VISVESVARAYA TECHNOLOGICAL UNIVERSITY

“Jnana Sangama”, Belgaum – 590 018

A project report on

“Machine Learning Based Reservoir Outflow Management System”

Submitted in partial fulfillment of the Requirements for the VIII Semester
for the award of the degree of

BACHELOR OF ENGINEERING

in

INFORMATION SCIENCE & ENGINEERING

submitted by

Aman Raj Singh (1CR16IS011)
Darshan C V (1CR16IS027)
Ganesh Swaroop V (1CR16IS033)
Kishore S (1CR16IS041)

Under the guidance of

Mr. Prasad B S
Assistant Professor
Dept. of ISE, CMRIT, Bengaluru

CMR INSTITUTE OF TECHNOLOGY

DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING

#132, AECS Layout, IT Park Road, Bengaluru-560037

2019-20

VISVESVARAYA TECHNOLOGICAL UNIVERSITY

“Jnana Sangama”, Belgaum – 590 018

DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING

Certificate

This is to certify that the project entitled, **“Machine Learning based Reservoir Outflow Management System”**, is a bonafide work carried out by **KISHORE S (1CR16IS041), GANESH SWAROOP V (1CR16IS033), DARSHAN C V (1CR16IS027)**, and **AMAN RAJ SINGH (1CR16IS011)** in partial fulfillment of the award of the degree of Bachelor of Engineering in Information Science & Engineering of Visvesvaraya Technological University, Belgaum, during the year 2019-20. It is certified that all corrections/suggestions indicated during reviews have been incorporated in the report. The project report satisfies the academic requirements in respect of the project work prescribed for the said Degree.

---------------------------------	--------------------------------	---------------------------------
Name & Signature of Guide	**Name & Signature of HoD**	**Name & Signature of Principal**
Mr. Prasad B S **Assistant Professor** **Dept. of ISE, CMRIT**	**Dr. M Farida Begam** **Professor and HoD** **Dept. of ISE, CMRIT**	**Dr. Sanjay Jain** **Principal** **CMRIT**

External Viva

Name of the examiner	**Signature with date**
1.	
2.	

CMR INSTITUTE OF TECHNOLOGY
BANGALORE-560037

DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING

Declaration

We, **KISHORE S (1CR16IS041), GANESH SWAROOP V (1CR16IS033), DARSHAN C V (1CR16IS027)**, and **AMAN RAJ SINGH (1CR16IS011)**, bonafide students of CMR Institute of Technology, Bangalore, hereby declare that the dissertation entitled, "**Machine Learning based Reservoir Outflow Management System**" has been carried out by us under the guidance of Mr. Prasad B S, Assistant Professor, CMRIT, Bengaluru, in partial fulfillment of the requirements for the award of the degree of Bachelor of Engineering in Information Science Engineering, of the Visvesvaraya Technological University, Belgaum during the academic year 2019-2020. The work done in this dissertation report is original and it has not been submitted for any other degree in any university.

Place: Bengaluru

Kishore S (1CR16IS041)
Ganesh Swaroop V (1CR16IS033)
Darshan C V (1CR16IS027)
Aman Raj Singh (1CR16IS011)

DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING

Acknowledgement

The satisfaction and euphoria that accompany a successful completion of any task would be incomplete without the mention of people who made it possible. Success is the epitome of hard work and perseverance, but steadfast of all is encouraging guidance.

So, it is with gratitude that we acknowledge all those whose guidance and encouragement served as beacon of light and crowned our effort with success. We would like to thank Dr. Sanjay Jain, Principal, CMRIT, Bangalore, for providing an excellent academic environment in the college and his never ending support for the B.E program.

We would like to express our gratitude towards Dr. Farida Begam, Professor and HOD, Department of Information Science & Engineering CMRIT, Bangalore, who provided guidance and gave valuable suggestions regarding the project.

We consider it a privilege and honour to express our sincere gratitude to our internal guide Mr.Prasad B.S, Asst. Professor, Department of Information Science & Engineering, CMRIT, Bangalore, for their valuable guidance throughout the tenure of this project work.

We would like to express our sincere gratitude to our well-wisher and external guide Mr. Nischal Harohalli Padmanabha, Vice President of Engineering, Omnius, Berlin, Germany for his valuable guidance throughout the project work.

We would like to thank all the faculty members who have always been very cooperative and generous. Conclusively, we also thank all the non- teaching staff and all others who have done immense help directly or indirectly during our project.

Kishore S

Ganesh Swaroop V

Darshan C V

Aman Raj Singh

Abstract

Regardless of multiple dams built across the rivers to control the flow of water bodies, yet many calamities have occurred in recent times. One of the reasons for these is the legacy techniques being used on weather reports as well as in dams for flow management.
Since Machine Learning [1] algorithms are in hype in predicting future probabilities based on past data by using statistical method as its basis these techniques can be applied to train the machine on weather reports and Dam flow control and capacity data so as to provide efficient control over Dam water level management and create better alert systems in case of calamities. Also, we use the ensembling/ bagging technique on the results of the machine learning algorithms to improve the accuracy of the model.

Contents

List of Figures

List of Tables

Chapter 1

Preamble

1.1 Introduction

Water is an essential element for human survival and plays a major role in producing food supplies and in daily activities. But if uncontrolled or if not handled properly it can turn fatal to human life. Hundreds have died and thousands have lost their houses due to recent monsoon shifts and climatic changes. High rainfall in unexpected seasons have caused disastrous incidents in many regions of India.

India is agriculture based country. Hundreds of reservoirs and similar structures have been built across the country against various rivers and their tributaries for irrigation and domestic use of water in over thousands of cities. Rainfall is a major factor which affects most of the agricultural practices as large number of farmer community still follow farming practices that depend on reservoir and canal water supply, which inturn depends on rainfall. It is a known fact that monsoon and climate shifts have resulted in a huge problem in predicting rainfall and thereby managing its effects. One of many such unfortunate scenarios is overflow of reservoirs, causing a deluge in the geographically low and neighboring regions of catchment areas.

Reservoirs are large natural or artificial lake like structure used as a source of water supply. It can be built in many ways but is usually built by building a wall like structure across a watercourse that drains existing body of water, interrupting the watercourse to form an embayment within it. A dam constructed in a valley relies on the natural topography to provide most of the basin of the reservoir. Dams are typically located at a narrow part of a valley downstream of a natural basin. The valley sides act as natural walls, with the dam located at the narrowest practical point to provide strength and the lowest cost of construction.

These dam gates will be opened when reservoir capacity is maxed out and is about to overflow. But during varying rainfall rates and high inflow, opening dams can cause

floods and disaster. Machine learning algorithms can be used to train models to predict inflow so that water level and outflow rates can be decided with high accuracy so as to manage emergency situations.
An attempt is made to use machine learning algorithms to predict inflow and water level at the reservoir and hence improve the reservoir outflow management. In the traditional system, a monthly inflow forecast is designed by the reservoir management authority based on monthly rain predictions. But under recent circumstances, the rain and climate shifts have been highly unlikely to be predicted prior to a month. Hence the model proposed could stand as an option that provides a daily prediction of expected reservoir Inflow based on a recent rainfall pattern in the reservoir's catchment area.
Most of the research tries to predict rainfall anomalies and flood prediction as in Supriya *et al.* [13] using various regression models. Although some of the models do predicts yearly and monthly reservoir Inflow such as Somchit *et al.* [4] which predicts monthly inflow estimate for a reservoir in Thailand using wavelet artificial neural networks and Tiantian *et al.* [5] uses ANN, random forest, and Support vector machine(SVMs) to predict the monthly Inflow, very few have attempted daily rainfall predictions. One of the reasons being that on a daily basis it is extremely hard to understand rain patterns and correlate it to reservoir inflow for the model. Also, Inflow or in the broad aspect, the river water flow depends on various aspects like precipitation, atmospheric pressures, temperatures, regional aspects of river basins, and also population using those waters. Even then finding the source of the river its regional climate requires a lot of groundwork and analysis. This experiment is an attempt to predict daily Inflow on a daily basis using only past reservoir data and rainfall patterns in the respective catchment areas.
This experiment is performed on Harangi Dam situated in Kodagu, Karnataka state which stores water from the Harangi river which belongs to the list of main tributaries of the Kaveri river. Karnataka State Natural Disaster Monitor Center (KSNDMC) have been generous to provide us with 8 years of rainfall data, and made available the reservoir details for the same duration on their official website. Machine learning algorithms LOWESS (Locally Weighted Regression), logistic regression and deep learning algorithms like Recurrent neural networks and LSTM are used to develop a prediction model using climate information and reservoir parameters. Regression and Classification approaches are to evaluate the model. Ensembling or bootstrapping techniques using Linear Regression and Gradient boosted regression are used to improve prediction results.

1.2 Existing System

- Basically dam/reservoir gates are opened to release water when dam water level is at its full capacity. In other scenarios, opening release valves will be different based on the type of the dam.
- Only a limited number of dams exist solely for flood control which are known as attenuation or balancing reservoirs. The rest are solely meant for the purpose of storing water.
- Water inflow is acknowledged by calculating water level rise within a time interval, precalculated reservoir area and elevation values. In other cases, monthly predictions are made based on hydrological model of the reservoir.

1.3 Drawbacks

- No prediction methods are used to predict water inflow on a daily basis which is critical in current climatic situation.
- Varying rainfall and climatic changes can result in inflow which might exceed the water level capacities but releasing large amounts of water in a flash can result in floods.
- Difficult to vacate flood prone areas in the event of aforementioned scenarios.

1.4 Proposed System

- Water inflow can be predicted well in advance using various Machine learning methodologies.
- Various machine learning models can be trained to propose an near accurate model based on rainfall and other related data available on KSNDMC website.
- The data will be used to train the model to predict water inflow, water level that can be expected in the future.
- The system will suggest the amount of water to be released and at what increments in case of flood alerts so as to minimize the loss.

1.5 Plan of Implementation

- Harangi Reservoir in Karnataka State will be chosen for analysis and building the Machine learning model
- The first step will be extracting and cleaning the dataset from KSNDMC website. It involves both direct request for weather related data from KSNDMC as well as web scraping of reservoir related data from official KSNDMC website which is uploaded by Karnataka state Water Resource Department. We will use pre-processing techniques to deal with any missing or erroneous values.
- Once the data has been preprocessed, we will perform data analysis. This will help in finding relationships between the various attributes in the dataset, giving us an idea about the importance of each attribute.
- After preprocessing and analysis of the data, we will test the dataset with different machine learning models by training them, attempting to find a solution which gives the highest possible accuracy.
- Once the model is trained, the model will be developed as a web application for prediction on a daily basis along with an interface for visualizing the dam parameters with dataset attributes.

1.6 Problem Statement

Uneven rainfall and climate shifts have adverse effects in reservoir outflow management and have caused disasters. An attempt is made to use machine learning algorithms to predict inflow and water level on a daily basis at the Harangi reservoir., Karnataka and hence provide better reservoir outflow management system.

1.7 Objective of the Project

- The main objective of the project is to design a system which is capable of predicting reservoir inflow on daily basis and hence predicting overflow and suggesting early release of dam water with the quantity and increments.
- The system should predict reservoir level and other vital parameters of reservoir using Machine Learning on a daily basis.
- Data for training and daily prediction are to be obtained from official Karnataka State Natural Disaster Monitor Center website

- Different machine learning models is to be trained and tested to obtain as high accuracy as possible.

Chapter 2

Literature Survey

Regression Analysis of Annual Maximum Daily Rainfall and StreamFlow for Flood Forecasting in Vellar River Basin P.Supriya a* , M.Krishnaveni , M.Subbulakshmi .Aquatic Procedia 4 (2015) 1054 – 1061[13]
The paper talks on methods that can be applied on rainfall data to predict calamities and drought in the near future, the methods like regression analysis(using Datafit 9 software) and Thiessen polygon generation are used. The testing is done on Vellar river basin by training the model on 32 years of rainfall data. It describes the relation between weighted annual maximum daily rainfall, catchment area and Annual maximum daily streamflow.

Soft computing approach for rainfall-runoff modelling: A review Vinay Chandwani*, Sunil Kumar Vyas, Vinay Agrawal, Gunwant Sharma (ICWRCOE 2015) Aquatic Procedia 4 (2015) 957 – 963[14]
The review paper is discussion of methods that can be helpful in understanding relationships among multivariate problem models. Machine learning techniques and its application in the field of rainfall runoff models are discussed in detail Major techniques are :Artificial Neural Networks(ANN), Genetic algorithms and Fuzzy Logic. It also sites the major variables such as catchment areas, rainfall data, climatic parameters etc.

R. J. Abrahart, L. M. See. Neural network modelling of non-linear hydrological relationships. Hydrology and Earth System Sciences Discussions, European Geosciences Union, 2007, 11 (5), pp.1563-1579. ffhal-00305094f [15]
This is a paper on an experiment for evaluating capabilities of Artificial Neural Network for obtaining regression equations and predictions based on the equations. Eight

neural network models developed: four full or partial emulation of a recognised non-linear hydrological rainfall runoff model and four solutions developed on an identical set of inputs and a calculated runoff coefficient output used to train the network. Comparisons and outcome of experiments are discussed.

Long-Term Rainfall Analysis and Runoff Estimation in Mountainous Watershed: A Case Study from Mhadei River Basin, Goa and Karnataka Manoj Ibrampurkar1* and A. G. Chachadi2 Gond. Geol. Mag., V. 27(2), December, 2012. pp. 153-158[16]

The paper discuss the important parameters to be considered in the construction of Dam management and river water management which includes analysis on factors like catchment areas, river basin area, cultivation and drainage factors in those regions, land use/land cover and river basin-wise and catchment area wise rainfall data.

Somchit Amnatsan, Sayaka Yoshikawa and Shinjiro Kanae, Improved Forecasting of Extreme Monthly Reservoir Inflow Using an Analogue-Based Forecasting Method: A Case Study of the Sirikit Dam in Thailand, 9 November 2018[4]

The paper presents an experiment in predicting monthly inflow of a reservoir using the wavelet artificial neural network (WANN) model, and the weighted mean analogue method (WMAM). These techniques were used to forecast the monthly reservoir inflow of the Sirikit Dam, located in the Nan River Basin, one of the eight sub-basins of the Chao Phraya River Basin in Thailand.

Tiantian Yang, Ata Akbari Asanjan, Edwin Welles, Xiaogang Gao, Soroosh Sorooshian and Xiaomang Liu, Developing reservoir monthly inflow forecasts using artificial intelligence and climate phenomenon information, May 2017[5]

In this study, Random Forest [RF), Artificial Neural Network (ANN), and Support Vector Regression (SVR) are employed and compared with respect to their capabilities for predicting 1 month-ahead reservoir inflows for two headwater reservoirs in USA and China. Both current and lagged hydrological information and 17 known climate phenomenon indices, i.e., PDO and ENSO, etc., are selected as predictors for simulating reservoir inflows.

