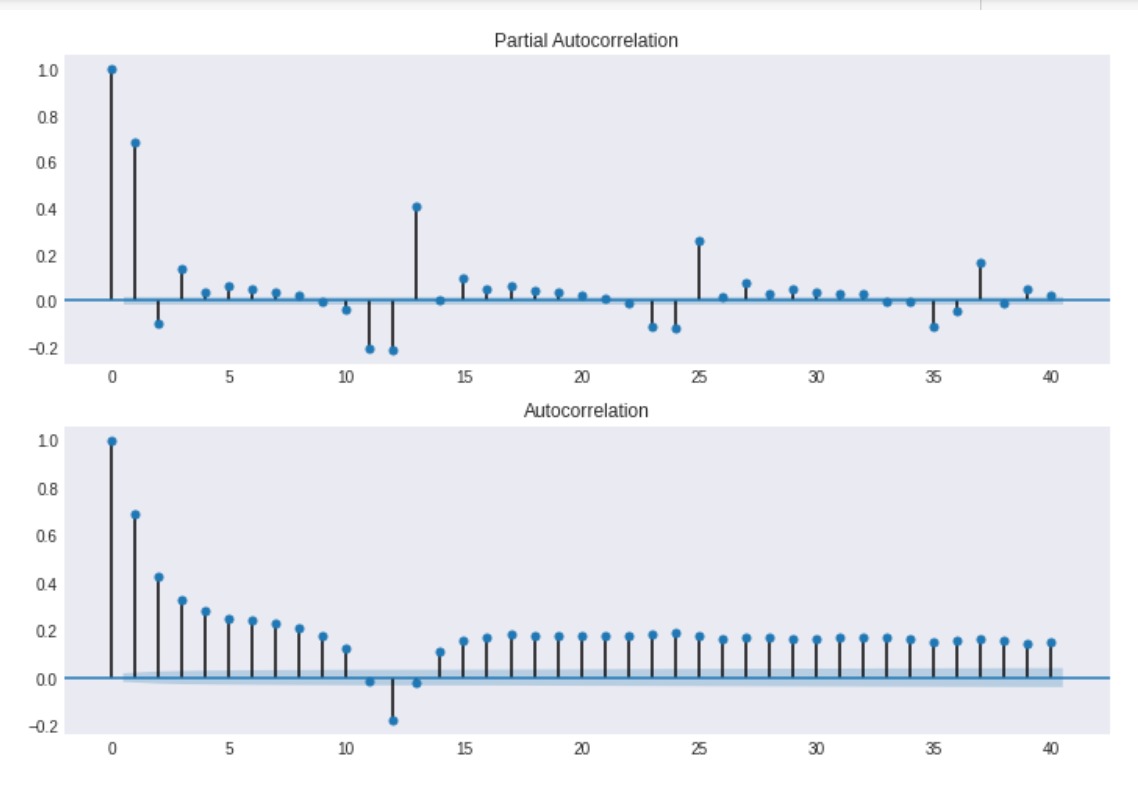
**Fig 7 Temperature forecast changes**

****

**Fig 8 Temperature graph using ADF Test Statistics**



**Fig 9 Time of execution Graph**



**Fig 10 Correlation and Autocorrelation**

**REFERENCES**

* A. Kumar, J. Luo and G. Bennett, "Statistical Evaluation of Lower Flammability.
* Ackerman, K. V., and E. T. Sundquist. 2008. Comparison of two U.S. power-plant carbon dioxide emissions data sets. Environmental Science & Technology 42(15):5688-5693.
* Adams, P. N., and D. L. Inman. 2009. Climate Change and Potential Hotspots of Coastal Erosion Along the Southern California Coast—Final Report. CEC-500-2009-022-F, Sacramento, California Energy Commission.
* Adger, W. N., I. Lorenzoni, and K. O’Brien. 2009a. Adaptation now. In Adapting to Climate Change: Thresholds, Values, Governance. W. N. Adger, I. Lorenzoni, and K. L. O’Brien, eds. Cambridge: Cambridge University Press.
* Adger, W. N., J. Paavola, S. Huq, and M. J. Mace, eds. 2006. Fairness in Adaptation to Climate Change. Cambridge, MA: MIT Press.
* Adger, W. N., S. Agrawala, M. M. Q. Mirza, C. Conde, K. L. O’Brien, J. Pulhin, R. Pulwarty, B. Smit, and K. Takahashi. 2007. Assessment of adaptation practices, options, constraints and capacity. In Climate Change 2007: Impacts, Adaptation and Vulnerability: Contribution of Working Group II to the Fourth Assessment Report of the Intergovernmental Panel on Climate Change.
* Adger, W. N., S. Dessai, M. Goulden, M. Hulme, I. Lorenzoni, D. R. Nelson, L. O. Naess, J. Wolf, and A. Wreford. 2009b. Are there social limits to adaptation to climate change? Climatic Change 93(3-4):335-354.
* Adler, M. D., and E. A. Posner. 2006. New Foundations of Cost-Benefit Analysis. Cambridge, MA: Harvard University Press. Agrawala, S. 2004. Adaptation, development assistance and planning: Challenges and opportunities. IDS Bulletin—Institute of Development Studies 35:50-54.
* Agrawal, A. 2008. The role of local institutions in adaptation to climate change. In Social Dimensions of Climate Change Workshop. Washington, DC: Social Development Department, The World Bank.
* Agrawal, A., and N. Perrin. 2008. Climate Adaptation, Local Institutions, and Rural Livelihoods. Ann Arbor, MI: International Forestry Resources and Institutions Program, University of Michigan.
* AGU (American Geophysical Union). 2009. Geoengineering the Climate System. A Position Statement of the American Geophysical Union (adopted by the AGU Council on December 13, 2009). Washington, DC: AGU.
* Airame, S., J. E. Dugan, K. D. Lafferty, H. Leslie, D. A. Mcardle, and R. R. Warner. 2003. Applying ecological criteria to marine reserve design: A case study from the California Channel Islands. Ecological Applications 13 (1):S170-S184.
* Akbari, H., M. Pomerantz, and H. Taha. 2001. Cool surfaces and shade trees to reduce energy use and improve air quality in urban areas. Solar Energy 70(3):295-310.
* Akbari, H., S. Menon, and A. Rosenfeld. 2009. Global cooling: Increasing world-wide urban albedos to offset CO2. Climatic Change 94(3-4):275-286.
* Albrecht, A., D. Schindler, K. Grebhan, U. Kohnle, and H. Mayer. 2009. Storminess over the North-Atlantic European region under climate change—a review. Allgemeine Forst And Jagdzeitung 180(5-6):109-118.
* Aldy, J. E., and R. N. Stavins. 2007. Architectures for Agreement. Cambridge, MA: Cambridge University Press.
* Alheit, J., and E. Hagen. 1997. Long-term climate forcing of European herring and sardine populations.1997. Long-term climate forcing of European herring and sardine populations. Fisheries Oceanography 6:130-139.
