

CHAPTER 1

INTRODUCTION

1.1 DROUGHT PREDICTION SYSTEM

Water shortage is one of the difficult issues in the cutting edge world. It influences all mainland's and around 3 billion individuals around the world in any event a month out of consistently. In excess of 15 percent of the absolute populace around the globe needs access to clean drinking water. The water emergency is expanding radically and has gotten one of the significant issues over the globe. A report proposes that the United States of America alone squanders 495 billion liters of new water every week. As just however one-hundredth of surface water is crisp and reasonable for human utilization, it turns out to be vital that we spare water all together that our people in the future endure. The unchecked use of water and extraordinary climate have intensified things and in a matter of seconds, there will be a water deficiency, anomalies in market interest, groundwater shrinkage, among different difficulties.

From time to time the updates on shortage of water in a specific region over the world has been a typical news bringing about an enormous number of passing's because of the absence of water inside the human body causing different infections, for example, drying out, and so forth.

Regardless of having the information on how much harm a dry spell can do to a nation or the world overall. The administration can do little to facilitate the agony of the individuals languishing. The losses of life in a dry season are disturbing and the instances of dry spells in pretty much every nation has gotten practically normal and are expanding each year. Taking the case of the 2016 dry season in India, one of the most noticeably awful dry seasons in history of the country influenced around 330 million individuals. Shortage doesn't just influence people straightforwardly yet in addition to numerous relative issues which can cause a chain of issues on the planet.

A dry spell in a specific zone extraordinarily influences the vegetation around there and for the most part pulverizes it. This causes a great deal of issues for the poor

segment of the general public that fundamentally contains labourers and furthermore influences the working effectiveness of the individuals.

Water shortage is one of the significant issues in the world. It as of now influences each landmass and around 2.8 billion individuals around the globe in any event one month out of consistently. More than 1.2 billion individuals need access to clean drinking water. The water emergency has gotten one of the significant worries over the globe. A report recommends that the only US squanders 7 billion gallons of drinking water every day. As just short of what one percent of earth's surface water is reasonable for human utilization, it becomes urgent that we spare water with the goal that our people in the future endure. The unchecked utilization of water and extraordinary climate conditions have compounded the circumstance and right away there will be a new water deficiency, abnormalities in organic market, groundwater shrinkage, among different difficulties.

From time to time the updates on shortage of water in a specific zone over the world has been a typical news bringing about an enormous number of passing because of absence of water to ascend in the costs of the day by day articles and along these lines the dry season in a solitary area of a nation influences the entire country. These evil impacts now and then spread to rather an enormous territory and result in a worldwide emergency.

The harm it dispenses on the individuals and the correct working of the general public can be minimalized with an expectation framework that could really tell the assessed measure of assets that will be expected to handle such a circumstance.

1.2 NEED OF THE PREDICTION SYSTEM

Drought is among the most disastrous natural hazards and occurs in virtually all geographical areas. Severe drought events in recent decades, including 2010–2011 East Africa drought, 2011 Texas drought, 2012 U.S. Central Great Plains drought, and 2012–2015 California drought, have caused huge losses to agriculture, society, and ecosystems with profound impacts on crop production and water supply . Extensive impacts of drought in past decades at regional and global scales call for improved capability to cope with drought. Drought prediction plays a key role in drought early warning to mitigate its impacts. Drought is a complicated phenomenon and is among

the least understood natural hazards due to its multiple causing mechanisms or contributing factors operating at different temporal and spatial scales. Drought mostly originates from precipitation deficit, while in certain cases it may result from the anomaly of other variables, such as temperature or evapotranspiration. Specifically, high temperature may lead to increased evaporation and reduced soil moisture, causing drought in agricultural sectors. Moreover, drought may not be a purely natural hazard, for human activities such as land use changes and reservoir operation may alter hydrologic processes and affect drought development. Overall, the development and evolution of drought result from complicated interactions among meteorological anomalies, land surface processes, and human activities. Drought may occur with multiple processes driving its onset, persistence or recovery, happening at a wide range of time scales (sub seasonal/weekly, seasonal, multiyear, or decadal) and across different spatial scales (local, regional, continental, and global). Drought is commonly characterized at the seasonal time scale. Recently, it has been highlighted that drought may occur at the sub seasonal scale. For example, the 2012 central U.S. drought with rapid onset in May is generally referred to as flash drought (typically occurs for a few days or weeks), which results from concurrent soil moisture deficit and anomalously high temperatures (and increased evaporation).

Drought prediction generally refers to the prediction of drought severity (e.g., values of a specific drought indicator). In certain cases drought prediction also refers to other properties, such as drought duration and frequency, or phases, such as drought onset, persistence, and recovery. In this study, we mainly focus on the prediction of drought severity at the seasonal time scale, which centers around the current drought prediction efforts and is particularly related to the operational early warning to mitigate drought impacts.

1.3 OUTLINE OF THE PROJECT

The Proposed system works on identifying the sources by operating on a prediction system based on the Classification approach of Machine Learning in expecting the drought location of an area by processing various aspects that are indicators of a preceding drought such as the rainfall, moisture and groundwater data.

The version proposed here has been considered as a classification trouble. Classification is the problem of identifying which of a group of categories (sub-populations) a replacement commentary belongs, on the thought of a schooling set of facts containing observations (or instances) whose class club is recognized . The classification works at the prediction of the drought regions by producing one of the two consequences i.e. “Yes” and “No” to the matter statement - if the place is going to own a drought or not.

The affected areas are visualized on a world map highlighting the areas to be affected by the drought with the help of different indexed color patterns showing the various intensities of the probable drought in that region.

1.3.1 ADVANTAGES

- Serves as an important component of drought early warning systems, which can improve drought preparedness and increase resilience to drought impacts.
- Technological advances and new research are improving accuracy of forecasting systems.
- Prediction of the droughts can help in scaling down the losses of agriculture exponentially.
- The knowledge of an upcoming epidemic can help the people and the government to get ready from the start in all aspects i.e. emotionally, financially, etc.
- The effective prediction can also help in decreasing the death tolls due to droughts.

1.3.2 DISADVANTAGES

- Long range drought forecasts have a high degree of uncertainty.
- The effects of climate change result in increasingly unpredictable weather patterns, making forecasts less precise.
- Models may produce a wide range of probabilities, with a healthy level of uncertainty.
- Some systems, such as model simulations, require a high level of technical expertise to install and operate.

CHAPTER 2

LITERATURE SURVEY

The industries use high-end technologies to make their work effectively, there is a chance of using the same technology in a different situation for a different purpose. One such example is automated system hybrid statistical dynamical framework, which works on identifying the sources by operating on a prediction system based on the Classification approach of Machine Learning. A subtle outline of aggregated climatic forecasts for climatic forecasting with a request made region. The results of the statistical precipitation model are also generated for the same forecast periods and predict the estimated amount of water resource needed in case the region suffers a drought in the upcoming year.

TITLE: Advancing Drought Understanding, Monitoring, and Prediction

AUTHOR: Annarita Mariotti

YEAR: 2013

DESCRIPTION: The Project deals with an automatic capacity to monitor droughts in near-real time and providing accurate drought prediction from weeks to seasons in advance can greatly reduce the severity of social and economic damage caused by drought, a leading natural hazard for North America. The congressional mandate to establish the National Integrated Drought Information System (NIDIS; Public Law 109–430) in 2006 was a major impulse to develop, integrate, and provide drought information to meet the challenges posed by this hazard. Significant progress has been made on many fronts. On the research front, efforts by the broad scientific community have resulted in improved understanding of North American droughts and improved monitoring and forecasting tools. We now have a better understanding of the droughts of the twentieth century including the 1930s “Dust Bowl”; we have developed a broader array of tools and datasets that enhance the official North American Drought Monitor based on different methodologies such as state-of-the-art land surface modeling (e.g., the North American Land Data Assimilation System) and remote sensing (e.g., the evaporative stress index) to better characterize the occurrence and severity of drought in its multiple manifestations. In addition, we have new tools for drought prediction [including the new National Centers for Environmental Prediction (NCEP) Climate Forecast System, version 2, for operational prediction and an experimental National Multimodel Ensemble] and have explored diverse methodologies including ensemble hydrologic prediction approaches. Broad NIDIS-inspired progress is influencing the development of a Global Drought Information System (GDIS) under the auspices of the World Climate Research Program.

As part of its year 1 efforts, the task force has developed a drought testbed framework that individual research groups can use to test/evaluate methods and ideas. Central to this is a focus on three high-profile North American droughts, which are key areas

for NIDIS early warning system development (1998–2004 western U.S. drought, 2006/07 southeastern U.S. drought, and 2011/12 Texan–Mexican drought over the southern plains). The framework facilitates collaboration among projects, defines metrics to assess the quality of monitoring and prediction products, and helps to develop an experimental drought monitoring and prediction system that incorporates and assesses recent advances. Three working groups (WG) were formed to address the major aspects of the testbed: 1) WG-Metrics, to define and apply metrics to evaluate advances in drought monitoring and prediction; 2) WG-Case Studies, to analyze drought cases by integrating all aspects of drought research; and 3) WG-Experimental System, to incorporate research advances in an experimental drought monitoring and prediction system and assess improvements. To date, the Drought Task Force has proposed a Journal of Hydrometeorology special collection entitled “Advances in Drought Monitoring and Prediction” that will include research papers from individual task force members as well as a number of collective papers.