Chapter 3

Theoretical Background

3.1 Overview on Machine Learning

Machine learning is an application of artificial intelligence (AI) that gives systems the ability to automatically learn and evolve from experience without being specially programmed by the programmer. The process of learning begins with observations or data, such as examples, direct experience, or instruction, in order to look for patterns in data and make better decisions in the future based on the examples that we provide. The main aim of machine learning is to allow computers to learn automatically and adjust their actions to improve the accuracy and usefulness of the program, without any human intervention or assistance. Traditional writing of programs for a computer can be defined as automating the procedures to be performed on input data in order to create output artifacts. Almost always, they are linear, procedural and logical. A traditional program is written in a programming language to some specification, and it has properties like:

- We know or can control the inputs to the program.
- We can specify how the program will achieve its goal.
- We can map out what decisions the program will make and under what conditions it makes them.
- Since we know the inputs as well as the expected outputs, we can be confident that the program will achieve its goal

Traditional programming works on the premise that, as long as we can define what a program needs to do, we are confident we can define how a program can achieve that goal. This is not always the case as sometimes, however, there are problems that you can represent in a computer that you cannot write a traditional program to solve. Such problems resist a procedural and logical solution. They have properties such as:

- The scope of all possible inputs is not known beforehand.
- You cannot specify how to achieve the goal of the program, only what that goal is.
- You cannot map out all the decisions the program will need to make to achieve its goal.
- You can collect only sample input data but not all possible input data for the program.

3.1.1 Supervised and Unsupervised Learning

Supervised learning: ”Supervised learning is a type of machine learning algorithm that uses a known dataset (called the training dataset) to make predictions. The training dataset includes input data and response values.”,[17].
Unsupervised learning: ”Unsupervised learning is a type of machine learning algorithm used to draw inferences from datasets consisting of input data without labeled responses.”, [18]. In data mining and machine learning an abundance of models and algorithms can be found, but most fundamentally these are divided into supervised and unsupervised learning. One fundamental example has been mentioned in the foregoing section, the clustering of iris-species. Former is a supervised process where data points are labeled (”species A”, ”species B” or ”species C”) and labels are calculated for new data points. Comparing calculated labels according to the trained model with the original label gives the model’s accuracy, hence supervised.
Unsupervised learning on the other hand does not require any labeling, since the algorithm is searching for a pattern in the data. This might be useful when categorizing customers into different groups without a priori knowledge of which groups they belong to.

3.1.2 Types of Machine Learning Algorithm

For machine learning many different algorithms can be found. For simplicity these can be subdivided into four categories, where each category is good for different kind of problems. **Anomaly detection algorithms**, are good for finding unusual data points. **Trained classification algorithms** can be used to categorize unseen data. As an example, it could be used to take in data from a phone on movement to categorize what activity is being performed. **Clustering algorithms** group data into clusters and look for the greatest similarities. This can be used to find unknown connections on huge sets of data. **Regression algorithms** are used to find patterns and build models to predict numerical values from datasets. These will take multiple

inputs and determine how much each input affects the output.
Within the regression category there are many different basic algorithms. **Linear Regression** is the most classic type which solves linear relationships between inputs and outputs. **Neural network regression** is most common in deep learning and adaptable to regression problems, but might be too complex for simple regression problems and requires thorough training.
Machine learning algorithms are tools to automatically make decisions from data in order to achieve some over-arching goal or requirement. The promise of machine learning is that it can solve complex problems automatically, faster and more accurately than a manually specified solution, and at a larger scale. Over the past few decades, many machine learning algorithms have been developed by researchers, and new ones continue to emerge and old ones modified. In this project, we have focused on only supervised learning methods since our dataset contains labels.

3.2 Overview on Deep Learning

Deep learning (also known as deep structured learning or differential programming) is part of a broader family of machine learning methods based on artificial neural networks with representation learning. Learning can be supervised, semi-supervised or unsupervised Most modern deep learning models are based on artificial neural networks, specifically, Recurrent Neural Networks and Convolutional Neural Networks (CNN)s.
In deep learning, each level learns to transform its input data into a slightly more abstract and composite representation. In an image recognition application, the raw input may be a matrix of pixels; the first representational layer may abstract the pixels and encode edges; the second layer may compose and encode arrangements of edges; the third layer may encode a nose and eyes; and the fourth layer may recognize that the image contains a face. Importantly, a deep learning process can learn which features to optimally place in which level on its own. (Of course, this does not completely eliminate the need for hand-tuning; for example, varying numbers of layers and layer sizes can provide different degrees of abstraction.)
Deep learning architectures can be constructed with a greedy layer-by-layer method. Deep learning helps to disentangle these abstractions and pick out which features improve performance. For supervised learning tasks, deep learning methods eliminate feature engineering, by translating the data into compact intermediate representations akin to principal components, and derive layered structures that remove redundancy in representation. Deep learning algorithms can be applied to unsupervised learning tasks. This is an important benefit because unlabeled data are more abundant than

the labeled data. Examples of deep structures that can be trained in an unsupervised manner are neural history compressors and deep belief networks.

3.3 Time Series Problem

As our problem is related to forecasting future, it is directly related to **Time Series Forecasting**. A time series is a series of data points indexed (or listed or graphed) in time order. Most commonly, a time series is a sequence taken at successive equally spaced points in time. Thus it is a sequence of discrete-time data.
Time series forecasting is the use of a model to predict future values based on previously observed values. While regression analysis is often employed in such a way as to test theories that the current values of one or more independent time series affect the current value of another time series, this type of analysis of time series is not called ”time series analysis”, which focuses on comparing values of a single time series or multiple dependent time series at different points in time. Time series data have a natural temporal ordering. This makes time series analysis distinct from cross-sectional studies, in which there is no natural ordering of the observations.

3.4 Recurrent Neural Networks and LSTMs

A recurrent neural network (RNN) is a class of artificial neural networks where connections between nodes form a directed graph along a temporal sequence. This allows it to exhibit temporal dynamic behavior. Derived from feedforward neural networks, RNNs can use their internal state (memory) to process variable length sequences of inputs. This makes them applicable to tasks such as unsegmented, connected handwriting recognition or speech recognition.

Both finite impulse and infinite impulse recurrent networks can have additional stored states, and the storage can be under direct control by the neural network. The storage can also be replaced by another network or graph, if that incorporates time delays or has feedback loops. Such controlled states are referred to as gated state or gated memory, and are part of **long short-term memory networks (LSTMs)** and gated recurrent units. This is also called Feedback Neural Network.

3.4.1 Long Short Term Memory

Long short-term memory (LSTM) is an artificial recurrent neural network (RNN) architecture[1] used in the field of deep learning. Unlike standard feedforward neural networks, LSTM has feedback connections. It can not only process single data points (such as images), but also entire sequences of data (such as speech or video).

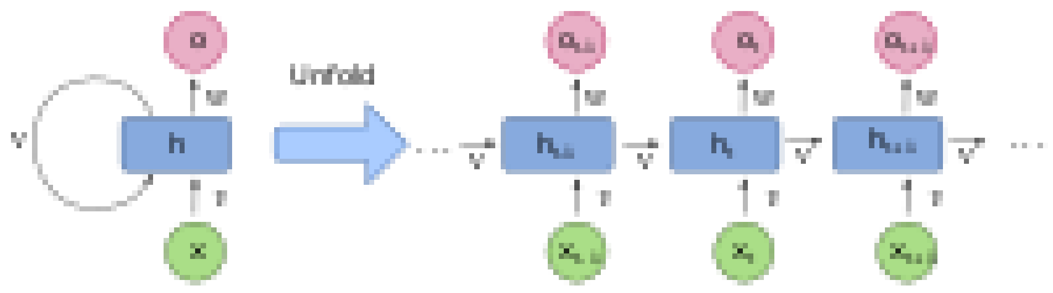


Figure 3.1: Recurrent Neural Network

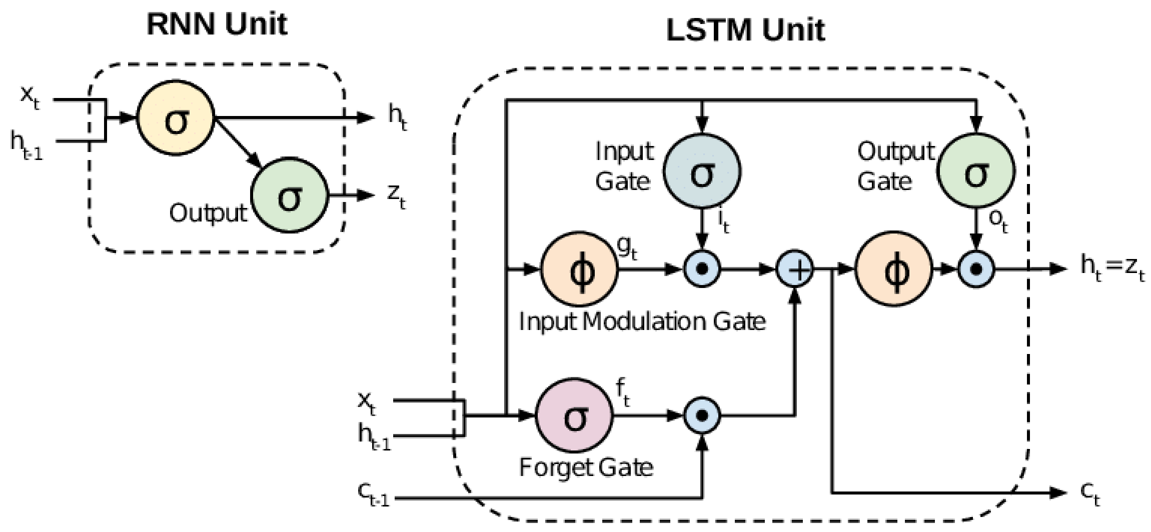


Figure 3.2: Comparison between RNN and LSTM unit

Working

The key to LSTMs is the cell state, the horizontal line running through the top of the diagram. The cell state is like a carousel. It carries information along the cell's time steps, with some minor arithmetic interactions. Gates control these information through, as to add new information and remove unwanted information. They involve sigmoid and tahH activated neural net layer and a few point-wise multiplications. They output a value between 0 or 1 and in most of the cases it is either 0 or 1, describing how much of each component should be let through. First gate is a forget gate which creates a sense of how much information of the old input needs to be forgotten. This information is multiplied with cell state, which removes unwanted memory or the memory that needs to be forgotten
$f_t = \sigma(W_f.[h_{t-1}, x_t] + b_f)$

Next is the input gate which adds new memory to cell state. This has two parts. First the input gate layer with sigmoid activation decides which values to be added. Next, a tanH layer creates a vector of new cell state values by pointwise multiplication with input gate layer values that could be added to the state. This new cell value is added with previously processed cell state.
$i_t = \sigma(W_i.[h_{t-1}, x_t] + b_i)$
$C_o = tanH(W_C.[h_{t-1}, x_t] + b_C)$
$c_t = f_t * c_{t-1} + i_t * C_o$
Finally, the output gate decides the output. This output will be based on the cell state, but will be a filtered version. First, a sigmoid layer is applied, which decides what parts of the cell state we're going to output. Next tanH will be applied on the cell state(to push the values to be between -1 and 1) and multiply the output of the sigmoid gate with it, so that only the necessary part of output is given out.
$o_t = \sigma(W_o.[h_{t-1}, x_t] + b_o)$
$h_t = o_t * tanH(c_t)$

3.5 Locally Weighted Regression

Locally Weighted Regression is a non-parametric algorithm, that is, the model does not learn a fixed set of parameters as is done in ordinary linear regression. It breaks the data-points into subsets and applies Linear Regression to the split subsets, and then augmented to improve the Prediction/Classification. As mentioned above it is a non-parametric algorithm,ie they do not have a fixed set of parameters. They learn as they hover through the dataset. This algorithm is introduced to overcome the limitations in predicting non-linear data by regression algorithms. Locally Weighted Regression is a confluence of 2 supervised Algorithm. One is K-Nearest Neighbours and the other is either Linear Regression or Polynomial Regression.
The k-Nearest Neighbors (KNN) algorithm is supervised, lazy learning based machine learning algorithm which can be used to learn both classification and regression problems. It works on the belief that similar things lie in close proximity. Laconically, to put it poetically, birds of same feather flock together. It calculates the Euclidean distance of two points and the classifies the dataset. Linear regression is used for finding linear relationship between target value and one or many dependent values. Linear regression performs the task to predict a dependent variable value (y) based on a given independent variables (x).
The augmentation of the above mentioned algorithms leads to the arisal of Locally Weighted Regression. Since this algorithm overcomes the disadvantages of Linear Regression, this can be used for smoothness of the predicted output.

Mathematical Motivation

Kernels are mathematical functions which are used to find the relation between parameters(ie between dependent and independent parameters) with exponentially reduced iterations(Less/No Iterations). Kernels are usually used for multi-dimensional spaced points. These points are reduced to single-dimensional computations which is then made easy for calculations. In Locally Weighted Regression the kernel used is :

$$w^{(i)} = e^{(-(x^{(i)}-x)^2/2\tau^2)}$$

Normal Equation: This is a beautiful mathematical expression which is used as an alternative for Gradient Descent. Gradient Descent is a costly iterative process with respect to time complexity. To overcome this, normal equation is used to which provides the output, in a single calculation. Thus this Equation cuts the temporal cost for many algorithms. In Locally Weighted Regression the weighted normal equation is shown below:

$$\Theta = (X^T.W.X)^{-1}.(X^T.W.y)$$

Smoothing Factor: In Locally Weighted Regression, the kernel function has a constant called Tau which is usually used as a Hyper-parameter(Hyper-parameters are tweaked to obtain improved predictions from the model). So, this tau is tweaked to obtained better results. Hyper-parameters should be used appositely,if the value provided is large then it may under-fit the model and if the value provided is less the it may over-fit the model. Tau is represented as as: τ

Algorithm

The integration of the mathematical models, lead to the emergence of Locally Weighted Regression.

Algorithm: To predict the model(h(x)) for the given data(X),

$$h(x_0) = x_0^T\beta(x_0)$$

The next step is to find the kernel for the beta function,

$$w^{(i)} = e^{(-(x^{(i)}-x)^2/2\tau^2)}$$

The next step is to find the Beta function which is,

$$\Theta = (X^T.W.X)^{-1}.(X^T.W.y)$$

These Mathematical representations are the Crux of this algorithm.