* Allan, J. D., M. A. Palmer, and N. L. Poff. 2005. Climate change and freshwater ecosystems. Pp. 272-290 in Climate Change and Biodiversity. T. E. Lovejoy and L. Hannah, eds. New Haven, CT: Yale University Press.
* Alley, R. B., J. Marotzke, W. D. Nordhaus, J. T. Overpeck, D. M. Peteet, R. A. Pielke, Jr., R.Alley, W. M. 1984. The Palmer Drought Severity Index—Limitations and assumptions. Journal of Climate and Applied Meteorology 23:1100-1109.
* Alley, W. M., T. E. Reilly, and F. O. Lehn. 1999. Sustainability of ground-water resources.
* Anderies, J. M., M. A. Janssen, and E. Ostrom. 2004. A framework to analyze the robustness of social-ecological systems from an institutional perspective. Ecology and Society 9(1):18. Available at [http://www.ecologyandsociety.org/vol9/iss1/art18/print.pdf.](http://www.ecologyandsociety.org/vol9/iss1/art18/print.pdf) Accessed May 10, 2010.
* Anderson, P. J., and J. F. Piatt. 1999. Community reorganization in the Gulf of Alaska following ocean climate regime shift. Marine Ecology-Progress Series 189:117-123.
* Andersson, K., and E. Ostrom. 2008. Analyzing decentralized resource regimes from a polycentric perspective. Policy Sciences 41(1):71-93.
* Andreadis, K. M., and D. P. Lettenmaier. 2006. Trends in 20th century drought over the continental United States. Geophysical Research Letters 33(10).
* Andreae, M. O., and P. Merlet. 2001. Emission of trace gases and aerosols from biomass burning. Global Biogeochemical Cycles 15(4):955-966.
* Angel, S., S. Sheppard, and D. Civco. 2005. The Dynamics of Global Urban Expansion. Washington, DC: Department of Transport and Urban Development, The World Bank.
* Angert, A., S. Biraud, C. Bonfils, C. C. Henning, W. Buermann, J. Pinzon, C. J. Tucker, and Annan, J. D., J. C. Hargreaves, R. Ohgaito, A. Abe-Ouchi, and S. Emori. 2005. Efficiently constraining climate sensitivity with paleoclimate simulations. Scientific Online Letters on the Atmosphere 1:181-184.
* Ansolabehere, S., and D. M. Koninsky. 2009. Public attitudes toward construction of new power plants. Public Opinion Quarterly 73(3):566-577.
* Anthoff, D., R. Nicholls, and R. Tol. 2010. The economic impact of substantial sea-level rise. Mitigation and Adaptation Strategies for Global Change 15(4):321-335.
* Anthoff, D., R. S. J. Tol, and G.W. Yohe. 2009. Risk aversion, time preference, and the social cost of carbon. Environmental Research Letters 4(2):024002.
* Antle, J. M. 2009. Agriculture and the Food System. Washington, DC: Resources for the Future.
* Armitage, D., F. Berkes, and N. Doubleday, eds. 2007. Adaptive Co-Management: Collaboration, Learning, and Multi-Level Governance. Seattle, WA: University of Washington Press.
* Arneth, A., N. Unger, M. Kulmala, and M. O. Andreae. 2009. Clean the air, heat the planet? Science 326(5953):672-673.
* Arriaga, L., A. E. Castellanos, E. Moreno, and J. Alarcon. 2004. Potential ecological distribution of alien invasive species and risk assessment: A case study of buffel grass in arid regions of Mexico. Conservation Biology 18(6):1504-1514.
* Arrigo, K. R., G. Van Dijken, and S. Pabi. 2008. Impact of a shrinking Arctic ice cover on marine primary production. Geophysical Research Letters 35 (19).
* Arrow, K., P. Dasgupta, L. Goulder, G. Daily, P. Ehrlich, G. Heal, S. Levin, K.-G. Maler, S. Schneider, D. Starrett, and B. Walker. 2004. Are we consuming too much? Journal of Economic Perspectives 18(3):147-172.
* Arrow, K., R. Solow, P. R. Portney, E. E. Leamer, R. Radner, and H. Schuman. 1993. Report of the NOAA Panel on Contingent Valuation. Federal Register 58(1993):4601-4614.artificial intelligence models for estimating air temperatures. Stoch Environ Res Risk Assess(2020).