TITLE: Global integrated drought monitoring and prediction system

AUTHOR: Zengchao Hao

YEAR: 2014

DESCRIPTION: Drought is by far the most costly natural disaster that can lead to widespread impacts, including water and food crises. Here they present data sets available from the Global Integrated Drought Monitoring and Prediction System (GIDMaPS), which provides drought information based on multiple drought indicators. The system provides meteorological and agricultural drought information based on multiple satellite-, and model-based precipitation and soil moisture data sets. GIDMaPS includes a near real-time monitoring component and a seasonal probabilistic prediction module. The data sets include historical drought severity data from the monitoring component, and probabilistic seasonal forecasts from the prediction module. The probabilistic forecasts provide essential information for early warning, taking preventive measures, and planning mitigation strategies. GIDMaPS data sets are a significant extension to current capabilities and data sets for global drought assessment and early warning. The presented data sets would be instrumental in reducing drought impacts especially in developing countries. There results indicate that GIDMaPS data sets reliably captured several major droughts from across the globe.

GIDMaPS drought monitoring and prediction

The algorithm of the GIDMaPS is schematically illustrated below the figure. GIDMaPS integrates precipitation and soil moisture data from model simulations and remote sensing observations including the Modern-Era Retrospective analysis for Research and Applications (MERRA-Land), North American Land Data Assimilation System (NLDAS), Global Land Data Assimilation System (GLDAS) and the Global Drought Climate Data Record (GDCDR). GDCDR combines real-time Precipitation Estimation from Remotely Sensed Information using Artificial Neural Networks (PERSIANN) satellite data with long-term GPCP observations using a Bayesian algorithm.

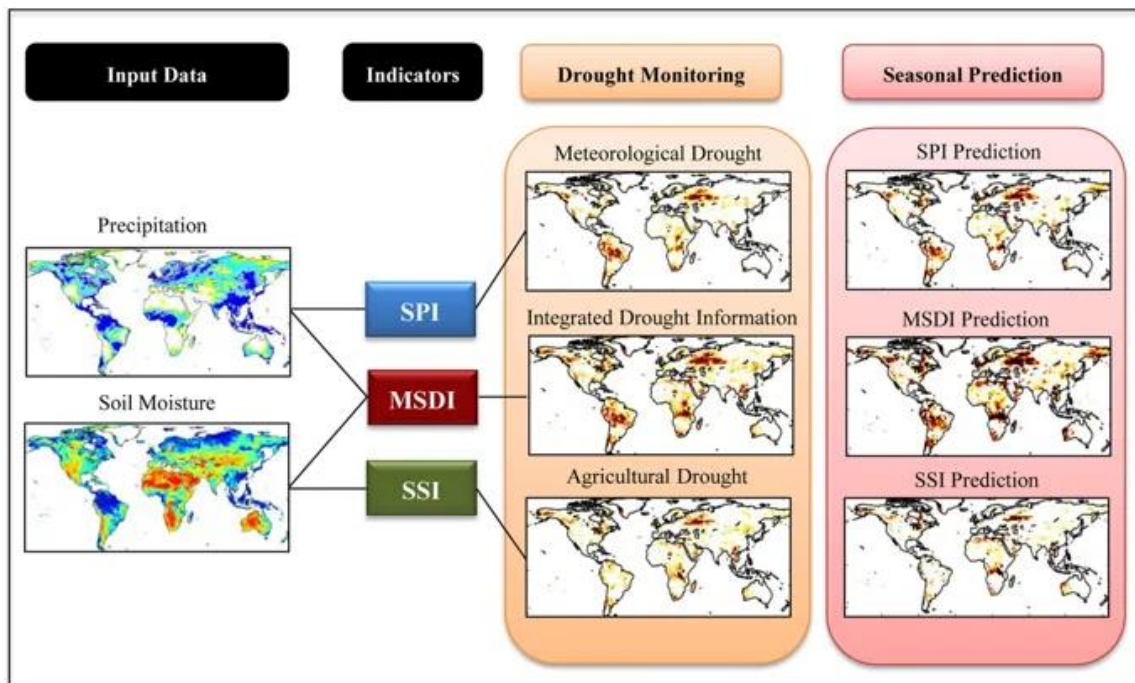


Fig 2.1 : Schematic view of GIDMaPS(Global Integrated Drought Monitoring and Prediction System)

TITLE: A hybrid statistical- dynamical framework for meteorological drought prediction: Application to the southwestern United States.

AUTHOR: Amir AghaKouchak

YEAR: 2014

DESCRIPTION: Amir AghaKouchak.[2] proposed hybrid statistical dynamical framework A subtle outline of aggregated climatic forecasts for climatic forecasting with a request made in the south western United States. The results of the statistical precipitation model are also generated for the same forecast periods. This step involves the conditional forecast of precipitation anomaly given the global teleconnection indices

This study proposes a hybrid precipitation prediction framework that combines the dynamical NMME forecasts with statistical simulations in each $1^\circ \times 1^\circ$ grid cell across the study area. The purpose of combining these two types of models is to select the best forecasts from the ensemble of dynamical and statistical predictions. They have used hybrid statistical- dynamical precipitation forecasting algorithms. The North America Multi- Model Ensemble (NMME) [Kirtman et al., 2014] is an ensemble of simulations from multiple dynamical models and provides probabilistic forecasts with some information regarding forecast uncertainty.

Expert Advice (EA) Algorithm: In the hybrid model, the Expert Advice (EA) algorithm [Cheng and AghaKouchak, 2015] is used to combine the dynamical and analog- year statistical components. In application, the final ensemble in the hybrid model consists of the two statistical analog- year model predictions, the grand ensemble mean of the

NMME forecasts, and the ensemble mean of each participating model in the NMME, that is, eight ensemble members from eight models—making a total of eleven inputs to the EA algorithm. The EA algorithm assigns different weights to the ensemble members based on their performance during a training period, and returns their weighted average as the best ensemble response.

TITLE: Prediction of Severe Drought Area Based on Random Forest: Using Satellite Image and Topography Data.

AUTHOR: Haikung park, Kyun Ping, and Dong Kung Lee.

YEAR: 2015

DESCRIPTION:

The ultimate objective of drought prediction is to prepare a mitigation plan in advance, rather than resolve intellectual curiosity about nature. Drought forecasting plays an important role in mitigating the negative effects of drought]; hence, various approaches for predicting droughts have constantly been attempted, such as stochastic methods, combined statistical and dynamical models, categorical prediction, machine learning approaches, and hybrid models. However, drought prediction is still challenging because, in addition to precipitation deficiency, complicated interactions among other variables, such as temperature, evapotranspiration, land surface processes, and human activities, also contribute to droughts . Further, meteorological anomalies, due to climate changes have made it increasingly difficult to predict precipitation, which forms the basis for forecasting agricultural drought-forecasting methods that are easy to use and scalable are required to realize practical drought mitigation plans. Moreover, these methods must consider the uncertainty of the precipitation forecasts and spatial information. To achieve this objective herein, we designed a model, called the severe drought area prediction (SDAP) model, to estimate soil moisture index (SMI) maps after several months using surface factors. All variables were composed of surface factors derived from remote sensing data, which are related to soil moisture, to obtain spatial information of agricultural drought.

The Agricultural droughts determined by soil moisture must be predicted several months ahead for proper and rapid resource allocation , because this allocation can mitigate the effects of upcoming droughts by supplying timely water and guaranteeing suitable crop growth and availability of food Water 2019,11, 705; doi:10.3390/w11040705 www.mdpi.com/journal/water Water 2019,11, 705 2 of 15 resources. Droughts are generally classified into four categories, namely, meteorological, hydrological, agricultural, and socio-economic droughts.

The model predicts the area of severe agricultural drought in terms of preparation information to help mitigate agricultural drought in case there exists a possibility of

meteorological drought. This model was proposed to help planning of priority and rapid water allocation for predominantly drought-affected risk area in occurring a drought. Therefore, prediction, in the SDAP model, does not imply a prediction of the occurrence of drought. Rather, it provides information on the future development and evolution of agricultural drought, assuming that rainfall-deficit conditions continue to prevail, in preparation for drought mitigation plans. We believe this is a realistic approach that avoids the uncertainty problem associated with meteorological drought forecasting. In order to design the SDAP model, we classified the land surface factors that affect the soil moisture into four categories, which are vegetation, topographic, thermal, and water factors during the non-rainfall period. Thermal increase is one of the core factors that increases the risk of agricultural drought by causing loss of soil moisture due to increased evaporation], while the vegetation factor delays the loss of soil moisture by slowing the increase in surface heat. The topography factor is another important determinant of soil moisture. The water content status is also a factor related to the soil moisture remaining after the drought period. These environmental factors, which are used as the initial land conditions as a predictor for drought forecasting, are regressed on the soil moisture after the non-rainfall period and can then predict soil moisture during short-term drought. In this regard, our model is designed to regress 15 input variables (derived from four surface factors) to the soil moisture index using the random forest (RF) algorithm.

CHAPTER 3

METHODOLOGY

The Proposed system works on identifying the sources by operating on a prediction system based on the Naive Bayesian Classifier expecting the drought location of a rustic with the aid of processing the rainfall information in that place.

The system proposed here has been considered as a classification problem. Classification is the problem of identifying which of a group of categories (sub-populations) a replacement commentary belongs, on the thought of a schooling set of facts containing observations (or instances) whose class club is recognised . The classification works at the prediction of the drought regions by producing one of the two consequences i.e. “Yes” and “No” to the matter statement - if the place is going to own a drought or not.