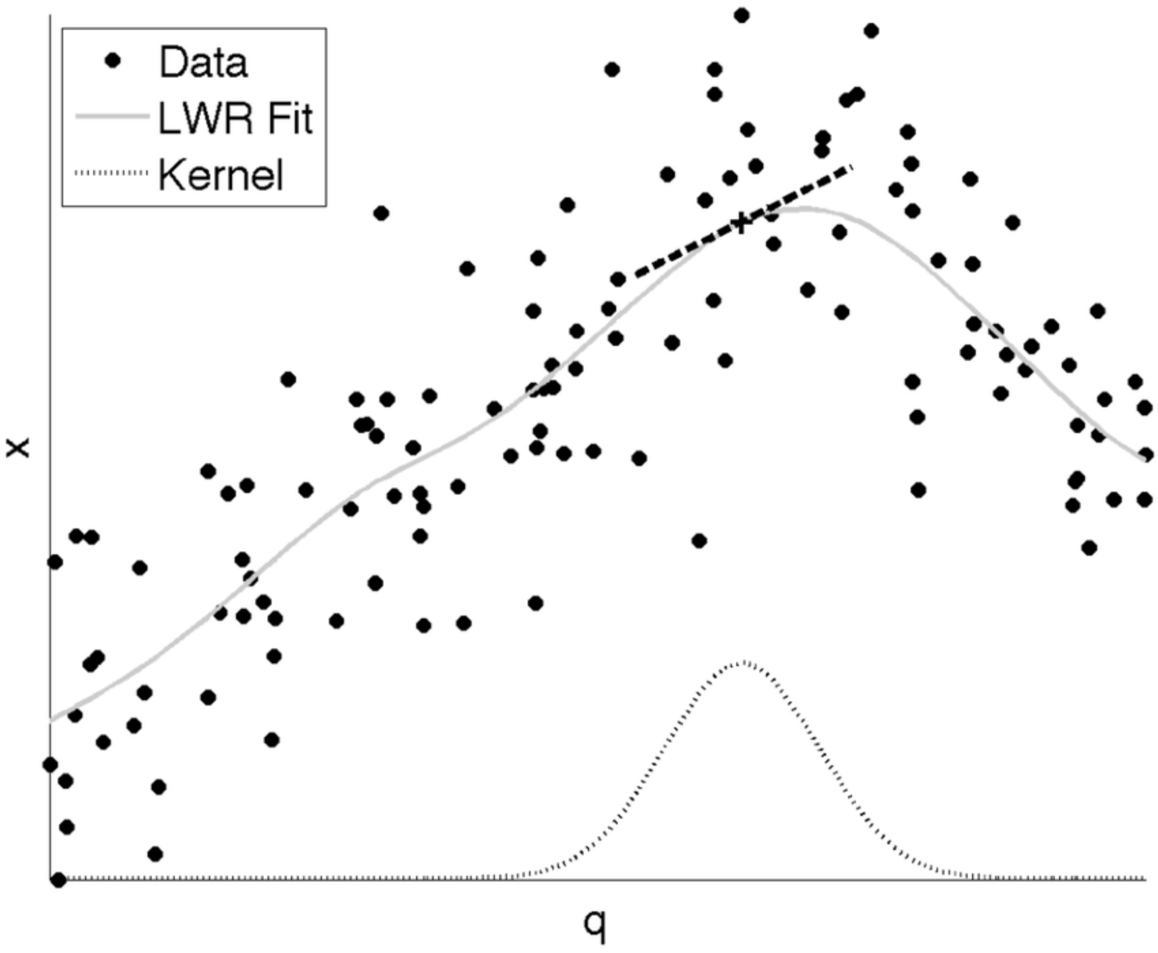


Figure 3.3: Locally Weighted Regression

3.6 Ensembling Frameworks

In statistics and machine learning, ensemble ways in which use multiple learning algorithms to induce higher second-sighted performance than is also obtained from any of the constituent learning algorithms alone. A machine learning ensemble consists of a finite set of different models, however generally permits for way more versatile structure to exist among those alternatives. Figure 3.4 shows the rough plan of approach employed in this analysis. Boosting is an ensembling meta-algorithm for reducing bias and variance in supervised learning, and there are many machine learning algorithms that convert weak models to robust ones. the thought is that, as we have a tendency to introduce a lot of straightforward models, the model becomes a stronger and stronger predictor. In boosting nomenclature, the straightforward models area unit referred to as weak models or weak learners. Within the context of regression, we have a tendency to build numerical predictions ie, continuous worth prediction.

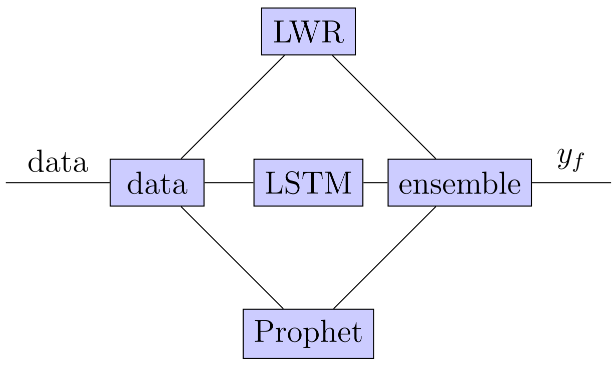


Figure 3.4: Ensemble system architecture

3.6.1 Gradient Boosted Regression

Training a regression model is a matter of fitting a function $f(x_d)$ where x_d is the feature vector, through the data points as best we can. Given a feature vector and scalar target value y for a single observation y_f as the addition of M weak models $f_m(x_d)$

$$y_f = \sum_{m=1}^{M} (f_m(x_d))$$

Gradient Boosting for regression builds an additive model just as explained above, along with that it allows for the optimization of arbitrary differentiable loss functions[8]. In each stage, a cost function is calculated, the negative gradient of the given loss function is used to update the weights assigned to the weak models.

3.7 Web development and Data Extraction

This section discuss the topics and frameworks used for web portal and data extraction performed for collecting dataset from KSNDMC official website

3.7.1 HTML

Hypertext Markup Language (HTML) is the standard markup language for creating web pages and applications. Each page contains a series of connections to other pages called hyperlinks. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document. HTML elements are delineated by tags, written using angle brackets. Tags such as <img> and <input/> introduce content into the page directly. Others such as <p>. . . .</p> surround and provide information about document text and may include other tags as sub-elements.

Browsers do not display the HTML tags, but use them to interpret the content of the page. HTML code ensures the proper formatting of text and images so that your Internet browser may display them as they are intended to look.

3.7.2 Bootstrap

Bootstrap is a web framework that focuses on simplifying the development of informative web pages (as opposed to web apps). The primary purpose of adding it to a web project is to apply Bootstrap's choices of color, size, font and layout to that project. As such, the primary factor is whether the developers in charge find those choices to their liking. Once added to a project, Bootstrap provides basic style definitions for all HTML elements. The result is a uniform appearance for prose, tables and form elements across web browsers. In addition, developers can take advantage of CSS classes defined in Bootstrap to further customize the appearance of their contents.

3.7.3 Web Scraping

Web Scraping (also termed Screen Scraping, Web Data Extraction, Web Harvesting etc.) is a technique employed to extract large amounts of data from websites whereby the data is extracted and saved to a local file in your computer or to a database in table (spreadsheet) format. Data displayed by most websites can only be viewed using a web browser. They do not offer the functionality to save a copy of this data for personal use. The only option then is to manually copy and paste the data - a very tedious job which can take many hours or sometimes days to complete. Web Scraping is the technique of automating this process, so that instead of manually copying the data from websites, the Web Scraping software will perform the same task within a fraction of the time.

3.8 Machine Learning Tools

Tools are a big part of machine learning and choosing the right tool can be as important as working with the best algorithms.Machine learning tools are not just implementations of machine learning algorithms. They can be, but they can also provide capabilities that you can use at any step in the process of working through a machine learning problem.
There are many different software tools available to build machine learning models and to apply these models to new, unseen data. These tools typically contain libraries implementing some of the most popular machine learning algorithms. Following are

the major tools used in the project-

- SciKitLearn
- Keras
- Facebook's Prophet
- Flask

3.8.1 SciKit-learn

SciKit learn is an open source machine learning library built for python. Since its release in 2007, Scikit-learn has become one of the most popular open source machine learning libraries. Scikit-learn (also called sklearn) provides algorithms for many machine learning tasks including classification, regression, dimensionality reduction and clustering. It also provides utilities for extracting features, processing data and evaluating models. It provides in-built code for many of the popular machine learning algorithms. The documentation for scikit-learn is comprehensive, popular and well maintained. Sklearn is built on mature Python Libraries such as NumPy, SciPy, and matplotlib. It has a very active development community with regular update releases of the library.

3.8.2 Keras

Keras is an open-source neural-network library written in Python. It is capable of running on top of TensorFlow, Microsoft Cognitive Toolkit, R, Theano, or PlaidML. It is designed to enable fast experimentation with deep neural networks, it focuses on being user-friendly, modular, and extensible.
Keras contains numerous implementations of commonly used neural-network building blocks such as layers, objectives, activation functions, optimizers, and a host of tools to make working with image and text data easier to simplify the coding necessary for writing deep neural network code. In addition to standard neural networks, Keras has support for convolutional and recurrent neural networks. It supports other common utility layers like dropout, batch normalization and pooling.

3.8.3 Prophet

. It provides us with the ability to make time series predictions with good accuracy using simple intuitive parameters. Facebook has open-sourced its Prophet forecasting tool, designed to make it easier for experts and non-experts to make high-quality

forecasts.Prophet is an open source library published by Facebook that is based on decomposable (trend+seasonality+holidays) models. The code is available on GitHub in both Python and R.
Prophet is aimed specifically at business problems such as computer infrastructure capacity planning that have at least several months of data (preferably a year or more) and issues such as seasonality, "holidays" that can affect trends (such as Black Friday and Cyber Monday for retailers), and events that can have significant impacts (such as launching a new website when trying to forecast site traffic). Prophet can also handle some missing values and outliers.

3.8.4 Flask

Flask is a micro web framework written in Python. The microframework Flask is based on the Pocoo projects Werkzeug and Jinja2. It is classified as a microframework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself.
Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools. Extensions are updated far more frequently than the core Flask program. Applications that use the Flask framework include Pinterest and LinkedIn

Chapter 4

System Requirements Specification

A system requirements specification (SRS) is a description of a software system to be developed. It lays out functional and nonfunctional requirements, and may include a set of use cases that describe user interactions that the software must provide.
In order to fully understand ones project, it is very important that they come up with an SRS listing out their requirements, how are they going to meet it and how will they complete the project. SRS also functions as a blueprint for completing a project with as little cost growth as possible. SRS is often referred to as the parent document because all subsequent project management documents,such as design specifications, statements of work, software architecture specifications, testing and validation plans, and documentation plans, are related to it.
Requirement is a condition or capability to which the system must conform. Requirement Management is a systematic approach towards eliciting, organizing and documenting the requirements of the system clearly along with the applicable attributes. The elusive difficulties of requirements are not always obvious and can come from any number of sources.

4.1 Functional Requirements

Functional Requirement defines a function of a software system and how the system must behave when presented with specific inputs or conditions. These may include calculations, data manipulation and processing and other specific functionality. Following are the functional requirements on the system:

- Data will be extracted from KSNDMC official website
- Data will be cleaned and vectorized and sent to ML models.
- Models will be trained based on data collected from KSNDMC website[9] .

- Daily prediction of dam parameters and alerts will be displayed through web interface
- Visualization of dam parameters against various weather parameters over past years will be made available through web interface.

4.1.1 DataSet

KSNDMC : Karnataka State Natural Disaster Monitor Center is Drought Monitoring Cell (DMC) established in 1988 as an institutional mechanism to monitor the Drought. Activities broadened to also include monitoring other natural disasters and renamed as Karnataka State Natural Disaster Monitoring Centre (KSNDMC) in 2007.

The Master control centre of KSNDMC is established in 10 acres and at Major Sandeep Unnikrishnan Road, Near Yelahanka-Attur Layout in Bangalore. Now it also includes monitoring of other natural disasters viz., Floods, Hailstorms, Gale-Winds, Storm-surges, Earthquakes, Landslides, Tsunamis etc. KSNDMC prepares and disseminates Reports on Daily/Weekly/ Monthly/Seasonal and Annual at Hobli level and also provides Rainfall and Weather information at Gramapanchayath level. DataSet and daily data is extracted from official KSNDMC website **http://www.ksndmc.org/**

4.2 Non-Functional Requirements

Non functional requirements are the requirements which are not directly concerned with the specific function delivered by the system. They specify the criteria that can be used to judge the operation of a system rather than specific behaviours. They may relate to emergent system properties such as reliability, response time and store occupancy. Non functional requirements arise through the user needs, because of budget constraints, organizational policies and the need for interoperability with other software and hardware systems.

4.2.1 Product Requirements

- **Correctness:** Project must follow a well defined set of procedures and rules to engage a conversation with the user and a pre-trained machine learning model to compute, also rigorous testing is performed to confirm the correctness of the data.
- **Modularity:** The complete product is broken up into many modules and well defined interfaces are developed to explore the benefit of flexibility of the product.

- **Modularity:** The complete product is broken up into many modules and well defined interfaces are developed to explore the benefit of flexibility of the product.

- **Robustness:** This software is being developed in such a way that the overall performance is optimized and the user can expect the results within a limited time with utmost relevance and correctness.

Non functional requirements are also called the qualities of a system. These qualities can be divided into execution quality and evolution quality. Execution qualities are security and usability of the system which are observed during run time, whereas evolution quality involves testability,maintainability, extensibility or scalability.

4.2.2 Basic Operational Requirements

The customers are those that perform the primary functions of system engineering, with special emphasis on the operator as the key customer. Operational requirements will define the basic need and, at a minimum, will be related to these following points:-

- **Mission profile or scenario :** It describes the procedures used to accomplish mission objective. It also finds out the effectiveness or efficiency of the system. In our case a web portal with all functional requirements satisfied.

- **Performance and related parameters :** It points out the critical system parameters to accomplish the mission. The response time of the predictions, data visulization, effectiveness in how to convey the alerts and suggestions, along with a responsive user interface.

- **Utilization environments :** It gives a brief outline of system usage. Finds out appropriate environments for effective system operation. The required tools and frameworks like keras, facebook prophet, flask and also web frameworks like bootstrap, jquery and selenium(for web scraping)

- **Operational Life Cycle :** It defines the system lifetime. In our case the life cycle continues until the performance of the model is not degraded. In such case the models must be retrained for g=better performance.

4.2.3 System Configurations

Hardware system Configuration

- Processor : 1.8 GHz or faster processor. Dual-core(minimum)

- RAM : 4 GB(for development) 2GB or above
- Hard Disk : 20-25 GB of free space(for development)
- Display : 1024x768 minimum screen resolution.

Software system Configuration

- Operating System :Windows 10/8.1 or Linux or Linux server
- Coding Language : Python
- Tools and frameworks
  - Jupyter Notebook
  - PyCharm
  - BeautifulSoup and/or Selenium
  - ScikitLearn
  - Keras
  - Facebook Prophet
  - TensorFlow backend
  - Flask(web Server)
  - Frontend Technologies(html, Bootstrap, Jquery)

Chapter 5

System Analysis

Analysis is the process of finding the best solution to the problem. System analysis is the process by which we learn about the existing problems, define objects and requirements and evaluates the solutions. It is the way of thinking about the organization and the problem it involves, a set of technologies that helps in solving these problems. Feasibility study plays an important role in system analysis which gives the target for design and development.

5.1 Feasibility Study

All systems are feasible when provided with unlimited resource and infinite time. Butunfortunately this condition does not prevail in practical world. So it is both necessaryand prudent to evaluate the feasibility of the system at the earliest possible time.

5.1.1 Economical Feasibility

This study is carried out to check the economic impact that the system will have on the organization. Since the project is Machine learning based, the cost spent in effectuating this project would not demand cost for softwares and related products, as most of the products are open source and free to use. Hence the project would consumed minimal cost and is economically feasible.

5.1.2 Technical Feasibility

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Since machine learning algorithms is based on pure math there is very less requirement for any professional software. And also most of the tools are open source. The best part is that we can run this software in any system without any software requirements which makes them highly portable. Also most of the documentation and tutorials make easy to learn the technology

5.1.3 Social Feasibility

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The main purpose of this project which is based on the dam inflow is to prevent flooding by predicting the inflow and thereby fluctuating the outflow according to the needs necessary at that moment. Thus, this is a noble cause for the sake of the society, a small step taken to achieve a secure future.

5.2 Analysis

5.2.1 Performance Analysis

Most of the software we use is open source and free. The models which we use in this software,learn only once ,i.e once they are trained they need not be again fed in for the training phase. One can directly predict for values, hence time-complexity is very less. Therefore this model is temporally sound.

5.2.2 Technical Analysis

As mentioned earlier,the tools used in building this software is open source. Each tool contains simple methods and the required methods are overridden to tackle the problem. The presentation layer helps one to use the software with ease.