* Arunachalam, V. S., and E. L. Fleischer. 2008. The global energy landscape and materials innovation.2008. The global energy landscape and materials innovation. MRS Bulletin 33(4):264-276.
* Arvai, J., G Bridge, N. Dolsak, R. Franzese, T. Koontz, A. Luginbuhl, P. Robbins, K.Richards, K. Smith Korfmacher, B. Sohngen, J. Tansey, and A. Thompson. 2006. Adaptive management of the global climate problem: Bridging the gap between climate research and climate policy. Climatic Change 78:217-225.
* B. Mohammadi, S. Mehdizadeh, F. Ahmadi, et al. Developing hybrid time series and B. Ustaoglu,H.K. Cigizoglu, and M. Karaca, (2008), Forecast of daily mean, maximum
* C. Lea, M. D. Flynn, R. Vidal, A. Reiter, and G. D. Hager. Temporal convolutional networks for action segmentation and detection. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2017. Computation 9,8 (1997), 1735–1780.
* D. Marutho, S. Hendra Handaka, E. Wijaya and Muljono, "The Determination of Cluster Number at k-Mean Using Elbow Method and Purity Evaluation on Headline News," 2018 Distance (LFD) using Four Hazardous Release Models", Process Safety Progress, 12(1), pp.
* Fung. 2005. Drier summers cancel out the CO2 uptake enhancement induced by warmer springs. Proceedings of the National Academy of Sciences of the United States of America 102:10823-10827.
* Grimm, N., Groffman, P., Staudinger, M., and Tallis, H. (2015). Climate change impacts on ecosystems and ecosystem services in the United States: process and prospects for sustained assessment. Climatic Change, 135(1), 97-109.
* Hammer G.L., J.W. Hansen, J.G. Phillips, J.W. Mjelde, H. Hill, H.A. Love, and A.Potgieter. 2001. Advances in Application of Climate Prediction in Agriculture. Agricultural Systems. 70(2-3)(November):515-553.
* In AMS (American Meteorological Society). 2009. Geoengineering the Climate System. A Policy Statement of the American Meteorological Society (adopted by the AMS Council on July 20, 2009). Washington, DC: AMS. and minimum temperature time series by three artificial neural network methods. Met. Apps, 15:431-445.
* International Seminar on Application for Technology of Information and Communication, 2018, pp. 533-538, doi: 10.1109/ISEMANTIC.2018.8549751.
* J Cifuentes, G Marulanda, A Bello, J Reneses. Air Temperature Forecasting Using
* K.H., Shin, et al. “Dynamical Prediction of Two Meteorological Factors Using the Deep Neural Network and the Long Short Term Memory $(1)$.”
* Keras Metrics—Keras documentation [WWW Document]. [Cross Reference]
* Lazo, J. K., Hosterman, H. R., Sprague-Hilderbrand, J. M., and Adkins, J. E. (2020). Impact-Based Decision Support Services and the Socioeconomic Impacts of Winter Storms, Bulletin of the American Meteorological Society, 101(5), E626-E639.
* M. L. Parry, O. F. Canziani, J. P. Palutikof, C. E. Hanson, and P. J. Van Der Linden, eds. Cambridge:M. L. Parry, O. F. Canziani, J. P. Palutikof, C. E. Hanson, and P. J. Van Der Linden, eds. Cambridge:Cambridge: Cambridge University Press.
* M. Norman. (2011). Parameterizations: Representing key processes in climate models
* Machine learning - Intuitive understanding of 1D, 2D, and 3D convolutions in convolutional neural networks - Stack Overflow. [Cross Reference] Machine Learning Techniques: A Review. Energies. 2020; 13(16):4215.
* N. A. Husaini, R. Ghazali, N. M. Nawi and L. H. Ismail, "Jordan pi-sigma neural network for temperature prediction" in Ubiquitous Computing and Multimedia Applications, Springer Berlin Heidelberg, pp. 547-558, 2011.