Naive Bayes methods are a set of supervised learning algorithms primarily based on making use of Bayes’ theorem with the “naive” assumption of conditional independence between each pair of functions given the value of the class variable. Bayes’ theorem states the subsequent relationship, given magnificence variable y and dependent feature vector x_1 through x_n ,:

Using naive conditional independence assumption that:

$$P(x_i|y, x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n) = P(x_i|y),$$

$$P(y | x_1, \dots, x_n) = \frac{P(y)P(x_1, \dots, x_n | y)}{P(x_1, \dots, x_n)}$$

for all i , this relation is simplified to the following formulae:

$$P(y | x_1, \dots, x_n) = \frac{P(y) \prod_{i=1}^n P(x_i | y)}{P(x_1, \dots, x_n)}$$

Since $P(x_1, \dots, x_n)$ is consistent given the input, we can use classification rule which is cited below:

$$P(y | x_1, \dots, x_n) \propto P(y) \prod_{i=1}^n P(x_i | y)$$

$$\Downarrow$$

$$\hat{y} = \arg \max_y P(y) \prod_{i=1}^n P(x_i | y),$$

and we are able to use Maximum A Posteriori (MAP) estimation to estimate $P(y)$ and $P(x_i | y)$; the previous is then the relative frequency of sophistication Y inside the training set. The special naive Bayes classifiers differ especially by way of the assumptions they make concerning the distribution of $P(x_i | y)$. In spite of their apparently over-simplified assumptions, naive Bayes classifiers have worked pretty nicely in lots of real-world situations, famously report class and spam filtering. They require a little quantity of coaching information to estimate the specified parameters.

Naive Bayes rookies and classifiers are often extraordinarily fast as compared to more state-of-the-art methods. The decoupling of the category conditional characteristic distributions means every distribution is often independently envisioned as a one-dimensional distribution. This, in turn, facilitates to alleviate problems stemming from the curse of dimensionality.

On the turn side, even though naive Bayes is called a decent classifier, it's far recognized to be an awful estimator, so the chance outputs from `predict_proba` are not to be taken too seriously.

The set of rules produces the areas wherein the possibilities of drought are there and then fetches the populace of that area. Spain's National Statistics Institute said that average family water consumption in Spain became 137 litres in keeping with character according to day in 2012. The populace is then used to offer an estimation using the formulae:

$$res = \sum (P) * est$$

where res = resources needed,

P = Population of that area,

est = estimated amount of water used per person in a day

3.1 ALGORITHM

3.1.1 CLASSIFICATION ALGORITHM

Classification is a technique to categorize our data into a desired and distinct number of classes where we can assign labels to each class.

Applications of Classification are: speech recognition, handwriting recognition, biometric identification, document classification etc.

Classifiers can be:

Binary classifiers: Classification with only 2 distinct classes or with 2 possible outcomes.

example: Male and Female

example: classification of spam email and non spam email

example: classification of author of book

example: positive and negative sentiment

example: classification of mood/feelings in songs/music

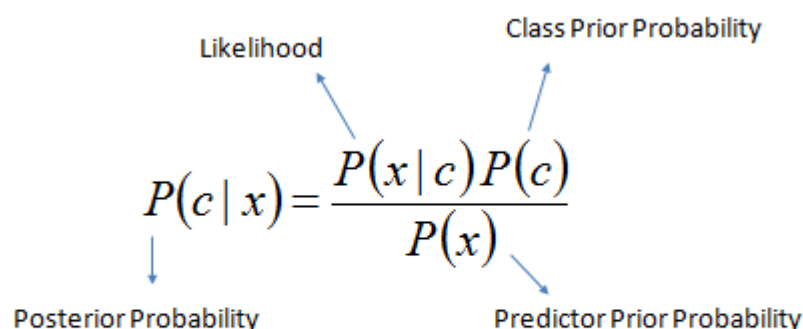
example: classification of types of crops

3.1.1.1 NAIVE BAYES CLASSIFIER

Naive Bayes is a simple technique for constructing classifiers: models that assign class labels to problem instances, represented as vectors of feature values, where the class labels are drawn from some finite set. There is not a single algorithm for training

such classifiers, but a family of algorithms based on a common principle: all naive Bayes classifiers assume that the value of a particular feature is independent of the value of any other feature, given the class variable. For example, a fruit may be considered to be an apple if it is red, round, and about 10 cm in diameter. A naive Bayes classifier considers each of these features to contribute independently to the probability that this fruit is an apple, regardless of any possible correlations between the color, roundness, and diameter features.

For some types of probability models, naive Bayes classifiers can be trained very efficiently in a supervised learning setting. In many practical applications, parameter estimation for naive Bayes models uses the method of maximum likelihood; in other words, one can work with the naive Bayes model without accepting Bayesian probability or using any Bayesian methods.

$$P(c | x) = \frac{P(x | c)P(c)}{P(x)}$$


$$P(c | X) = P(x_1 | c) \times P(x_2 | c) \times \dots \times P(x_n | c) \times P(c)$$

Despite their naive design and apparently oversimplified assumptions, naive Bayes classifiers have worked quite well in many complex real-world situations. In 2004, an analysis of the Bayesian classification problem showed that there are sound theoretical reasons for the apparently implausible efficacy of naive Bayes classifiers. Still, a comprehensive comparison with other classification algorithms in 2006 showed that Bayes classification is outperformed by other approaches, such as boosted trees or random forests.

Advantages:

- This algorithm requires a small amount of training data to estimate the necessary parameters.
- Naive Bayes classifiers are extremely fast compared to more sophisticated methods.

Disadvantages:

- Naive Bayes is known to be a bad estimator.

3.1.1.2 *DECISION TREE CLASSIFIER*

Decision tree learning is a method commonly used in data mining.[1] The goal is to create a model that predicts the value of a target variable based on several input variables.

A decision tree is a simple representation for classifying examples. For this section, assume that all of the input features have finite discrete domains, and there is a single target feature called the "classification". Each element of the domain of the classification is called a class. A decision tree or a classification tree is a tree in which each internal (non-leaf) node is labeled with an input feature. The arcs coming from a node labeled with an input feature are labeled with each of the possible values of the target or output feature or the arc leads to a subordinate decision node on a different input feature. Each leaf of the tree is labeled with a class or a probability distribution over the classes, signifying that the data set has been classified by the tree into either a specific class, or into a particular probability distribution (which, if the decision tree is well-constructed, is skewed towards certain subsets of classes).

A tree is built by splitting the source set, constituting the root node of the tree, into subsets - which constitute the successor children. The splitting is based on a set of splitting rules based on classification features. This process is repeated on each derived subset in a recursive manner called recursive partitioning. The recursion is completed when the subset at a node has all the same values of the target variable, or when splitting no longer adds value to the predictions. This process of top-down

induction of decision trees (TDIDT) is an example of a greedy algorithm, and it is by far the most common strategy for learning decision trees from data[citation needed].

In data mining, decision trees can be described also as the combination of mathematical and computational techniques to aid the description, categorization and generalization of a given set of data.

Data comes in records of the form:

$$(\mathbf{x}, Y) = (x_1, x_2, x_3, \dots, x_k, Y)$$

The dependent variable, Y , is the target variable that we are trying to understand, classify or generalize. The vector \mathbf{x} is composed of the features, x_1, x_2, x_3 , etc., that are used for that task.

Advantages:

- Simple to understand and interpret.
- Able to handle both numerical and categorical data.
- Requires little data preparation.
- Possible to validate a model using statistical tests.
- Performs well with large datasets.
- Mirrors human decision making more closely than other approaches.

Disadvantages:

- Trees can be very non-robust. A small change in the training data can result in a large change in the tree and consequently the final predictions.
- Decision-tree learners can create over-complex trees that do not generalize well from the training data. (This is known as overfitting.) Mechanisms such as pruning are necessary to avoid this problem (with the exception of some algorithms such as the Conditional Inference approach, that does not require pruning).
- For data including categorical variables with different numbers of levels, information gain in decision trees is biased in favor of attributes with more levels. However, the issue of biased predictor selection is avoided by the

Conditional Inference approach, a two-stage approach, or adaptive leave-one-out feature selection.

3.1.1.3 *RANDOM FOREST CLASSIFIER*

Decision trees are a popular method for various machine learning tasks. Tree learning "come closest to meeting the requirements for serving as an off-the-shelf procedure for data mining", say Hastie, "because it is invariant under scaling and various other transformations of feature values, is robust to inclusion of irrelevant features, and produces inspectable models. However, they are seldom accurate".

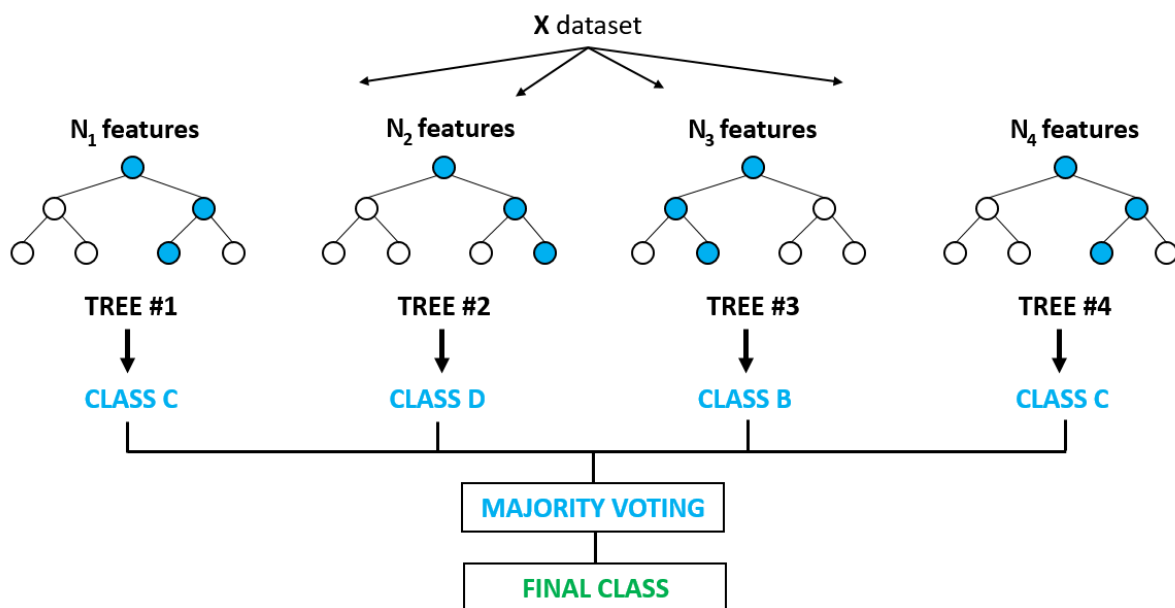


Fig.3.1. Random forest classifier

In particular, trees that are grown very deep tend to learn highly irregular patterns: they overfit their training sets, i.e. have low bias, but very high variance. Random forests are a way of averaging multiple deep decision trees, trained on different parts of the same training set, with the goal of reducing the variance. This comes at the expense of a small increase in the bias and some loss of interpretability, but generally greatly boosts the performance in the final model.