5.2.3 Economical Analysis

The completion of this project is not free of cost in its entirety. Even though the software used in building the model is free of cost, there are a few expenditures which is spent. Some of them include, the travelling to KSNDMC organization in Bangalore, the server for storing the data, hosting web application etc.

Chapter 6

System Design

6.1 System Development Methodology

System Development methodology is the development of a system or method for a unique situation. Having a proper methodology helps us in bridging the gap between the problem statement and turning it into a feasible solution. It is usually marked by converting the System Requirements Specifications (SRS) into a real world solution. System design takes the following inputs: Statement of work, Requirement determination plan, Current situation analysis. Proposed system requirements including a conceptual data model and metadata (data about data). The development method followed in this project is waterfall model. Although waterfall model is used as project model, a spiral model is used for ML model selection and development implementation.

6.1.1 Model Phases

The waterfall model is a sequential software development process, in which progress is seen as flowing steadily downwards (like a waterfall) through the phases of Requirement initiation, Analysis, Design, Implementation, Testing and maintenance.

Requirement Analysis: This phase is concerned about collection of requirement of the system. This process involves generating document and requirement review.

System Design: Keeping the requirements in mind the system specifications are translated in to a software representation. In this phase the designer emphasizes on:- algorithm, data structure, software architecture etc.

Coding: In this phase programmer starts his coding in order to give a full sketch of product. In other words system specifications are only converted in to machine

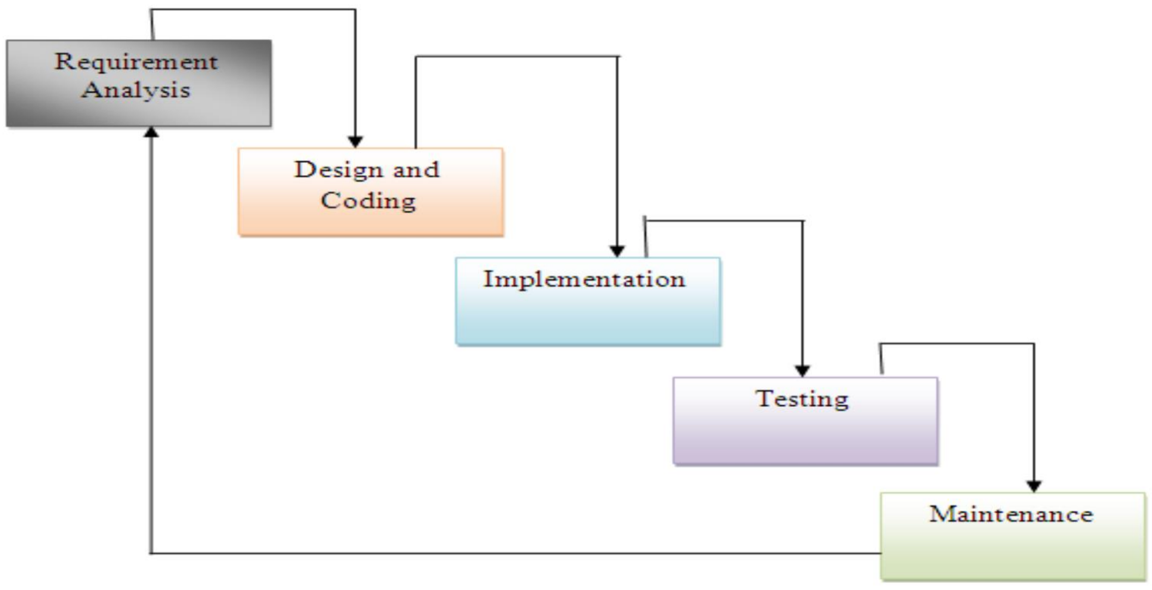


Figure 6.1: Waterfall Model

readable compute code.

Implementation: The implementation phase involves the actual coding or programming of the software. The output of this phase is typically the library, executables, user manuals and additional software documentation

Testing: In this phase all programs (models) are integrated and tested to ensure that the complete system meets the software requirements. The testing is concerned with verification and validation.

Maintenance: The maintenance phase is the longest phase in which the software is updated to fulfill the changing customer needs, adapt to accommodate changes in the external environment, correct errors and oversights previously undetected in the testing phase, enhance the efficiency of the software.

6.1.2 Advantages of Waterfall model

- Clear project objective
- Stable project requirements
- Progress of system is measurable.
- Logic of software development is clearly understood.
- Better resource allocation.

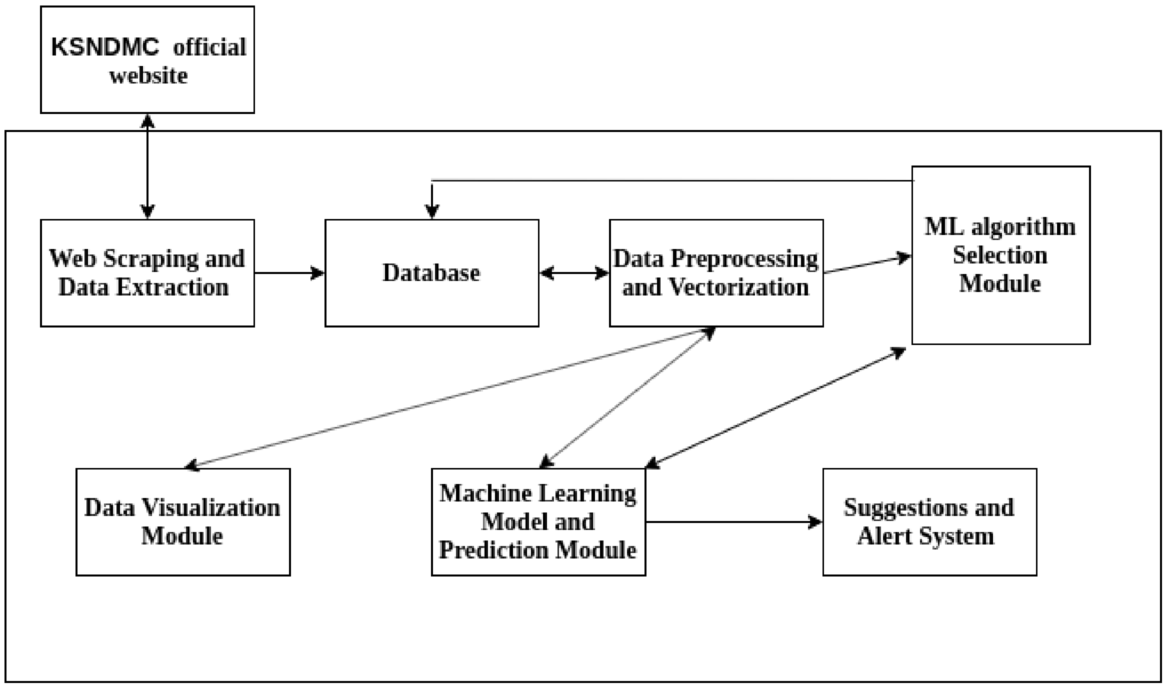

KSNDMC official website
Web Scraping and Data Extraction
Database
Data Preprocessing and Vectorization
ML algorithm Selection Module
Data Visualization Module
Machine Learning Model and Prediction Module
Suggestions and Alert System

Figure 6.2: System Architecture

6.2 System Architecture

A system architecture is a conceptual model using which we can define the structure and behaviour of that system. It is a formal representation of a system. Depending on the context, system architecture can be used to refer to either a model to describe the system or a method used to build the system. Building a proper system architecture helps in analysis of the project, especially in the early stages. Figure 6.2 depicts the system architecture and is explained in the following section.

- **KSNDMC :** The dataSet is obtained from official website of Karnataka State Natural Disaster Monitor Center whose operation is explained in aforementioned chapters.
- **Web Scraping and Data Extraction :** The module will use suitable web scraping tools to collect the dataset from aforementioned website. The extracted data will be cleaned at preliminary level and will be stored in database.
- **Database :** The dataset and files are of limited size, considering the fact that there are no transactional overhead or necessity for complex storage architecture **Unix file system** is used as database or in other words Storage system and all data are just stored as files.

- **Data Preprocessing and Vectorization :** Data preprocessing is a data mining technique that involves transforming raw data into an understandable format. Real-world data is often incomplete, inconsistent and is likely to contain many errors. Data preprocessing is a method used to resolve such issues. Since the data is not raw data but will be processed by KSNDMC before uploading vectorization will be on spotlight. DataSet will be passed to further modules in form of vectors for increasing computational efficiency.

- **ML algorithm selection module :** Multiple ML models are compared and reviewed to provide best result. Although it is run iteratively to get best results, once model is trained it is not changed unless the dataset has grown atleast 200% its size.

- **ML prediction module :** Daily data of weather and Dam parameters will be fed to the module for prediction of dam parameters and suggestions.

- **Suggestions and alert System :** The module collects the predicted information and generates suitable alerts and suggestions based on the received information.

- **Data Visualization module :** The module will contain visual comparison of dam parameters with various weather parameters over the past years. Visualization frameworks like MatplotLib will be used.

6.3 Sequence Diagram

A Sequence diagram is an interaction diagram that shows how processes operate with one another and what is their order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called event diagrams or event scenarios.

Sequence diagram are an easy and intuitive way of describing the behavior of a system by viewing the interaction between the system and the environment. A sequence diagram shows an interaction arranged in a time sequence. A sequence diagram has two dimensions: vertical dimension represents time, the horizontal dimension represents the objects existence during the interaction.

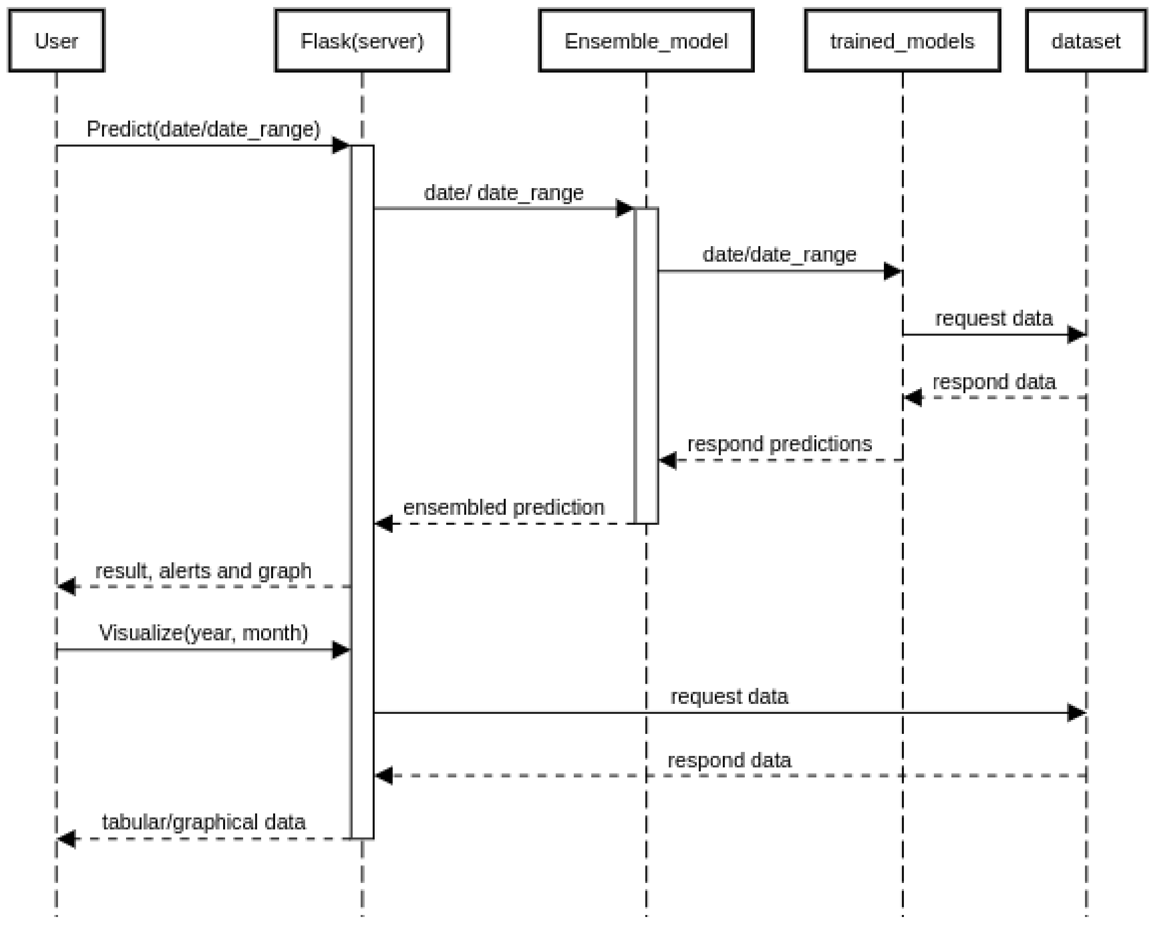


Figure 6.3: Sequence diagram: Prediction and Visualization

The sequence diagram depicted in figure 6.3 shows sequential process for two different transactions where user interacts with the web server. One of them being prediction for a given date or for a date range. As seen in the figure, request is sent to an instance of ensemble model which invokes all the trained base models. These base models collects the dataset for respective dates and predicts the inflow. Based on inflow predictions ensemble model instance predicts the reservoir parameters which is used by web server to generate results, graphs and alerts. The second transaction is data visualization for user specified duration(year and month). The server directly requests the relevant data and generates tabular or graphical representation based on user's decision.

Chapter 7

Implementation

Implementation of aforementioned system architecture is nothing but implementing all the modules, but in real scenario many modules overlap with each other. For example Suggestion and alert module was integrated with prediction yet holding its modularity. Following are the tasks performed to realize the project.

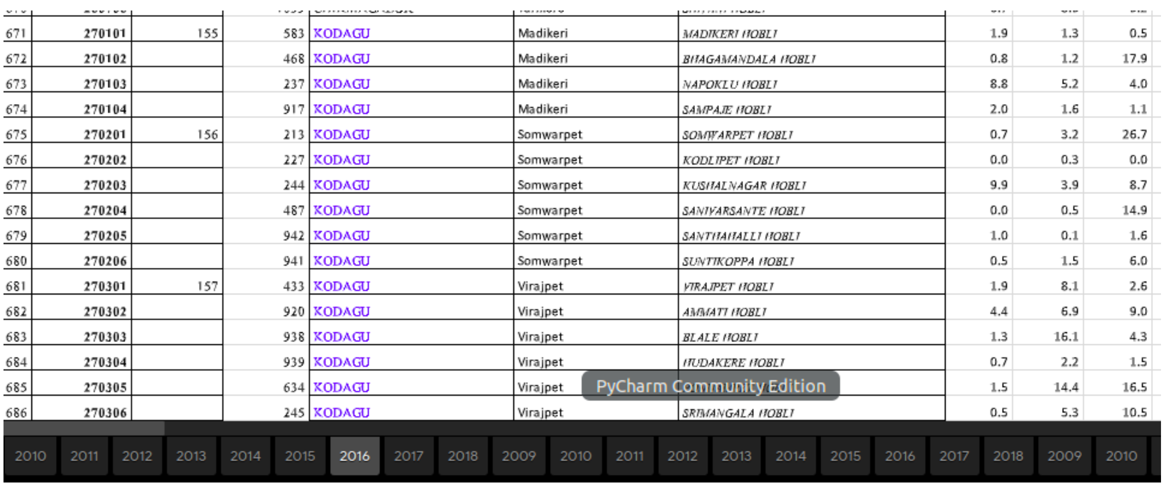
7.1 Data Collection and Preprocessing

7.1.1 Data Collection

The data collection is performed in two ways. The first is requesting rainfall dataset from KSNDMC directly. This process involved email conversation with KSNDMC officials, submission of formal request letter acknowledge by project guide and HoD, personal visit to KSNDMC, Bengaluru, Karnataka. The data was sent over email in the Excel Sheet format. Figure 7.1 shows a snippet of dataset.