Network for effective weather forecasting using time-series data from the local weather station. Soft Comput 24, 16453–16482 (2020).

* P. Hewage, A. Behera, M. Trovati, et al. Temporal convolutional neural (TCN)
* P. Hewage, M. Trovati, E. Pereira, et al. Deep learning-based effective fine-grained
* Pierrehumbert, P. B. Rhines, T. F. Stocker, L. D. Talley, and J. M. Wallace. 2003. Abrupt climate change. Science 299(5615):2005-2010.
* R. Isabelle and T. Günther. 2019. “Visualization of Neural Network Predictions for Weather Forecasting.” Computer Graphics Forum 38 (1): 209–20. doi:10.1111/cgf.13453.
* S. Geological Survey Circular. Washington, DC: U.S. Geological Survey, Department of the Interior. Available at [http://purl.access.gpo.gov/GPO/LPS22429.](http://purl.access.gpo.gov/GPO/LPS22429) Accessed December 3, 2009.
* S. HOCHREITER, AND J. SCHMIDHUBER, Long short-term memory. Neural
* S. Stephen W (1997). "13.Convolution". The Scientist and Engineer's Guide to Digital Signal Processing (1 ed.). California Technical Publishing. ISBN 0-9660176-3-3.
* Suggested Citation:"References." National Research Council. 2010. Advancing the Science of Climate Change. Washington, DC: The National Academies Press. doi: 10.17226/12782.×
* Suggested Citation:"References." National Research Council. 2010. Advancing the Science of Climate Change. Washington, DC: The National Academies Press. doi: 10.17226/12782.×
* Understanding LSTM Networks. [Cross Reference] weather forecasting model. Pattern Anal Applic 24, 343–366 (2021). Without resolving them. Wiley Interdisciplinary Reviews: Climate Change. 2. 10.1002/wcc.122.
* Y. Wu, W. Ouyang, Z. Hao, C. Lin, H. Liu, Y. Wang, Assessment of soil erosion characteristics in response to temperature and precipitation in a freeze-thaw watershed, Geoderma, Volume 328, 2018, Pages 56-65, ISSN 0016-7061.
* Z. Karevan, S. Mehrkanoon and J. A. K. Suykens, "Black-box modeling for temperature prediction in weather forecasting," 2015 International Joint Conference on Neural Networks (IJCNN), Killarney, Ireland, 2015, pp. 1-8, doi: 10.1109/IJCNN.2015.7280671.