As part of their construction, random forest predictors naturally lead to a dissimilarity measure among the observations. One can also define a random forest dissimilarity

measure between unlabeled data: the idea is to construct a random forest predictor that distinguishes the “observed” data from suitably generated synthetic data. The observed data are the original unlabeled data and the synthetic data are drawn from a reference distribution. A random forest dissimilarity can be attractive because it handles mixed variable types very well, is invariant to monotonic transformations of the input variables, and is robust to outlying observations. The random forest dissimilarity easily deals with a large number of semi-continuous variables due to its intrinsic variable selection; for example, the "Addcl 1" random forest dissimilarity weighs the contribution of each variable according to how dependent it is on other variables. The random forest dissimilarity has been used in a variety of applications, e.g. to find clusters of patients based on tissue marker data.

Advantages:

- The predictive performance can compete with the best supervised learning algorithms
- They provide a reliable feature importance estimate
- They offer efficient estimates of the test error without incurring the cost of repeated model training associated with cross-validation

Disadvantages:

- An ensemble model is inherently less interpretable than an individual decision tree
- Training a large number of deep trees can have high computational costs (but can be parallelized) and use a lot of memory
- Predictions are slower, which may create challenges for applications.

3.2 USER INTERFACE

The User Interface is made up of various components of which the most important components are flask, html, css, javascript and various API's.

3.2.1 FLASK

Flask is a micro web framework written in Python. 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.

3.2.2 HTML5

HTML5 is the latest evolution of the standard that defines HTML. The term represents two different concepts. It is a new version of the language HTML, with new elements, attributes, and behaviors, and a larger set of technologies that allows the building of more diverse and powerful Web sites and applications. This set is sometimes called HTML5 & friends and often shortened to just HTML5.

Designed to be usable by all Open Web developers, this reference page links to numerous resources about HTML5 technologies, classified into several groups based on their function.

- Semantics: allowing you to describe more precisely what your content is.
- Connectivity: allowing you to communicate with the server in new and innovative ways.
- Offline and storage: allowing webpages to store data on the client-side locally and operate offline more efficiently.
- Multimedia: making video and audio first-class citizens in the Open Web.
- 2D/3D graphics and effects: allowing a much more diverse range of presentation options.
- Performance and integration: providing greater speed optimization and better usage of computer hardware.
- Device access: allowing for the usage of various input and output devices.
- Styling: letting authors write more sophisticated themes.

3.2.3 CSS

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language like HTML. CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript.

CSS is designed to enable the separation of presentation and content, including layout, colors, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple web pages to share formatting by specifying the relevant CSS in a separate .css file, and reduce complexity and repetition in the structural content.

Separation of formatting and content also makes it possible to present the same markup page in different styles for different rendering methods, such as on-screen, in print, by voice (via speech-based browser or screen reader), and on Braille-based tactile devices. CSS also has rules for alternate formatting if the content is accessed on a mobile device.

The name cascading comes from the specified priority scheme to determine which style rule applies if more than one rule matches a particular element. This cascading priority scheme is predictable.

The CSS specifications are maintained by the World Wide Web Consortium (W3C). Internet media type (MIME type) text/css is registered for use with CSS by RFC 2318 (March 1998). The W3C operates a free CSS validation service for CSS documents.

In addition to HTML, other markup languages support the use of CSS including XHTML, plain XML, SVG, and XUL.

3.3.4 JAVASCRIPT

JavaScript often abbreviated as JS, is a programming language that conforms to the ECMAScript specification. JavaScript is high-level, often just-in-time compiled, and multi-paradigm. It has curly-bracket syntax, dynamic typing, prototype-based object-orientation, and first-class functions.

Alongside HTML and CSS, JavaScript is one of the core technologies of the World Wide Web. JavaScript enables interactive web pages and is an essential part of web applications. The vast majority of websites use it for client-side page behavior, and all major web browsers have a dedicated JavaScript engine to execute it.

As a multi-paradigm language, JavaScript supports event-driven, functional, and imperative programming styles. It has application programming interfaces (APIs) for working with text, dates, regular expressions, standard data structures, and the Document Object Model (DOM). However, the language itself does not include any input/output (I/O), such as networking, storage, or graphics facilities, as the host environment (usually a web browser) provides those APIs.

Originally used only in web browsers, JavaScript engines are also now embedded in server-side website deployments and non-browser applications. Although there are similarities between JavaScript and Java, including language name, syntax, and respective standard libraries, the two languages are distinct and differ greatly in design.

3.3.5 MAPBOX API

The Mapbox web services APIs allow you to programmatically access Mapbox tools and services. You can use these APIs to retrieve information about your account, upload and change resources, use core Mapbox tools, and more.

Mapbox APIs are divided into four distinct services: Maps, Navigation, Search, and Accounts. Each of these services has its own overview page in this documentation. These overview pages are divided into the individual APIs that make up the service. The documentation for each API is structured by endpoints. An endpoint is a specific method within an API that performs one action and is located at a specific URL.

3.3.6 JQUERY

jQuery is a JavaScript library designed to simplify HTML DOM tree traversal and manipulation, as well as event handling, CSS animation, and Ajax. It is free, open-source software using the permissive MIT License. As of May 2019, jQuery is used by 73% of the 10 million most popular websites. Web analysis indicates that it is the most

widely deployed JavaScript library by a large margin, having 3 to 4 times more usage than any other JavaScript library.

jQuery's syntax is designed to make it easier to navigate a document, select DOM elements, create animations, handle events, and develop Ajax applications. jQuery also provides capabilities for developers to create plug-ins on top of the JavaScript library. This enables developers to create abstractions for low-level interaction and animation, advanced effects and high-level, themeable widgets. The modular approach to the jQuery library allows the creation of powerful dynamic web pages and Web applications.

The set of jQuery core features—DOM element selections, traversal and manipulation—enabled by its selector engine (named "Sizzle" from v1.3), created a new "programming style", fusing algorithms and DOM data structures. This style influenced the architecture of other JavaScript frameworks like YUI v3 and Dojo, later stimulating the creation of the standard Selectors API. Later, this style has been enhanced with a deeper algorithm-data fusion in an heir of jQuery, the D3.js framework.

Microsoft and Nokia bundle jQuery on their platforms. Microsoft includes it with Visual Studio for use within Microsoft's ASP.NET AJAX and ASP.NET MVC frameworks while Nokia has integrated it into the Web Run-Time widget development platform.

3.3.7 PYTHON

Python is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python's design philosophy emphasizes code readability with its notable use of significant whitespace. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects.

Python is dynamically typed and garbage-collected. It supports multiple programming paradigms, including procedural, object-oriented, and functional programming. Python is often described as a "batteries included" language due to its comprehensive standard library.

Python was conceived in the late 1980s as a successor to the ABC language. Python 2.0, released in 2000, introduced features like list comprehensions and a garbage collection system capable of collecting reference cycles. Python 3.0, released in 2008, was a major revision of the language that is not completely backward-compatible, and much Python 2 code does not run unmodified on Python 3.

The different python libraries used in the development of the project are:

3.3.7.1 PYTHON SKLEARN LIBRARY

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python.

It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use.

The library is built upon the SciPy (Scientific Python) that must be installed before you can use scikit-learn. This stack that includes:

- NumPy: Base n-dimensional array package
- SciPy: Fundamental library for scientific computing
- Matplotlib: Comprehensive 2D/3D plotting
- IPython: Enhanced interactive console
- Sympy: Symbolic mathematics
- Pandas: Data structures and analysis

Extensions or modules for SciPy are conventionally named SciKits. As such, the module provides learning algorithms and is named scikit-learn.

The vision for the library is a level of robustness and support required for use in production systems. This means a deep focus on concerns such as easy of use, code quality, collaboration, documentation and performance.

Although the interface is Python, c-libraries are leverage for performance such as numpy for arrays and matrix operations, LAPACK, LibSVM and the careful use of cython.

3.3.7.1.1 PYTHON - NUMPY

NumPy is the fundamental package for scientific computing with Python. It contains among other things:

- a powerful N-dimensional array object
- sophisticated (broadcasting) functions
- tools for integrating C/C++ and Fortran code
- useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined. This allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

3.3.7.1.1 *PYTHON - MATPLOTLIB*

matplotlib.pyplot is a plotting library used for 2D graphics in python programming language. It can be used in python scripts, shell, web application servers and other graphical user interface toolkits.