The reservoir details were collected by web scraping techniques using Selenium web driver. The program performs some automated tasks which involved launching chome web browser, reaching KSNDMC website, applying required parameters and r=finally recording the required details for respective reservoir. Following is the code snippet for **Scraping reservoir details**.

```
from selenium import webdriver
import csv
csv_file = open('Harangi2019.csv', 'w')
browser = webdriver.
Chrome('/home/kishora/Desktop/chromedriver')
browser.get('https://www.ksndmc.org/Reservoir_Details.aspx')
browser.implicitly_wait(30)
```



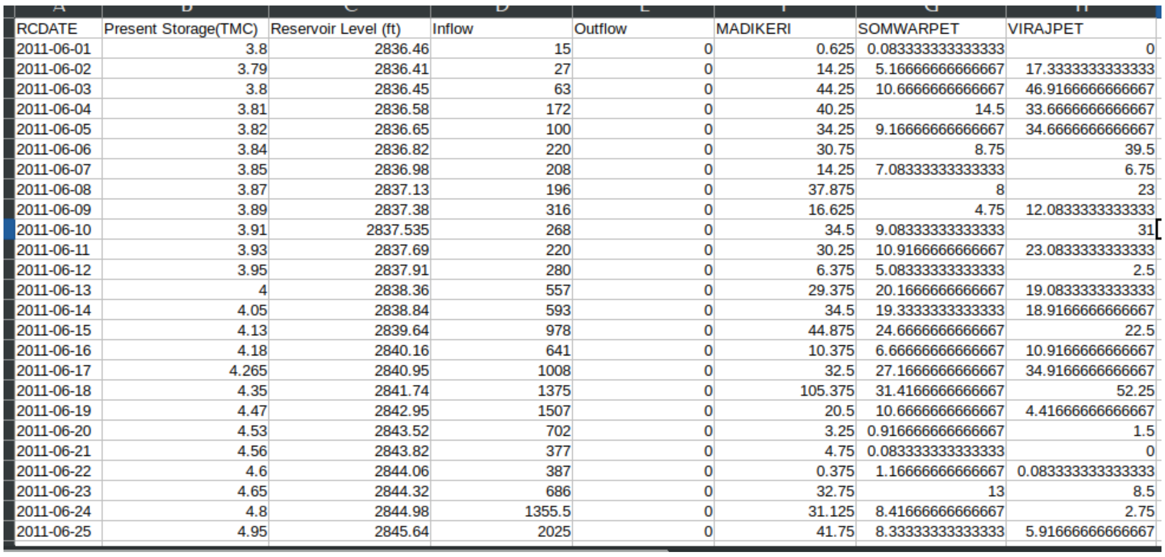
671	270101	155	583	KODAGU	Madikeri	MADIKERI HOBLI	1.9	1.3	0.5
672	270102		468	KODAGU	Madikeri	BHAGAMANDALA HOBLI	0.8	1.2	17.9
673	270103		237	KODAGU	Madikeri	NAPOKLU HOBLI	8.8	5.2	4.0
674	270104		917	KODAGU	Madikeri	SAMPAJE HOBLI	2.0	1.6	1.1
675	270201	156	213	KODAGU	Somwarpet	SOMWARPET HOBLI	0.7	3.2	26.7
676	270202		227	KODAGU	Somwarpet	KODLIPET HOBLI	0.0	0.3	0.0
677	270203		244	KODAGU	Somwarpet	KUSHALNAGAR HOBLI	9.9	3.9	8.7
678	270204		487	KODAGU	Somwarpet	SANIVARSANTE HOBLI	0.0	0.5	14.9
679	270205		942	KODAGU	Somwarpet	SANTHAHALLI HOBLI	1.0	0.1	1.6
680	270206		941	KODAGU	Somwarpet	SUNTIKOPPA HOBLI	0.5	1.5	6.0
681	270301	157	433	KODAGU	Virajpet	VIRAJPET HOBLI	1.9	8.1	2.6
682	270302		920	KODAGU	Virajpet	AMMATI HOBLI	4.4	6.9	9.0
683	270303		938	KODAGU	Virajpet	BLALE HOBLI	1.3	16.1	4.3
684	270304		939	KODAGU	Virajpet	HUDAKERE HOBLI	0.7	2.2	1.5
685	270305		634	KODAGU	Virajpet	[illegible]	1.5	14.4	16.5
686	270306		245	KODAGU	Virajpet	SRIMANGALA HOBLI	0.5	5.3	10.5

Figure 7.1: Dataset from KSNDMC

```
selctions = browser.
find_element_by_xpath("//select[@class = 'resback']")
selctions.
find_element_by_xpath("//option[@value = 'Harangi']").click()
browser.implicitly_wait(10)
year = browser.
find_element_by_xpath("//select[@class = 'yearback']")
year.
find_element_by_xpath("//option[@value = '2018']").click()
browser.implicitly_wait(30)
```

7.1.2 Preprocessing

Data preprocessing is an important step as it helps in cleaning the data and making it suitable for use in machine learning algorithms. Most of the focus in preprocessing is to remove any outliers or erroneous data, as well as handling any missing values. Missing data can be dealt with in two ways. The first method is to simply remove the entire row which contains the missing or erroneous value. While this an easy to execute method, it is better to use only on large datasets. Using this method on small datasets can reduce the dataset size too much, especially if there are a lot of missing values. This can severely affect the accuracy of the result. Since ours is a relatively small dataset, we will not be using this method. One more reason for not dropping rows is the fact that the project depends on time series data and hence removing rows will affect the results. Interpolation methods are used in cases where it is relevant and in other cases the missing data is replaced with zeros.



RCDATE	Present Storage(TMC)	Reservoir Level (ft)	Inflow	Outflow	MADIKERI	SOMWARPET	VIRAJPET
2011-06-01	3.8	2836.46	15	0	0.625	0.0833333333333333	0
2011-06-02	3.79	2836.41	27	0	14.25	5.16666666666667	17.3333333333333
2011-06-03	3.8	2836.45	63	0	44.25	10.6666666666667	46.9166666666667
2011-06-04	3.81	2836.58	172	0	40.25	14.5	33.6666666666667
2011-06-05	3.82	2836.65	100	0	34.25	9.16666666666667	34.6666666666667
2011-06-06	3.84	2836.82	220	0	30.75	8.75	39.5
2011-06-07	3.85	2836.98	208	0	14.25	7.08333333333333	6.75
2011-06-08	3.87	2837.13	196	0	37.875	8	23
2011-06-09	3.89	2837.38	316	0	16.625	4.75	12.0833333333333
2011-06-10	3.91	2837.535	268	0	34.5	9.08333333333333	31
2011-06-11	3.93	2837.69	220	0	30.25	10.9166666666667	23.0833333333333
2011-06-12	3.95	2837.91	280	0	6.375	5.08333333333333	2.5
2011-06-13	4	2838.36	557	0	29.375	20.1666666666667	19.0833333333333
2011-06-14	4.05	2838.84	593	0	34.5	19.3333333333333	18.9166666666667
2011-06-15	4.13	2839.64	978	0	44.875	24.6666666666667	22.5
2011-06-16	4.18	2840.16	641	0	10.375	6.66666666666667	10.9166666666667
2011-06-17	4.265	2840.95	1008	0	32.5	27.1666666666667	34.9166666666667
2011-06-18	4.35	2841.74	1375	0	105.375	31.4166666666667	52.25
2011-06-19	4.47	2842.95	1507	0	20.5	10.6666666666667	4.41666666666667
2011-06-20	4.53	2843.52	702	0	3.25	0.916666666666667	1.5
2011-06-21	4.56	2843.82	377	0	4.75	0.0833333333333333	0
2011-06-22	4.6	2844.06	387	0	0.375	1.16666666666667	0.0833333333333333
2011-06-23	4.65	2844.32	686	0	32.75	13	8.5
2011-06-24	4.8	2844.98	1355.5	0	31.125	8.41666666666667	2.75
2011-06-25	4.95	2845.64	2025	0	41.75	8.33333333333333	5.91666666666667

Figure 7.2: Preprocessed Dataset

Converting the dataset into time series data is the most important step in the project. Every data feature of converted into event based where each event is every day. The rain dataset is received with dates as columns, which are converted to dated rows. At the end of this step, dataset will be date indexed. Figure 7.2 shows the preprocessed dataset in the time series based format ie with date index.

Following code snippet shows **Data Preprocessing**

```
import pandas as pd
import numpy as np
d=pd.read_csv('/Datasets/Harangi2019.csv')
d['Flow Date'] = pd.to_datetime
(d['Flow Date'],errors='coerce',dayfirst=True)
damInput = pd.DataFrame()
damInput = d
damInput.set_index(d['Flow Date'],inplace=True)
damInput.drop(columns='Flow Date',inplace=True)
#refill empty dates,add missing rows with NaN value
idx = pd.date_range('2019-01-01', '2019-12-31')
damInput=damInput.reindex(idx, fill_value=np.nan)
#replace NaN with average/median
of previous and next value to make a linear flow
damInput.interpolate(method='linear', inplace=True)
```

7.2 Data Analysis

One of the important step in building Machine learning models is analysing the the data. This helps in identifying relationships between the various attributes present in the dataset and also to gain more knowledge on data which is essential in building efficient models. We used graphs to visualize the relationships between the data attributes.
Figure 7.3 depicts the dataset of 8 years from 2011 to 2018 for Harangi reservoir in Kodagu, Karnataka. The first two are the inflow and outflow of the reservoir for 8 years, we can compare it with the rainfall(mm) in Kodagu for all three taluks namely Madikeri, Somwarpet and Virajpet. It is helpful in understanding the relationships and how and to what extent the rainfall affects the reservoir inflow.

```
import pandas as pd
import numpy as np
import matplotlib as mpl
from matplotlib import pyplot as pt
from matplotlib.figure import Figure
from matplotlib import pyplot as plt
mpl.rcParams['figure.figsize'] = (30, 14)
mpl.rcParams['axes.grid'] = True

fig = Figure()
axis = fig.add_subplot(4, 1, 1)

df = pd.read_csv
('Datasets/allYearHarangi.csv',index_col=0)
ax = df.iloc[:,2:].plot(subplots=True)
plt.plot(X[:x.shape[0]],Y[:x.shape[0]])
plt.plot(X[x.shape[0]-1:],Y[x.shape[0]-1:])
```

Figure 7.4 is pictorial representation of resrvoir state for the year 2018 especially during the monsoon season. The inflow rises to 77000 Cusec which is way beyond the average daily inflow of harangi reservoir which lies between 1000 to 10,000 Cusec. This helps us in understanding the flood situation during that time.

```
import pandas as pd
import numpy as np
import matplotlib as mpl

mpl.rcParams['figure.figsize'] = (30, 14)
```

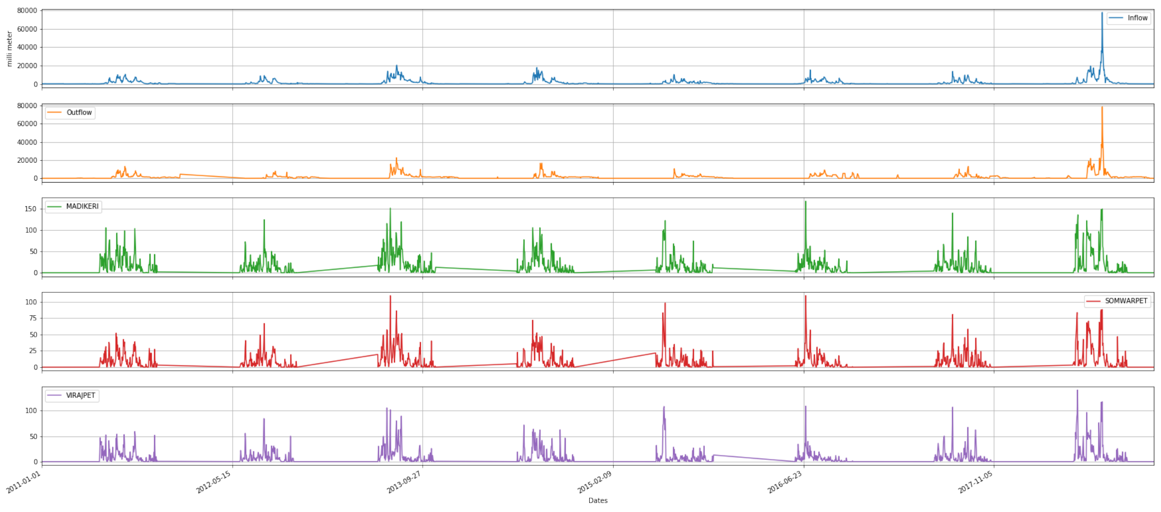


Figure 7.3: Visualization of dataset:Harangi

```
mpl.rcParams['axes.grid'] = True

damInput = pd.read_excel
('/Harangi.xlsx')
damInput.plot(colormap='Dark2')
damInput.plot(colormap='winter',subplots=True)
```

Figure 7.5 is also comparison of reservoir parameters like storage, reservoir height, inflow and outflow for the year 2018 to provide an insight of the data generated in that year.

7.3 Machine Learning Models

Various machine learning models were trained and evaluated for best results. Following sections shows the algorithms explored. For all the models, the dataset is split into training set and test set based on year. Out of 9 year dataset, first 7 years of data is used as training set and remaining two years data as test set. The dataset is converted into supervised time series dataset with 9 days lag based on model evaluation of lag 4, 9,10 and 12.

```
lag_val=9 #lag of 9 days
values = df.values
values = values.astype('float32')
reframed = series_to_supervised(values, lag_val, 1)
# drop columns we don't want to predict
reframed.drop
```

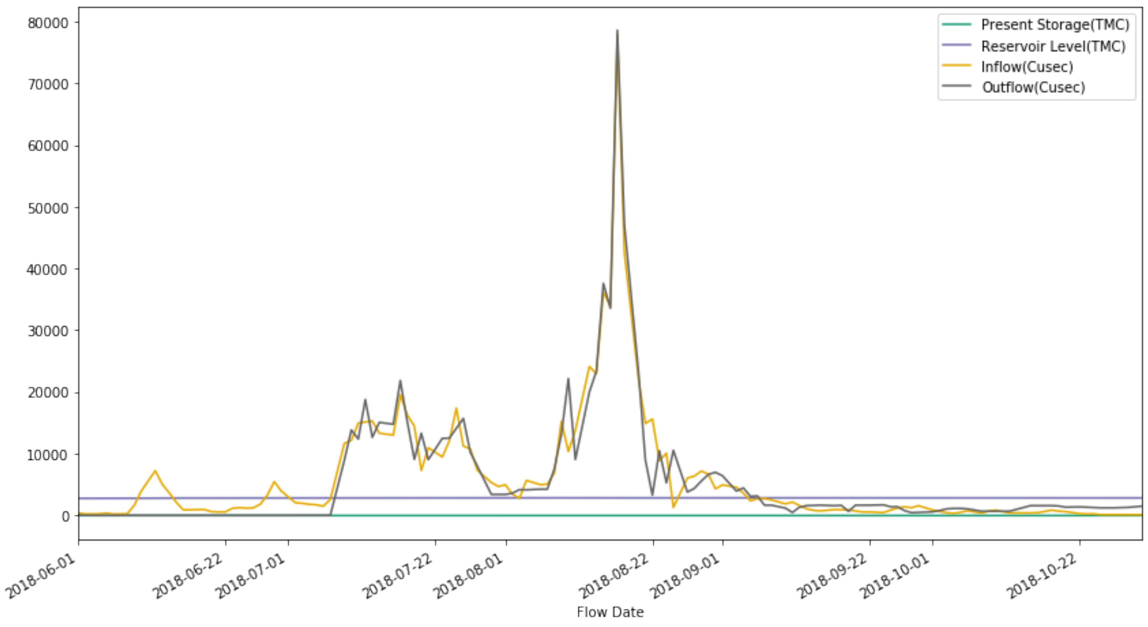


Figure 7.4: Floods in 2018

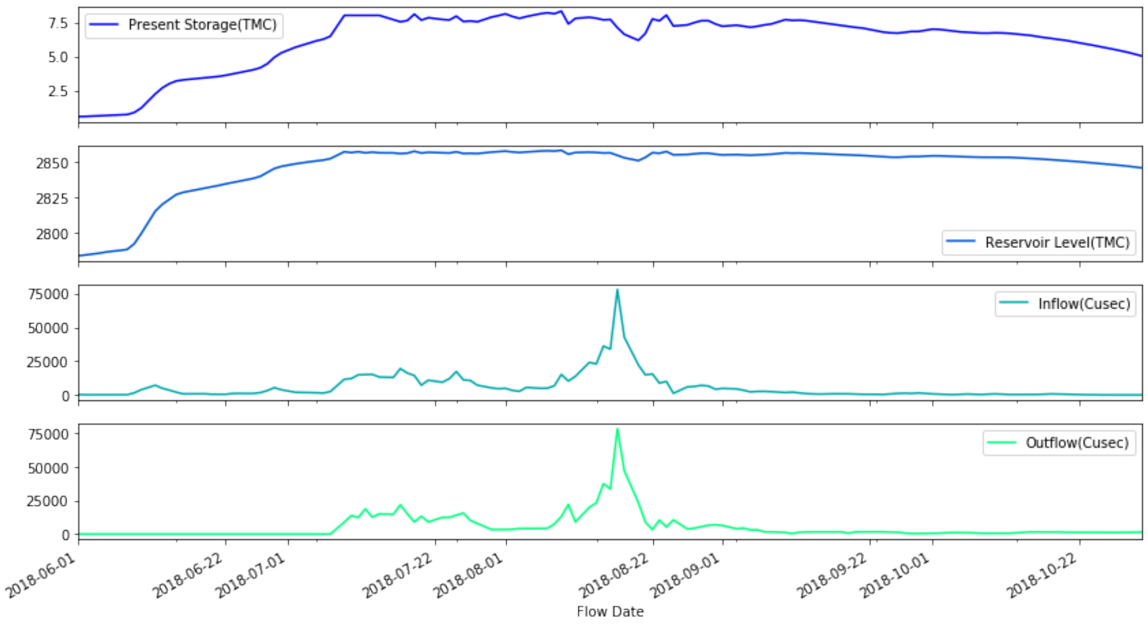


Figure 7.5: A comparison of reservoir parameters

```
(reframed.columns[[-1,-2,-3]], axis=1, inplace=True)
```

7.3.1 Recurrent Neural Networks with LSTM

The concepts behind RNNs and LSTMs are explained in aforementioned chapters. As we know LSTMs works best for time series problems, this algorithm was chosen. Keras framework with tensorflow backend is used to implement the algorithm, because tensorflow provides best platform for time consuming computations and works best for deep learning models.
A 9 days lag data is passed as input with 50 hidden lstm cells, Mean Absolute Error as loss function metrics along with ADAM optimizer to optimize the back-propagation(weight updation). The dataset is first converted to 9 days lag data, then split into train and test set. Then both train and test set is reshaped into three dimension data with lag days and number of features(columns) being the two dimensions and the other dimension being the value itself.
The following code snippet shows the **LSTM model** building and training along with reshaping the test and train set.

```
n_days_lag = 9
n_features = 4
# n_train_days = 365 * 7
train_X = train_x.reshape((train_x.shape[0], n_days_lag,
n_features))
test_X = test_x.reshape((test_x.shape[0], n_days_lag,
n_features))

print(train_X.shape, train_y.shape, test_X.shape,
test_y.shape)

Sequential()
model.add(LSTM(50,
input_shape=(train_X.shape[1], train_X.shape[2])))
model.add(Dense(1))
model.compile(loss='mae', optimizer='adam')
# fit network
history = model.fit(train_X, train_y, epochs=75,
batch_size=10, validation_data=(test_X, test_y),
verbose=2, shuffle=False)
```

```
# plot history
pyplot.plot(history.history['loss'], label='train')
pyplot.plot(history.history['val_loss'], label='test')
pyplot.legend()
pyplot.show()
```

Since the model is a deep learning model, especially RNN, it takes considerably long time to train the model given a large dataset. As the dataset is small, the system takes from few seconds to few minutes to train the model. But once the model is train the prediction part doesn't need much time, the output can be expected in few milli seconds.

7.3.2 Locally Weighted Regression

Locally Weighted Regression is a non-parametric algorithm, that is, the model does not learn a fixed set of parameters as is done in ordinary linear regression. It breaks the data-points into subsets and applies Linear Regression to the split subsets, and then augmented to improve the Prediction/Classification. The theory behind LWR is explained in previous chapters.
The LWR model is implemented instead of using a tool for applying it, so as to increase the learning process. It involves beta function, a kernel function and the main calling function named lwr. Following code snippet shows the python methods **implementing the Locally weighted regression**.

```
def lwr(x0, inp, out, k):
    m,n = np.shape(inp)
    ypred = np.zeros(m)
    ypred = x0 * beta(x0, inp, out, k)
    return ypred

def beta(point, inp, out, k):
    wt = kernal(point, inp, k)
    #print("The weight of betas is",wt)
    beta_val = (inp.T * (wt*inp)).I * inp.T * wt * out
    return beta_val

def kernal(point, inp, k):
    l,b = np.shape(inp)
    weights = np.mat(np.eye((l)))
    for i in range(l):
        diff = point - inp[i]
        weights[i,i] = np.exp

        (np.dot(diff, diff.T) / (-2.0 * (k**2)))
    return weights
```

Just like LSTM the dataset is first converted to 9 days lag data, then split into train and test set. As LWR is an instance based learning algorithm, regression will be applied to a set of nearest neighbours to predict the result. Thus to ease the process of searching nearest neighbours, the time series data is sorted based on months(seasonalities). This helps in improving the response time exceptionally. The code snippet shows the **Sorting of training set and LWR model trainig**.

```
idx = pd.date_range('2011-01-10', '2017-12-31')
reframed['Dates']=idx
reframed['month']=pd.DatetimeIndex
(reframed['Dates']).month
reframed=reframed.sort_values(by=['month','Dates'])
reframed.drop(columns=['month','Dates'],inplace=True)

ypred = []
train_X = train_x
train_y = Inflow
for i in test_x:
    ypred.append(lwr1(i, train_X, train_y, 7.15))
ypred = np.array(ypred).reshape
(Actuall_inflow.shape[0],1)
```

This model is a instance based learning model, it learns the model when prediction is requested. This results in zero training time. A. But during prediction,model takes a more time than a normal model as it has to search through the data points for nearest neighbours before predicting.

7.3.3 Facebook Prophet

Facebook has open-sourced its Prophet forecasting tool, designed to make it easier for experts and non-experts to make high-quality forecasts.Prophet is an open source library published by Facebook that is based on decomposable (trend+seasonality+holidays) models. The code is available on GitHub in both Python and R. This code is used to

implement time series prediction.
The dataset is first converted to 9 days lag data, but there is a catch. The dataset must be converted to a slightly different format so as to work with prophet. The prophet model expects only two columns as data, a date column and data that is to be predicted for a future date. But thankfully, additional features can be passed as input as additional regressors. The code snippet shows the conversion of normal data to time series data

```
def series_to_supervised(data, lag_days=1, lead_days=1,
dropnan=True):
    no_of_features = 1 if type(data) is list else data.shape[1]
    print(no_of_features)
    df = pd.DataFrame(data)
    cols, names = list(), list()
    # input sequence (t-n, ... t-1)
    for i in range(lag_days, 0, -1):
        cols.append(df.shift(i))
        names += [('var%d(t-%d)' % (j+1, i)) for j i
        range(no_of_features)]
    # forecast sequence (t, t+1, ... t+n)
    for i in range(0, lead_days):
        cols.append(df.shift(-i))
        if i == 0:
            names += [('var%d(t)' % (j+1)) for j in
            range(no_of_features)]
        else:
            names += [('var%d(t+%d)' % (j+1, i)) for
            in range(no_of_features)]
# put it all together
    agg = pd.concat(cols, axis=1)
    agg.columns = names
    # drop rows with NaN values
    if dropnan:
        agg.dropna(inplace=True)
    return agg
```

The code snippet is given below depicting **model design for Facebook Prophet**.

```
df = df.rename
```

```
(columns = {'Dates':'ds','var1(t)':'y'})

m = fb.Prophet(daily_seasonality = True)
m.add_seasonality(name='monthly', period=30, fourier_order=25)

for i in range(len(data1.columns)-1):
    m.add_regressor(data1.columns[i])

df_train = df.loc[df["ds"]<="2017-12-31"]
df_test  = df.loc[df["ds"]>="2018-01-01"]
m.fit(df_train)
```

Facebook prophet is a computational forecasting model developed to provide high performance. Evidently this model has extremely convincing performance in terms of training time as well as prediction time.

7.4 An attempt for Classification

In the following chapters we can notice that the RMSE for the predictions are extremely high, because of this fact new technique was applied to improve the accuracy. That was to convert regression into a classification problem. The classes were created based on data analysis performed on reservoir parameters which concluded that inflow above 11,000 cusec could cause overflow. Table 7.1shows the classification criteria. Although it had many drawbacks, this was a headstart to conviniently convey the alerts and suggestions.

Various classification algorithms were reviewed to gain better accuracy. This included LSTM classifiers, LWR classifiers, K-Means Clustering, Logistic Regression classification and Support Vector Machines. The results were measured in F1 score

Table 7.1: Classification criteria

Class	Inflow(between)
class 0	below 500
class 1	500 and 1000
class 2	1000 and 2500
class 3	2500 and 5000
class 4	5000 and 7000
class 5	7000 and 11000
class 6	11000 and 22000
class 7	above 22000

metrics (explained in following chapters) and the results were satisfactory. But then classifying the regression results based on aforementioned criteria provided with much better results, thus the classification algorithms weren't considered in the final model. The final prediction results provide both regression values with a confidence interval and classification class value to provide an insight of how critical the reservoir level is.

7.5 Ensembling the models

Before going to ensembling it must be understood that the trained models are dumped as pickle file using Pickle library provided by python as well as h5 module(by keras) for LSTM.

```
#storing model as h5
model.save('lstmTensorInf_forecast_model_lag9.h5')
#loading model from h5
lstm_lag9_model=load_model
('models/lstmTensorInf_forecast_model_lag9.h5')

import pickle
#storing model as pickle
with open('lwr_forecast_model_lag9.pckl', 'wb') as fout:
    pickle.dump(model, fout)
#loading model as pickle
with open
('models/lwrInf_forecast_model_lag9.pckl', 'rb') as
fin:
        lwr_lag9_model = pickle.load(fin)
    lwr_res = lwr_lag9_model.predict(lwr_test)
```

In statistics and machine learning, ensembling is the ways in which we use multiple learning algorithms to induce higher second-sighted performance than is also obtained from any of the constituent learning algorithms alone. In our case choosing one best model among the aforementioned algorithms weighed less than ensembling all models so as to get the best of all models. Even though it doesn't work in many cases, ensembling the models did improve the performance of the model. Grandient boosted regression is used as ensemble method. The prediction result given along with upper and lower bounds to provide a prediction interval with 95% confidence level instead of providing with single valued prediction.

Following is the code snippet for **Ensemble regression model**

```
from sklearn.ensemble import GradientBoostingRegressor
# Set lower and upper quantile
LOWER_ALPHA = 0.1
UPPER_ALPHA = 0.9
# Each model has to be separate
lower_model = GradientBoostingRegressor(loss="quantil
",alpha=LOWER_ALPHA)
# The mid model will use the default loss
mid_model = GradientBoostingRegressor(loss="ls")
upper_model = GradientBoostingRegressor(loss="quantile
",=UPPER_ALPHA)

X_train = inp[['lstm','proph','lwr']]
y_train = inp[['out']].values.reshape
(inp[['out']].shape[0],)
lower_model.fit(X_train, y_train)
mid_model.fit(X_train, y_train)
upper_model.fit(X_train, y_train)
# Record actual values on test set
predictions = pd.DataFrame(y_test)
# Predict
predictions['lower'] = lower_model.predict(X_test)
predictions['mid'] = mid_model.predict(X_test)
predictions['upper'] = upper_model.predict(X_test)
```

The ensembling model performs with reliable training time as well as prediction time as it is implemented using SciKitLearn libraries. But when it is used combined with all three prediction models, the prediction time is just the sum of all three base model's prediction time along with the time to generate the ensembled result.

7.6 Web Application

A web application was designed using flask server scripting (python) for user-model interaction. The web interface provides two features, one being prediction and other is data visualization. The prediction takes date or date range as input to provide prediction result.

Following is the code snippet for **serving prediction result** from flask

```
@app.route('/predict',methods=['GET', 'POST'])
def predictions():
    startDate = request.
    form.get('startdate',type=str)
    model = EnsembledRegression(startDate)
    if(request.form.get('enddate',type=str)!=''):
        endDate =
        request.form.get('enddate',type=str)
        results = model.predict(startDate, endDate)
        results.index = results.index.astype(str)
        figdata_png =
        base64.b64encode(plot_png(results.copy(),
        startDate,
        endDate).getvalue()).decode('ascii')
    else:
        results = model.predict(startDate)
        results.index = results.index.astype(str)
        figdata_png =
        base64.b64encode(plot_png(results.copy(),
        startDate).getvalue()).decode('ascii')

    results=results.round(decimals=2)
    results = results.drop(columns=['storage(tmc)Up',
    'storage(tmc)Low'])
    temp = results.to_dict('records')
    ind = results.index.values
    columnNames = results.columns.values[:-1]

    return render_template('tableTemplate.html',
    records=temp,
    colnames=columnNames,indexset=ind,plot=True,
    name=figdata_png)
```

The second feature is the data visualization option, user can choose to see data in a tabular manner or graphical. The data slice can be complete dataset, or a particular year. Along with that user can demand all 12 months data, data from monsoon(June to December) or non-monsoon months.

Following code snippet shows the functions **serving graphical data** to user.

```
@app.route('/visualize',methods=['GET', 'POST'])
def showViz():
    year = request.form.get('year',type=str)
    month = request.form.get('month',type=str)
    figdata_png =
    base64.b64encode(plot_viz_png(year,month)
    .getvalue()).decode('ascii')
    return render_template('imagetemplate.html',plot=
        True,name=figdata_png)

def plot_viz_png(year,month):
    fig = create_viz_figure(year,month)
    output = io.BytesIO()
    FigureCanvas(fig).print_png(output)
    output.seek(0)
    return output
```

Chapter 8

Testing

The program comprises several algorithms which are tested individually for accuracy. we check for the correctness of the program as a whole and how it performs.