There are several toolkits which are available that extend python matplotlib functionality. Some of them are separate downloads, others can be shipped with the matplotlib source code but have external dependencies.

- Basemap: It is a map plotting toolkit with various map projections, coastlines and political boundaries.
- Cartopy: It is a mapping library featuring object-oriented map projection definitions, and arbitrary point, line, polygon and image transformation capabilities.
- Excel tools: Matplotlib provides utilities for exchanging data with Microsoft Excel.
- Mplot3d: It is used for 3-D plots.
- Natgrid: It is an interface to the natgrid library for irregular gridding of the spaced data.

There are various plots which can be created using python matplotlib. The matplotlib makes the representation and visualization very easy to use and to understand. Some

of the plot representations which can be obtained using the matplotlib library of python are listed below:



Fig. 3.2. Plots of Matplotlib in Python

3.3.7.1.1 PYTHON - PANDAS

Pandas has so many uses that it might make sense to list the things it can't do instead of what it can do. This tool is essentially your data's home. Through pandas, you get acquainted with your data by cleaning, transforming, and analyzing it.

For example, say you want to explore a dataset stored in a CSV on your computer. Pandas will extract the data from that CSV into a DataFrame — a table, basically — then let you do things like:

- Calculate statistics and answer questions about the data, like
- What's the average, median, max, or min of each column?
- Does column A correlate with column B?
- What does the distribution of data in column C look like?
- Clean the data by doing things like removing missing values and filtering rows or columns by some criteria
- Visualize the data with help from Matplotlib. Plot bars, lines, histograms, bubbles, and more.
- Store the cleaned, transformed data back into a CSV, other file or database

3.4 FLOWCHART

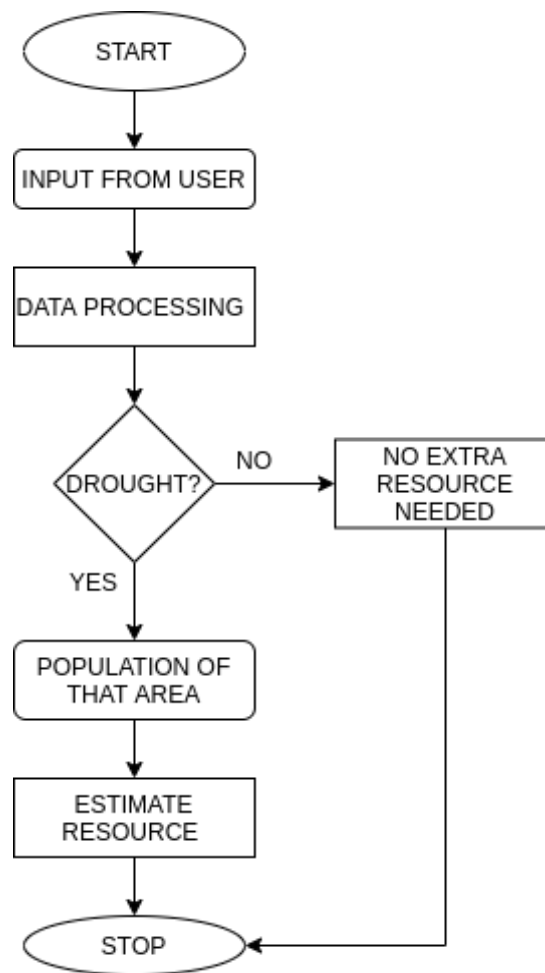


Fig. 3.3. Flowchart of water crisis prediction system

The above flowchart illustrates the model of the water crisis management system which works towards the prediction of droughts throughout the world by carefully processing the rainwater data along with several other factors. The system initiates with the users input of the location which is to be processed for the drought prediction. The location data is fetched and data processing begins for the selected region in order to predict the drought by mapping all the available constraints like rainwater, groundwater, humidity, moisture, etc. The classification model consisting of any of the classification algorithms efficiently finds out if the drought is expected in that region or not. If “yes” is the answer obtained by the classification model then the area is marked as a drought prone area on the map.

CHAPTER 4

RESULT AND DISCUSSIONS

4.1. DISCUSSION

4.1.1. *BUILDING THE PROJECT*

4.1.1.1 *CSV DATASET*

The project started with the search of appropriate data to be processed from various resources. The data that was needed for the proper implementation of the prediction system was supposed to have some constraints.

The required constraints in the dataset was:

1. Rainwater data
2. Pressure data
3. Humidity

And so on.

The dataset on which the first trial was performed was a very unsaturated and dirty data(had some unwanted values, many missing values and many mistakes). The data was put into preprocessing but the results of the preprocessed data were not satisfactory.

The dataset was again searched and this time the search ended on the dataset of South America which contained all the required fields needed for the planned prediction model i.e. rainfall, pressure, moisture and some other constraints.

4.1.1.2 *PREDICTION MODEL*

The prediction model was proposed as a Classification Problem as the problem statement was “To check if the selected region is going to have a drought or not”. Thus the problem was dealt like a simple “Yes” or “No” problem. The algorithms chosen to deal with the classification problem were:

- I. GaussianNB Classifier

- II. Decision Tree Classifier
- III. Random Forest Classifier

With the preprocessed data, the model was built for the individual algorithms and a confusion matrix was drawn for each of them showing their accuracy, recall and precision. These were the measures used to judge the performance of the designed classification model.

4.1.1.3 *DEPLOYMENT - HEROKU CLOUD*

The issue of unavailability of seamless and easy implementation of the project was faced at the early stage of the project. The collaboration as well as the cross platform working of the project was one of the most pressing issues that was being faced while developing and even after developing the project.

The simple solution we discovered was to use the pipelining concept provided by the Heroku Cloud service as a part of its Devops operation i.e. we can deploy the project by connecting a present repository (say Github in our case) to the cloud and make the necessary changes with each modulation in the source code of the master.

The advantages we got on cloud are:

1. Auto Deployment of the project.
2. Ease in collaboration
3. Easy Versioning, etc.

4.2. **PERFORMANCE ANALYSIS**

The proposed algorithm is evaluated based on different parameters such as true positive (TP), false positive (FP), True Negative (TN) and false negative (FN). TP is described as when a system predicts a drought when the drought is observed in that year. FP is described as when a system predicts a drought but no drought is observed in that particular area in that year. TN is when no drought is predicted and no drought is observed in that region in the year of prediction. FN is when no drought is predicted but the area seems to have the drought.

The measure of prediction is defined as the combination of two criteria are defined as follows

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}}$$

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}}$$

	Unnamed: 0	Year	Temperature	Pressure	Rainfall	Drought	Average Rainfall	Average Pressure	Average Temperature	Rainfall_T	Pressure_T	Temperature_T
Unnamed: 0	1.0	0.12	0.16	-0.12	-0.064	0.12	-0.081	-0.2	0.2	-0.0076	-0.012	-0.03
Year	0.12	1.0	0.047	0.08	-0.0097	-0.14	0.0	0.0	0.0	0.014	-0.11	-0.071
Temperature	0.16	0.047	1.0	-0.17	-0.0056	0.011	0.067	-0.32	0.76	0.079	-0.038	-0.6
Pressure	-0.12	0.08	-0.17	1.0	-0.15	-0.015	0.0089	0.67	-0.24	0.26	-0.72	-0.02
Rainfall	-0.064	-0.0097	-0.0056	-0.15	1.0	0.002	0.69	-0.02	0.073	-0.62	0.21	0.078
Drought	0.12	-0.14	0.011	-0.015	0.002	1.0	0.053	-0.024	0.015	0.074	0.0073	0.0013
Average Rainfall	-0.081	0.0	0.067	0.0089	0.69	0.053	1.0	-0.017	0.12	0.049	0.0022	0.0022
Average Pressure	-0.2	0.0	-0.32	0.67	-0.02	-0.024	-0.017	1.0	-0.4	0.01	-0.0093	0.0096
Average Temperature	0.2	0.0	0.76	-0.24	0.073	0.015	0.12	-0.4	1.0	0.0077	-0.006	-0.0015
Rainfall_T	-0.0076	0.014	0.079	0.26	-0.62	0.074	0.049	0.01	0.0077	1.0	-0.37	-0.13
Pressure_T	-0.012	-0.11	-0.038	-0.72	0.21	0.0073	0.0022	-0.0093	-0.006	-0.37	1.0	0.048
Temperature_T	-0.03	-0.071	-0.6	-0.02	0.078	0.0013	0.0022	0.0096	-0.0015	-0.13	0.048	1.0

Fig.4.1. Processed data for the classification model

The data can be analyzed on the basis of the confusion matrix obtained by using different classification algorithms and then the most efficient among them is chosen to be the prediction model for the project. On implementation of the confusion matrix we get three values on which the model is decided i.e. accuracy, precision and recall.