8.1 Unit Testing

Unit tests focus on ensuring that the correct changes to the world-state take place when a transaction is processed. The business logic in transaction processor functions should have unit tests, ideally with 100 percent code coverage. This will ensure that you do not have typos or logic errors in the business logic. The various modules can be individually run from a command line and tested for correctness. The tester can pass various values, to check the answer returned and verify it with the values given to him/her. The other work around is to write a script, and run all the tests using it and write the output to a log file and using that to verify the results. We tested each of the algorithms individually and made changes in pre-processing accordingly to increase the accuracy.

8.2 System Testing

System Testing is a level of software testing where a complete and integrated software is tested. The purpose of this test is to evaluate the system's compliance with the specified requirements. System Testing is the testing of a complete and fully integrated software product. Usually, software is only one element of a larger computer-based system. Ultimately, software is interfaced with other software/hardware systems. System Testing is actually a series of different tests whose sole purpose is to exercise the full computer-based system.Two Category of Software Testing are Black Box Testing and White Box Testing. System test falls under the black box testing category of software testing. Different Types of System Testing:

- Usability Testing - Usability Testing mainly focuses on the users ease to use the application, flexibility in handling controls and ability of the system to meet its objectives.

- Load Testing - Load Testing is necessary to know that a software solution will perform under real-life loads.

- Regression Testing- - Regression Testing involves testing done to make sure none of the changes made over the course of the development process have caused new bugs. It also makes sure no old bugs appear from the addition of new software modules over time.

- Recovery Testing - Recovery testing is done to demonstrate a software solution is reliable, trustworthy and can successfully recoup from possible crashes.

- Migration Testing - Migration testing is done to ensure that the software can be moved from older system infrastructures to current system infrastructures without any issues.

8.3 Functional Testing

Functional Testing is also known as functional completeness testing, Functional Testing involves trying to think of any possible missing functions. Testers might make a list of additional functionalities that a product could have to improve it during functional testing. In addition, business process flows like data fields, predefined processes and successive processes must be considered for testing.

Chapter 9

Results and Analysis

9.1 Result Analysis

Although it may seem illogical to predict inflow only with rainfall patterns as the independent factor, during the monsoon season of every year the inflow is evidently dependent mostly on rainfall even though there exist other affecting factors. To provide with better results, the predicted values from ensembled regression are classified based on criteria mentioned on Table 7.1.

The dataset used is split into train and test set based on years. Each regression model ie LWR, LSTM and facebook prophet, is trained with data from 2011 to 2017,

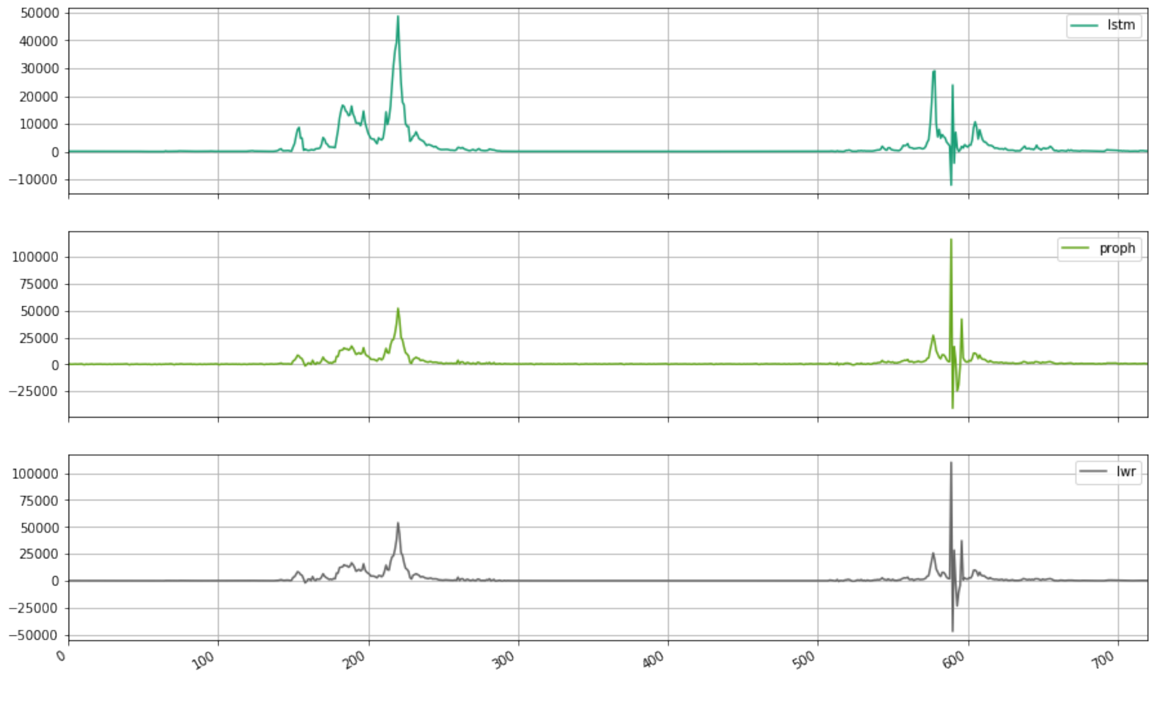


Figure 9.1: Results from LWR, LSTM and Prophet

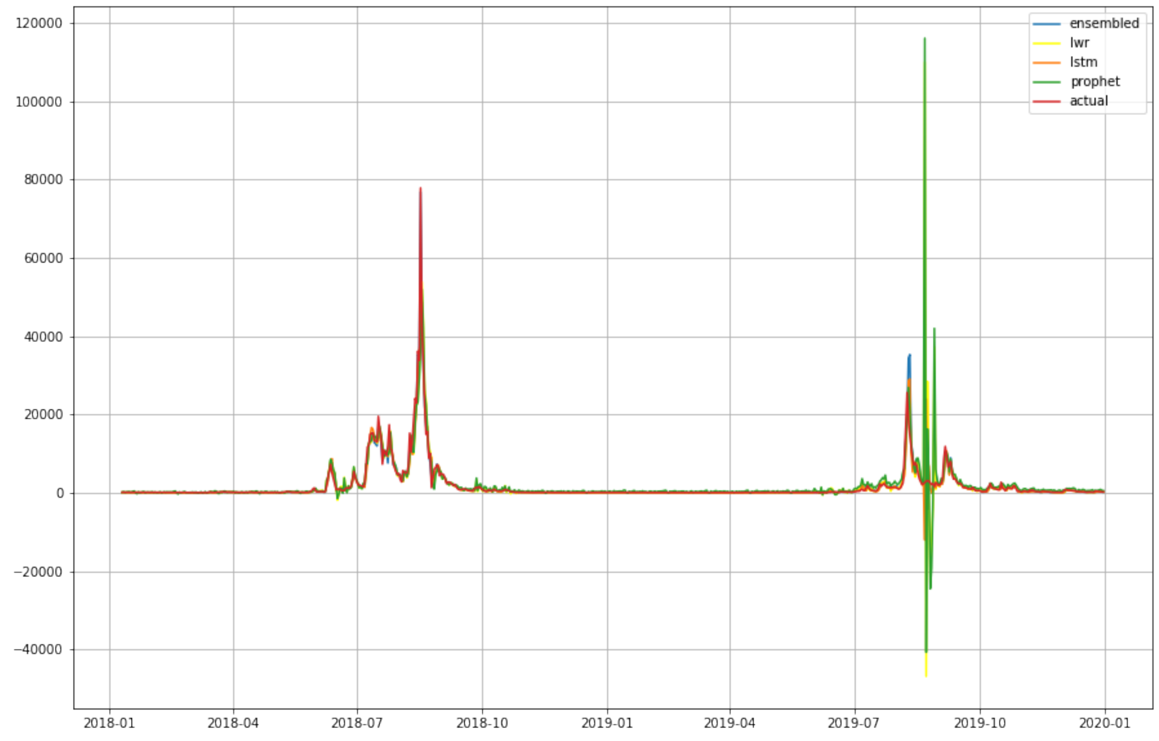


Figure 9.2: A Graphical comparison of results

and it is tested for two years, 2018 and 2019. The comparative visualization of results from all three models are represented in Figure 9.1. We can notice uneven variation in the prediction in 2019, it is an anomaly resulted in wrong prediction due to extremely heavy rain in Kodagu only for one day. These results are used to train and test the results of gradient boosted regression model which used as ensembling model for improving the results. Figure 9.2 depicts the comparison of results of base prediction models along with the ensembled one. The model is trained three different times to predict upper, result and lower inflow value with 95% confidence level so as to provide upper and lower bound creating a confidence interval for the results, which help in better understanding the results and gives an insight on model's confidence. Figure 9.3 shows the plot comparing actual inflow with upper, lower values of confidence interval and the predicted value to obtain a visual insight of how the prediction have improved in ensembled prediction results over the base model's predictions.

RMSE (Root mean Squared Error) is used as metrics to evaluate Regression output, whereas F1 or F score is used as metrics to evaluate the classification. Sometimes it's going to be fascinating to pick out a model with a lower accuracy as a result of it's a bigger prognostic power on the matter. For example, during a downside wherever there's an outsized category imbalance, a model will predict the worth of the bulk category for all predictions and succeed a high classification accuracy,but the down-

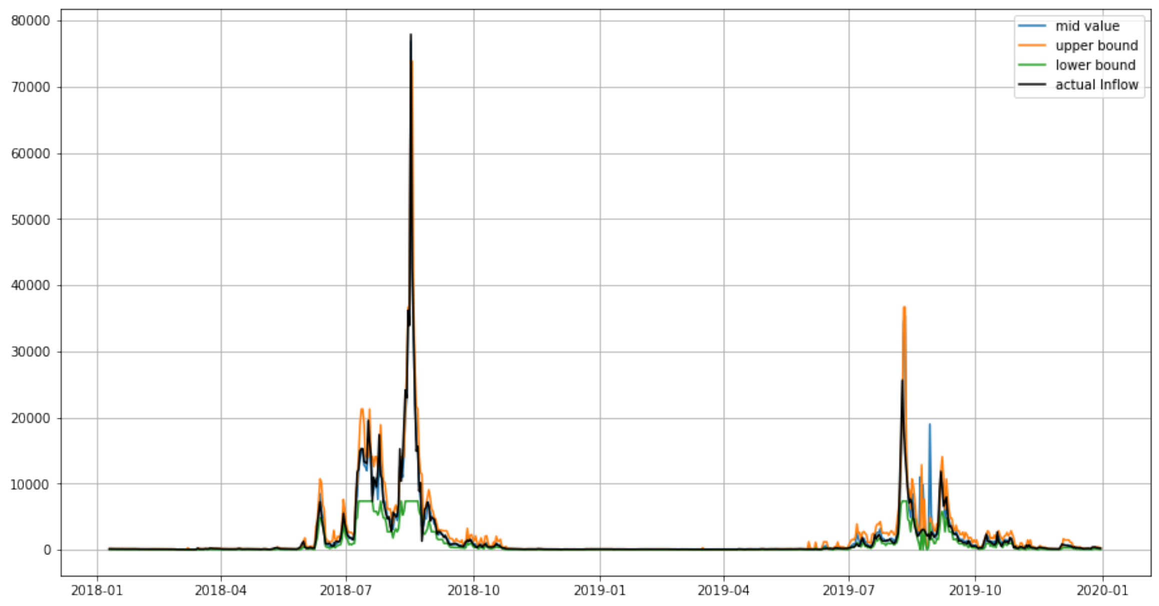


Figure 9.3: Ensembled Output

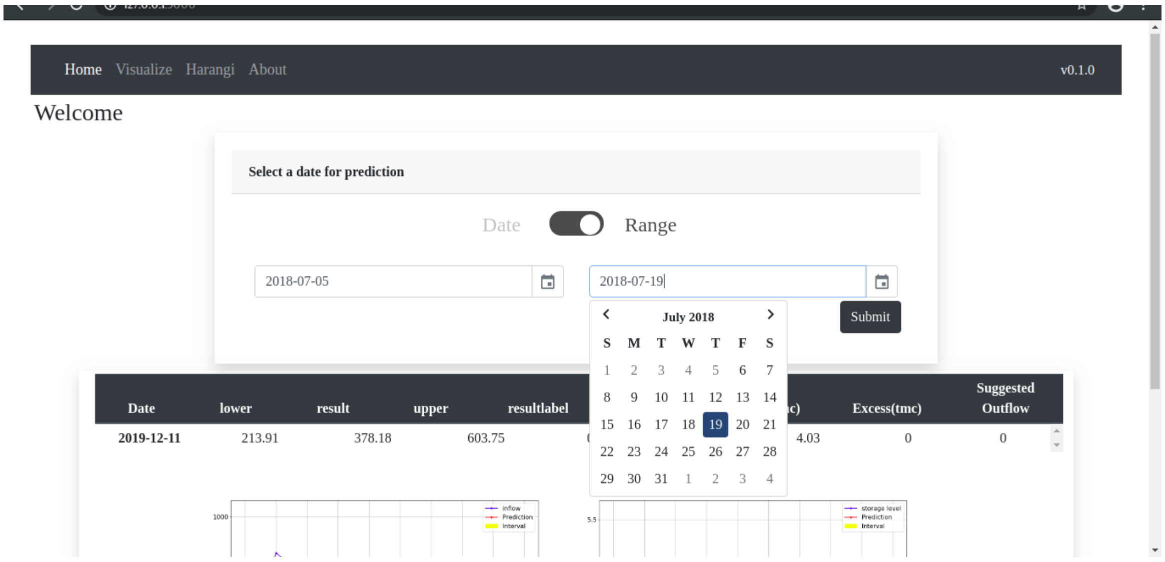
side is that this model isn't helpful within the problem domain. This is called the Accuracy Paradox. As we faced such a situation, these additional measures(precision, recall and F1 score) were used to evaluate a classifier. The harmonic mean of the precision and recall values is gives the F1 score. This efficiently reduces the effect of false positives in the evaluation.

Table 9.1 shows the results in terms of RMSE and F1 Score for regression and its

Table 9.1: Results obtained

Algorithm	RMSE	F1 Score
LSTM	2149	84%
LWR	4091	80%
Prophet	2411	74%
On Ensembling	1403	86%

corresponding classification respectively. We can observe that both RMSE and F1 Score are improved exceptionally on ensembling the results. F1 score of 84%, 80% and 74% for LWR, LSTM and facebook prophet are evidently improved to 86%.