Confusion Matrix, Accuracy=0.666667, Precision=0.494737, Recall=0.494737

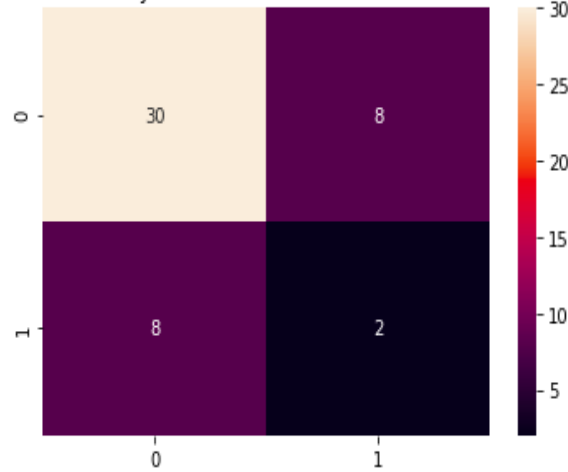


Fig.4.2. Confusion Matrix for Decision Tree

Confusion Matrix, Accuracy=0.791667, Precision=0.395833, Recall=0.500000

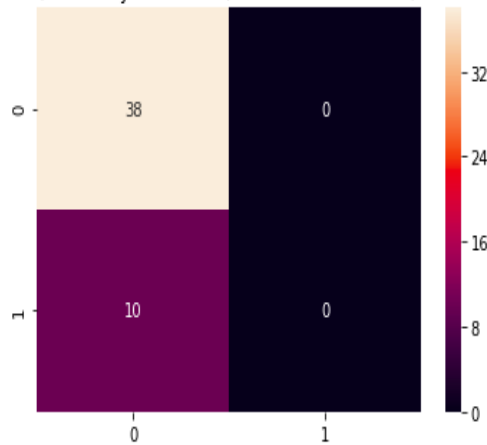


Fig. 4.3. Confusion Matrix for GaussianNB

Confusion Matrix, Accuracy=0.791667, Precision=0.395833, Recall=0.500000

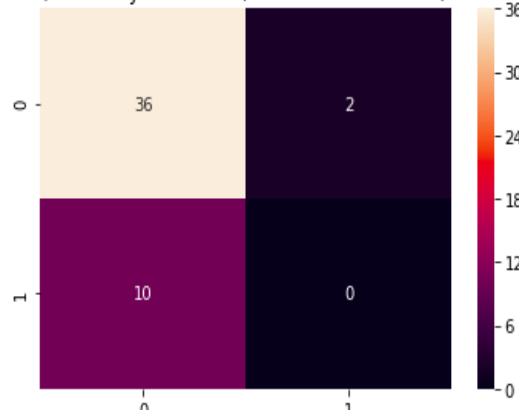


Fig. 4.4. Confusion Matrix for Random Forest Classifier

CHAPTER 5

CONCLUSION AND FUTURE WORK

5.1. CONCLUSION

The main feature of this project is the ability to predict the estimated amount of water resource needed in case the region suffers a drought in the upcoming year.

The ability of the prediction model to predict the drought in a region based on the rainfall data based on the random forest classification model with a 0.791667 % accuracy makes the model highly reliable and can be implemented with a proper set of data for better results.

5.2 FUTURE WORK

The present version of the prediction model can be improved by using proper and highly reliable data and also implementing the other constraints into consideration like the amount of water resources supplied each year in the form of water tankers and supplied water, current and reliable ground data on regional basis which is updated on a regular basis.

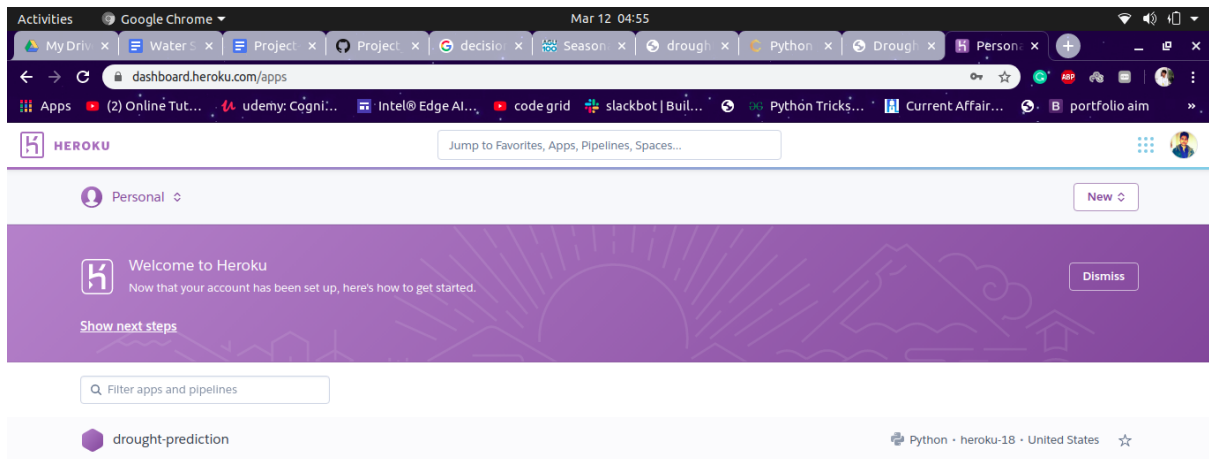
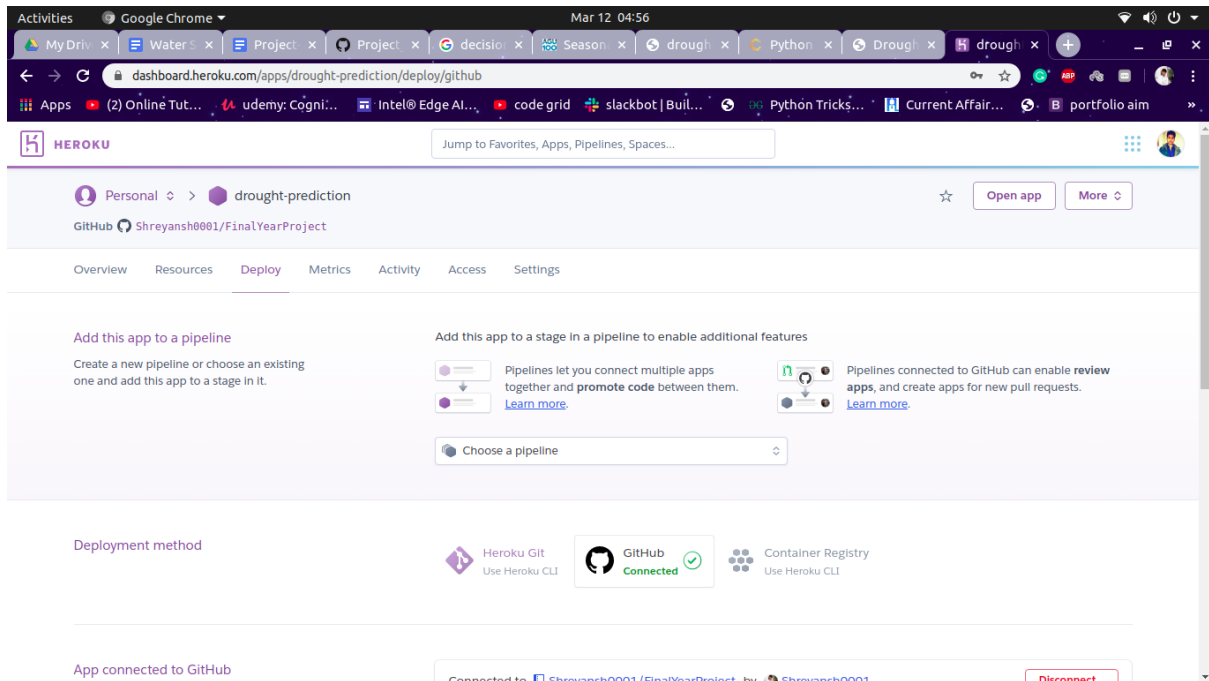
The accuracy can be increased exponentially if all such constraints are considered in the proposed model.