Welcome

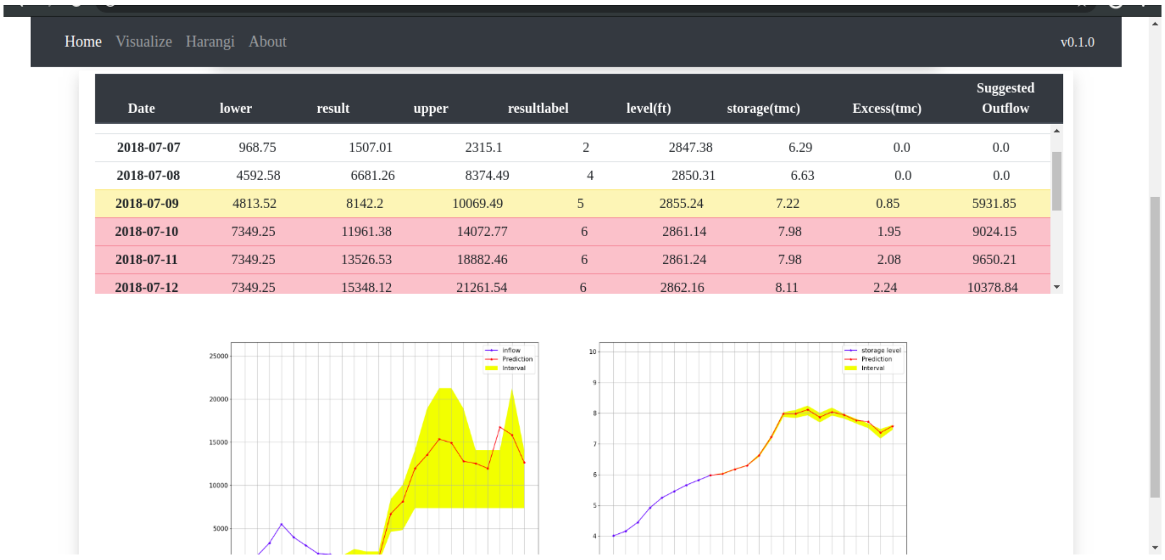
Select a date for prediction

2018-07-05

2018-07-19

Date	lower	result	upper	resultlabel	c)	Excess(tmc)	Suggested Outflow
2019-12-11	213.91	378.18	603.75	(	4.03	0	0

Figure 9.4: Prediction Date Interface

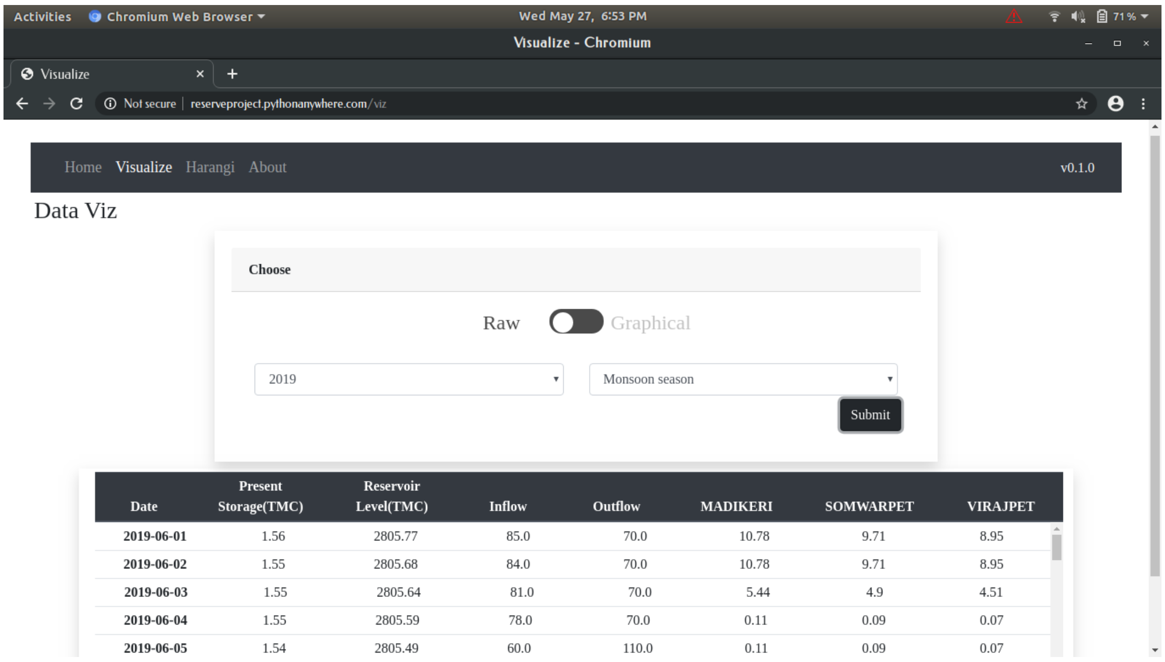


Date	lower	result	upper	resultlabel	level(ft)	storage(tmc)	Excess(tmc)	Suggested Outflow
2018-07-07	968.75	1507.01	2315.1	2	2847.38	6.29	0.0	0.0
2018-07-08	4592.58	6681.26	8374.49	4	2850.31	6.63	0.0	0.0
2018-07-09	4813.52	8142.2	10069.49	5	2855.24	7.22	0.85	5931.85
2018-07-10	7349.25	11961.38	14072.77	6	2861.14	7.98	1.95	9024.15
2018-07-11	7349.25	13526.53	18882.46	6	2861.24	7.98	2.08	9650.21
2018-07-12	7349.25	15348.12	21261.54	6	2862.16	8.11	2.24	10378.84

Figure 9.5: Output with Alerts and Graph

9.2 Snapshots

This section contains the snapshots of web application developed to communicate with the model with ease.



Data Viz

Choose

Raw Graphical

2019 | Monsoon season | Submit

Date	Present Storage(TMC)	Reservoir Level(TMC)	Inflow	Outflow	MADIKERI	SOMWARPET	VIRAJPET
2019-06-01	1.56	2805.77	85.0	70.0	10.78	9.71	8.95
2019-06-02	1.55	2805.68	84.0	70.0	10.78	9.71	8.95
2019-06-03	1.55	2805.64	81.0	70.0	5.44	4.9	4.51
2019-06-04	1.55	2805.59	78.0	70.0	0.11	0.09	0.07
2019-06-05	1.54	2805.49	60.0	110.0	0.11	0.09	0.07

Figure 9.6: Tabular Data

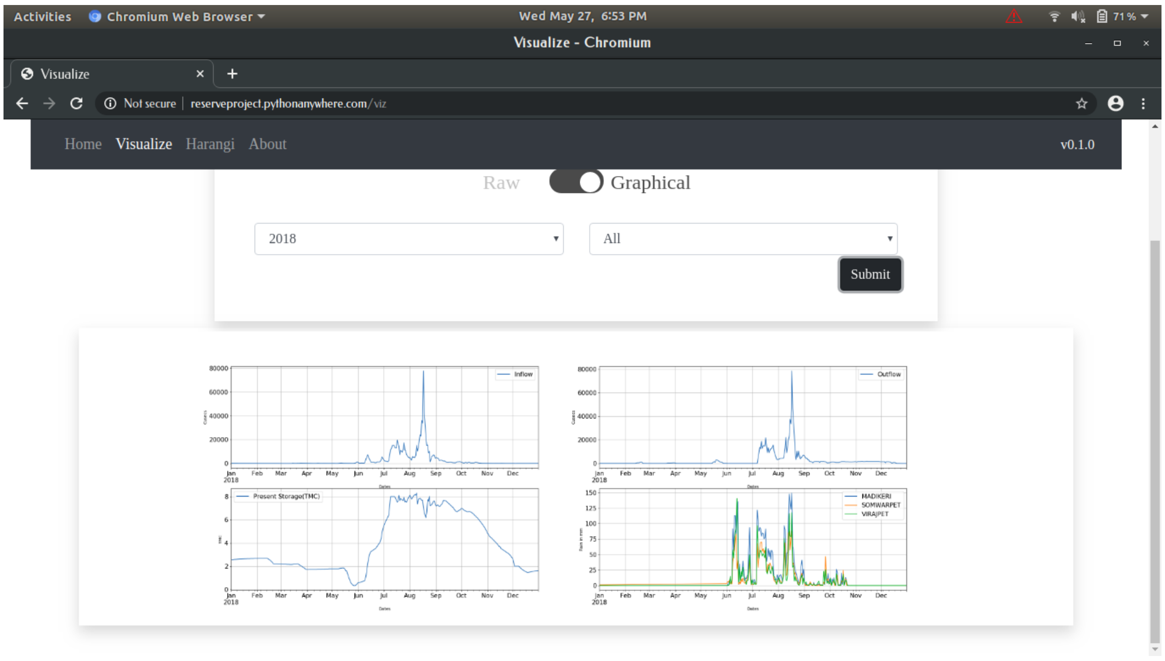
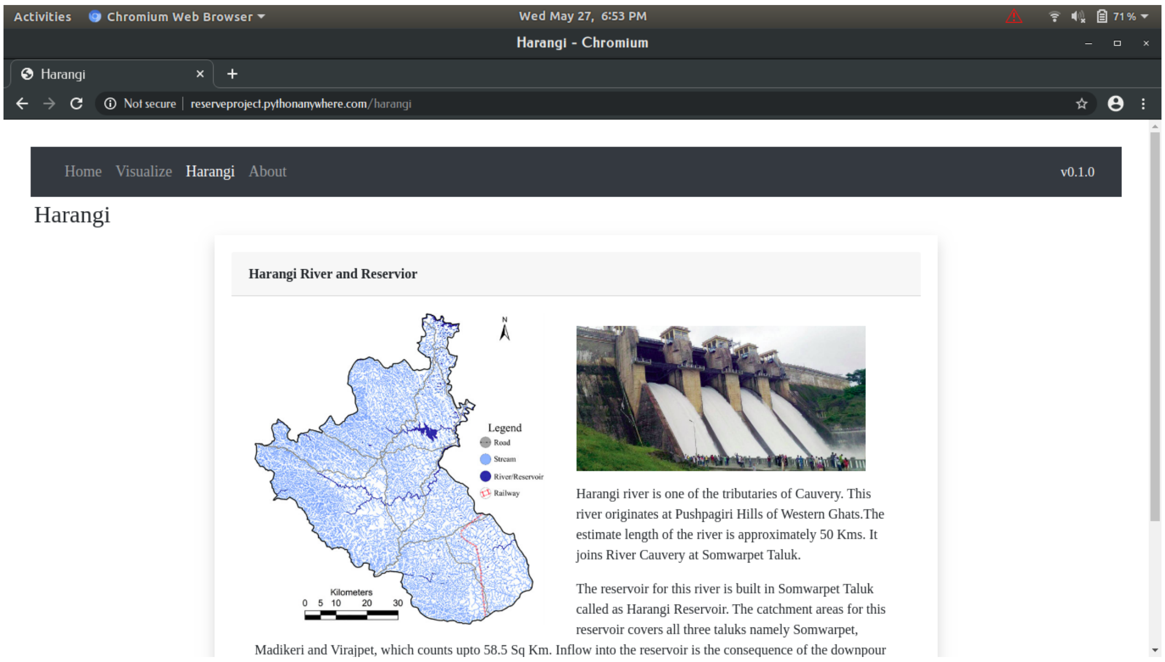


Figure 9.7: Graphical Data Visualization



Harangi

Harangi River and Reservior

Harangi river is one of the tributaries of Cauvery. This river originates at Pushpagiri Hills of Western Ghats.The estimate length of the river is approximately 50 Kms. It joins River Cauvery at Somwarpet Taluk.

The reservoir for this river is built in Somwarpet Taluk called as Harangi Reservoir. The catchment areas for this reservoir covers all three taluks namely Somwarpet, Madikeri and Virajpet, which counts upto 58.5 Sq Km. Inflow into the reservoir is the consequence of the downpour

Figure 9.8: About Harangi

Chapter 10

Conclusion and Future Scope

Although it takes a lot of ground work and analysis to identify all the dependent factors, collect relevant information, convert, extract and pre-process the data, so as to develop model with higher accuracy and precision, most of them affects the inflow only to a small extent making it hard to identify the factors that are actually crucial for prediction. This project used three models namely Locally Weighted Regression, Long Short Term Memory and Facebook Prophet with a limited number of dependent factors available to provide a dependable prediction model for better reservoir management. Despite their excellent performance their internal operations is not well understood. To get deeper insight into their internal working we followed and discussed the architectures and workings of the models used.
It was a great experience learning both deep learning and machine learning techniques as well as gaining the understanding on using the machine learning tools and frameworks. Nevertheless, Machine learning in general is an active area of research. Even though confusing, once the dimensionalities and vectorization concepts are well digested, developing the ML models could provide best results along with reasonable training time.
In future versions the model could be trained to further more reservoirs to provide management to further more reservoirs. Although it may include complex computation as multiple reservoirs join and seperates causing increased complexity, given enough time and resources the prediction models can be trained to give better results. Also various other deciding factors can be identified, collected and used for training to improve accuracy even though it takes lot of ground work and research.

Bibliography

[1] Tom M. Mitchell, **Machine Learning**, India Edition 2013, McGraw Hill Education.

[2] Warren S. Sarle . Neural Networks and Statistical Models Proceedings of the Nineteenth Annual SAS Users Group International Conference, April, 1994, SAS Institute Inc., Cary, NC, USA.

[3] Introduction to Multivariate Regression Analysis :A Review Paper HIPPOKRATIA 2010, 14 (Suppl 1): 23-28.

[4] Somchit Amnatsan, Sayaka Yoshikawa and Shinjiro Kanae, Improved Forecasting of Extreme Monthly Reservoir Inflow Using an Analogue-Based Forecasting Method: A Case Study of the Sirikit Dam in Thailand, 9 November 2018, Water 2018, 10, 1614; doi:10.3390/w10111614

[5] Tiantian Yang, Ata Akbari Asanjan, Edwin Welles, Xiaogang Gao, Soroosh Sorooshian and Xiaomang Liu, Developing reservoir monthly inflow forecasts using artificial intelligence and climate phenomenon information, May 2017, 10.1002/2017WR020482

[6] Sergey, Christian Szegedy (2015). "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift". ArXiv: 1502.03167 [cs.LG].

[7] Leo Breiman (1996). "BIAS, VARIANCE, AND ARCING CLASSIFIERS", Retrieved 19 January 2015."Arcing [Boosting] is more successful than bagging in variance reduction"

[8] https://explained.ai/gradient-boosting/L2-loss.html

[9] Karnataka State Natural Disaster Monitor Center official website : http://www.ksndmc.org/

[10] L R Medsker, L C Jain, "Recurrent Neural Networks, Design and Applications", 2001 CRC Press

[11] Locally Weighted Regression
https://medium.com/100-days-of-algorithms/day-97-locally-weighted-regression-c9cfaff087fb

[12] Deriving Normal Equations
https://ayearofai.com/rohan-3-deriving-the-normal-equation-using-matrix -calculus-1a1b16f65dda

[13] Regression Analysis of Annual Maximum Daily Rainfall andStream Flow for Flood Forecasting in Vellar River Basin P.Supriya a* , M.Krishnaveni a , M.Subbulakshmi . Aquatic Procedia 4 (2015) 1054 – 1061

[14] Soft computing approach for rainfall-runoff modelling: A review Vinay Chandwani*, Sunil Kumar Vyas, Vinay Agrawal, Gunwant Sharma (ICWRCOE 2015) Aquatic Procedia 4 (2015) 957 – 963

[15] R. J. Abrahart, L. M. See. Neural network modelling of non-linear hydrological relationships. Hydrology and Earth System Sciences Discussions, European Geosciences Union, 2007, 11 (5), pp.1563-1579. ffhal-00305094f

[16] Long-Term Rainfall Analysis and Runoff Estimation in Mountainous Watershed: A Case Study from Mhadei River Basin, Goa and Karnataka Manoj Ibrampurkar1* and A. G. Chachadi2 Gond. Geol. Mag., V. 27(2), December, 2012. pp. 153-158

[17] (2017). 'Supervised learning', Mathworks, [Online]. Available: https://se.mathworks.com/discovery/supervised-learning.html (accessed January 31, 2018).

[18] (2017). 'Unsupervised learning', Mathworks, [Online]. Available: https://se.mathworks.com/discovery/unsupervised-learning.html (accessed January 31, 2018).