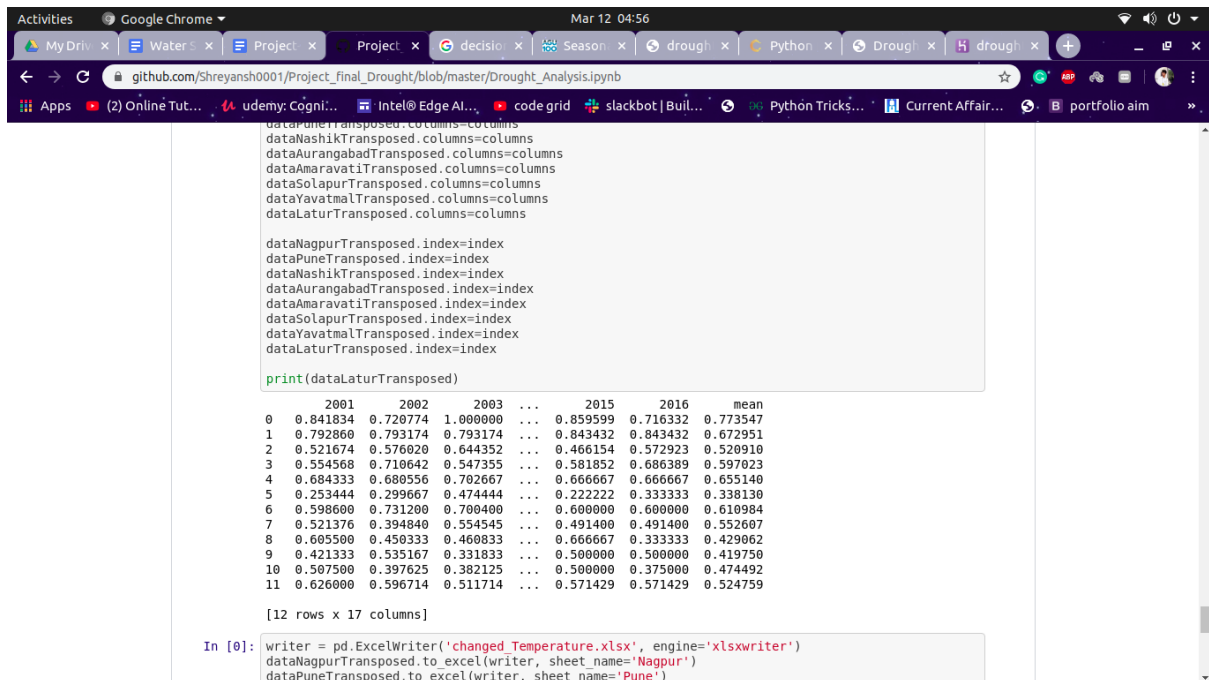
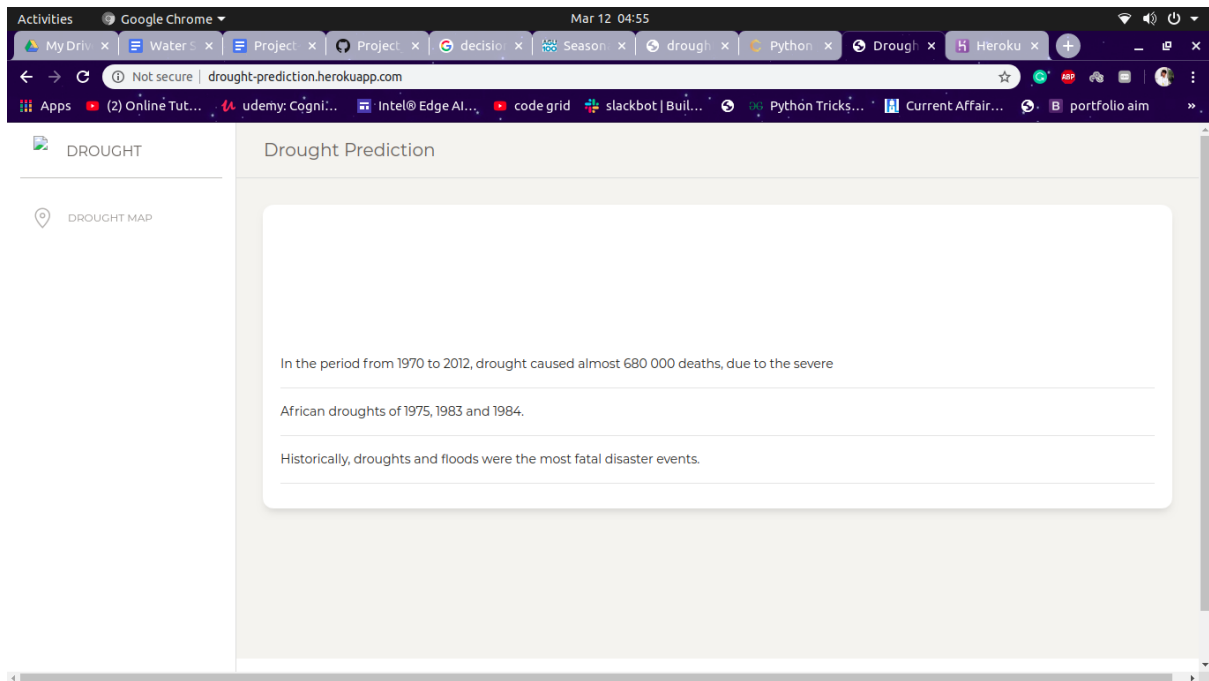
REFERENCES:

- 1) Aas, K., Czado, C., Frigessi, A., & Bakken, H. (2009). Pair-copula constructions of multiple dependence. *Insurance: Mathematics and Economics*, 44, 182–198.
- 2) AghaKouchak, A., Farahmand, A., Melton, F., Teixeira, J., Anderson, M., Wardlaw, B., & Hain, C. (2015). Remote sensing of drought: Progress, challenges and opportunities. *Reviews of Geophysics*, 53, 452–480. <https://doi.org/10.1002/2014RG000456>
- 3) Agresti, A. (2007). *An Introduction to Categorical Data Analysis*. Hoboken, NJ: John Wiley & Sons. <https://doi.org/10.1002/0470114754>
- 4) Arnal, L., Wood, A. W., Stephens, E., Cloke, H. L., & Pappenberger, F. (2017). An efficient approach for estimating streamflow forecast skill elasticity. *Journal of Hydrometeorology*, 18(6), 1715–1729. <https://doi.org/10.1175/JHM-D-16-0259.1>
- 5) Asoka, A., & Mishra, V. (2015). Prediction of vegetation anomalies to improve food security and water management in India. *Geophysical Research Letters*, 42, 5290–5298. <https://doi.org/10.1002/2015GL063991>
- 6) Avilés, A., Célleri, R., Paredes, J., & Solera, A. (2015). Evaluation of Markov Chain based drought forecasts in an Andean regulated river basin using the skill scores RPS and GMSS. *Water Resources Management*, 29(6), 1949–1963. <https://doi.org/10.1007/s11269-015-0921-2>
- 7) Bacanlı, U. G., Firat, M., & Dikbas, F. (2009). Adaptive neuro-fuzzy inference system for drought forecasting. *Stochastic Environmental Research and Risk Assessment*, 23, 1143–1154. <https://doi.org/10.1007/s00477-008-0288-5>
- 8) Bachmair, S., Stahl, K., Collins, K., Hannaford, J., Acreman, M., Svoboda, M., ... Fuchs, B. (2016). Drought indicators revisited: The need for a wider consideration of environment and society. *Wiley Interdisciplinary Reviews: Water*, 3(4), 516–536. <https://doi.org/10.1002/wat2.1154>
- 9) Bachmair, S., Svensson, C., Hannaford, J., Barker, L., & Stahl, K. (2016). A quantitative analysis to objectively appraise drought indicators and model drought impacts. *Hydrology and Earth System Sciences*, 20(7), 2589–2609. <https://doi.org/10.5194/hess-20-2589-2016>

- 10)Barros, A. P., & Bowden, G. J. (2008). Toward long-lead operational forecasts of drought: An experimental study in the Murray-Darling River basin. *Journal of Hydrology*, 357, 349–367. <https://doi.org/10.1016/j.jhydrol.2008.05.026>
- 11)Barua, S., Ng, A. W. M., & Perera, B. J. C. (2012). Artificial neural network-based drought forecasting using a nonlinear aggregated drought index. *Journal of Hydrologic Engineering*, 17, 1408–1413. [https://doi.org/10.1061/\(ASCE\)HE.1943-5584.0000574](https://doi.org/10.1061/(ASCE)HE.1943-5584.0000574)

SCREENSHOTS





```

for j in range(0,12):
    data.loc[count]=[i,months[j],"Yavatmal",dataYavatmalTemperature.loc[j,i],dataYavatmalPressure.loc[j,i],dataYavatmalRainfall.loc[j,i],dataYavatmalDrought.loc[j,i],dataYavatmalRainfall.loc[j,'mean'],dataYavatmalPressure.loc[j,'mean'],dataYavatmalTemperature.loc[j,'mean']]
    count=count+1

for i in range(2001,2017):
    for j in range(0,12):
        data.loc[count]=[i,months[j],"Latur",dataLaturTemperature.loc[j,i],dataLaturPressure.loc[j,i],dataLaturRainfall.loc[j,i],dataLaturDrought.loc[j,i],dataLaturRainfall.loc[j,'mean'],dataLaturPressure.loc[j,'mean'],dataLaturTemperature.loc[j,'mean']]
        count=count+1

print(data)

writer = pd.ExcelWriter('Integrated_Data_final.xlsx', engine='xlsxwriter')
data.to_excel(writer, sheet_name='Data')
writer.save()

Year      Month  ... Average Pressure  Average Temperature
0      2001  January  ...           0.787500           0.464230
1      2001  February  ...           0.812500           0.462685
2      2001  March     ...           0.705357           0.489369
3      2001  April     ...           0.575000           0.706559
4      2001  May       ...           0.364583           0.825874
...      ...      ...
1531   2016  August   ...           0.537500           0.552607
1532   2016  September ...           0.518417           0.429062
1533   2016  October  ...           0.382083           0.419750
1534   2016  November ...           0.375000           0.474492
1535   2016  December ...           0.339286           0.524759

[1536 rows x 10 columns]

In [0]: df2 = pd.read_excel("Integrated_Data_final.xlsx")
df2["Rainfall_T"] = df2["Average Rainfall"] - df2["Rainfall"]
df2["Pressure_T"] = df2["Average Pressure"] - df2["Pressure"]
df2["Temperature_T"] = df2["Average Temperature"] - df2["Temperature"]

```

```

In [0]: df2.corr(method='spearman').style.format("{:.2}").background_gradient(cmap=plt.get_cmap('coolwarm'), axis=1)

Out[0]:

```

	Unnamed: 0	Year	Temperature	Pressure	Rainfall	Drought	Average Rainfall	Average Pressure	Average Temperature	Rainfall
Unnamed: 0	1.0	0.12	0.16	-0.12	-0.064	0.12	-0.081	-0.2	0.2	-0.0076
Year	0.12	1.0	0.047	0.08	-0.0097	-0.14	0.0	0.0	0.0	0.014
Temperature	0.16	0.047	1.0	-0.17	-0.0056	0.011	0.067	-0.32	0.76	0.079
Pressure	-0.12	0.08	-0.17	1.0	-0.15	-0.015	0.0089	0.67	-0.24	0.26
Rainfall	-0.064	-0.0097	-0.0056	-0.15	1.0	0.002	0.69	-0.02	0.073	-0.62
Drought	0.12	-0.14	0.011	-0.015	0.002	1.0	0.053	-0.024	0.015	0.074
Average Rainfall	-0.081	0.0	0.067	0.0089	0.69	0.053	1.0	-0.017	0.12	0.049
Average Pressure	-0.2	0.0	-0.32	0.67	-0.02	-0.024	-0.017	1.0	-0.4	0.01
Average Temperature	0.2	0.0	0.76	-0.24	0.073	0.015	0.12	-0.4	1.0	0.0077
Rainfall_T	-0.0076	0.014	0.079	0.26	-0.62	0.074	0.049	0.01	0.0077	1.0
Pressure_T	-0.012	-0.11	-0.038	-0.72	0.21	0.0073	0.0022	-0.0093	-0.006	-0.37
Temperature_T	-0.03	-0.071	-0.6	-0.02	0.078	0.0013	0.0022	0.0096	-0.0015	-0.13

```

In [0]: df3 = df2.loc[df2['City'] == 'Nagpur']
df3 = df3[["Rainfall_T", "Pressure_T", "Temperature_T", "Drought"]]
df3.head(10)

```

Activities Google Chrome Mar 12 04:56

My Drive Water Project Project decisio Season drought Python Drought drought

github.com/Shreyansh0001/Project_final_Drought/blob/master/Drought_Analysis.ipynb

Apps (2) Online Tut... udemy: Cogni... Intel® Edge AI... code grid slackbot | Buil... Python Tricks... Current Affair... portfolio aim

```

1532 2016 September ... 0.510417 0.429862
1533 2016 October ... 0.382083 0.419750
1534 2016 November ... 0.375000 0.474492
1535 2016 December ... 0.339286 0.524759

[1536 rows x 10 columns]

In [0]: df2 = pd.read_excel("Integrated Data final.xlsx")
df2["Rainfall_T"] = df2["Average Rainfall"] - df2["Rainfall"]
df2["Pressure_T"] = df2["Average Pressure"] - df2["Pressure"]
df2["Temperature_T"] = df2["Average Temperature"] - df2["Temperature"]
df2.head(10)

Out[0]:

```

Unnamed: 0	Year	Month	City	Temperature	Pressure	Rainfall	Drought	Average Rainfall	Average Pressure	Average Temperature	Ra
0	2001	January	Nagpur	0.170774	0.800000	0.113438	0	0.137599	0.787500	0.464230	0.0
1	2001	February	Nagpur	0.391420	1.000000	0.001995	0	0.182323	0.812500	0.462685	0.1
2	2001	March	Nagpur	0.325112	0.714286	0.176877	0	0.174763	0.705357	0.489369	-0.1
3	2001	April	Nagpur	0.528225	0.400000	0.394282	0	0.165459	0.575000	0.706559	-0.1
4	2001	May	Nagpur	0.699889	0.166667	0.356878	0	0.227460	0.364583	0.825874	-0.1
5	2001	June	Nagpur	0.913111	0.000000	0.491575	0	0.327609	0.343750	0.670696	-0.1
6	2001	July	Nagpur	0.826600	0.500000	0.427118	0	0.666176	0.385417	0.819139	0.2
7	2001	August	Nagpur	0.960197	0.200000	0.436723	0	0.407994	0.375000	0.835809	-0.1
8	2001	September	Nagpur	0.655000	0.166667	0.144665	0	0.327997	0.343750	0.652260	0.1
9	2001	October	Nagpur	0.531667	0.500000	0.586957	0	0.213768	0.781250	0.500802	-0.1

```

In [0]: df2.corr(method='spearman').style.format("{:.2}").background_gradient(cmap=plt.get_cmap('coolwarm'), axis=1)

```

Activities Google Chrome Mar 12 04:57

My Drive Water Project Project decisio Season drought Python Drought drought

github.com/Shreyansh0001/Project_final_Drought/blob/master/Drought_Analysis.ipynb

Apps (2) Online Tut... udemy: Cogni... Intel® Edge AI... code grid slackbot | Buil... Python Tricks... Current Affair... portfolio aim

```

In [0]: df3.corr(method='spearman').style.format("{:.2}").background_gradient(cmap=plt.get_cmap('coolwarm'), axis=1)

Out[0]:

```

	Rainfall_T	Pressure_T	Temperature_T	Drought
Rainfall_T	1.0	-0.24	-0.28	0.061
Pressure_T	-0.24	1.0	-0.2	0.034
Temperature_T	-0.28	-0.2	1.0	-0.052
Drought	0.061	0.034	-0.052	1.0

```

In [0]: df3.plot.box(grid='True')

Out[0]: <matplotlib.axes._subplots.AxesSubplot at 0x7f02036b8828>

```

```

In [0]: from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(df3.iloc[:,0:-1], df3.iloc[:, -1])

In [0]: from sklearn.naive_bayes import GaussianNB
id1=GaussianNB()

```

APPENDIX:

UI source code:

```
<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="utf-8" />

    <link rel="apple-touch-icon" sizes="76x76" href="static/img/corn.png" />

    <link rel="icon" type="image/png" href="static/img/corn.png" />

    <meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1" />

    <title>

      Drought Vizualization

    </title>

    <meta

      content="width=device-width,    initial-scale=1.0,    maximum-scale=1.0,    user-
scalable=0, shrink-to-fit=no"

      name="viewport"

    />

    <!--    Fonts and icons    -->

    <link

      href="https://fonts.googleapis.com/css?family=Montserrat:400,700,200"

      rel="stylesheet"

    />
```

```

<link

    href="https://maxcdn.bootstrapcdn.com/font-awesome/latest/css/font-
awesome.min.css"

    rel="stylesheet"

/>

<!-- CSS Files -->

<link href="static/css/bootstrap.min.css" rel="stylesheet" />

<link href="static/css/paper-dashboard.css?v=2.0.0" rel="stylesheet" />

<script          src="https://api.tiles.mapbox.com/mapbox-gl-js/v1.4.1/mapbox-
gl.js"></script>

<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>

<link

    href="https://api.tiles.mapbox.com/mapbox-gl-js/v1.4.1/mapbox-gl.css"

    rel="stylesheet"

/>

<link

    rel="stylesheet"

    type="text/css"

    href="//code.jquery.com/ui/1.9.2/themes/base/jquery-ui.css"

/>

<link

    rel="stylesheet"

```

```

    type="text/css"

    href="//cdn.leafletjs.com/leaflet-0.7/leaflet.css"

/>

<link rel="stylesheet" href="static/css/style.css" />

</head>

<!-- Sidebar -->

<body class="">

  <div class="wrapper ">

    <div class="sidebar" data-color="white" data-active-color="danger">

      <div class="logo">

        <a class="simple-text logo-mini">

          <div class="logo-image-small">

          </div>

        </a>

        <a class="simple-text logo-normal">

          Drought Prediction

        </a>

      </div>

```

```

<div class="sidebar-wrapper">

  <ul class="nav">

    <li>

      <a href="/drought">

        <i class="nc-icon nc-pin-3"></i>

        <p>Drought Map</p>

      </a>

    </li>

  </ul>

</div>

</div>

<div class="main-panel">

  <!-- Page Title -->

  <nav

    class="navbar navbar-expand-lg navbar-absolute fixed-top navbar-transparent"

  >

    <div class="container-fluid">

```



```

<div class="navbar-wrapper">

  <div class="navbar-toggle"></div>

  <a class="navbar-brand">Drought in the United States</a>

</div>

<div

  class="collapse navbar-collapse justify-content-end"

  id="navigation"

>

  <li class="nav overlay-top">

    <p>

      <label id="month"></label>

    </p>

  </li>

  <ul class="navbar-nav">

    <li class="nav-overlay top">

      <input

        id="slider"

        type="range"

        min="0"

        max="11"

        step="1"

```

```

        value="0"

    />

</li>

<li class="nav-overlay top">

</li>

<li class="nav overlay top">

    <!-- <div class="overlay-top"> -->

    <div id="legend" class="legend">

        <div class="bar"></div>

        <div>Level D0 to D4</div>

    </div>

    <!-- </div> -->

</li>

</ul>

</div>

</div>

</nav>

<!-- Content -->

<div class="content">

```

```

<div class="row">

  <div class="col-md-12">

    <div class="card ">

      <div class="card-header ">

        <h5 class="card-title"></h5>

      </div>

      <div class="card-body ">

        <canvas width="400" height="100"></canvas>

      </div>

      <div

        id="map"

        class="map"

        style="width:100%; height: 800px;"

      ></div>

      <nav id="menu"></nav>

      <div class="card-footer ">

        <p>Accompanying Text</p>

        <hr />

      </div>

    </div>

  </div>

</div>

```

```

    </div>

</div>

<footer class="footer footer-black footer-white "></footer>

</div>

</div>

<!-- Core JS Files -->

<script src="static/js/core/jquery.min.js"></script>

<script src="static/js/core/popper.min.js"></script>

<script src="static/js/core/bootstrap.min.js"></script>

<script src="static/js/plugins/perfect-scrollbar.jquery.min.js"></script>

<!-- Chart JS -->

<script src="static/js/plugins/chartjs.min.js"></script>

<script src="https://d3js.org/d3.v5.min.js" charset="utf-8"></script>

<script src="static/js/config.js"></script>

<script src="static/js/logic.js"></script>

<!-- Notifications Plugin -->

<script src="static/js/plugins/bootstrap-notify.js"></script>

<!-- Control Center for Now Ui Dashboard: parallax effects, scripts for the example
pages etc -->

<script

src="static/js/paper-dashboard.min.js?v=2.0.0"

type="text/javascript"

```

```
></script>

</body>

</html>
```

PREDICTION MODEL CODE

```
import pandas as pd

from sklearn.metrics import
accuracy_score,confusion_matrix,precision_score,recall_score

import matplotlib.pyplot as plt

import seaborn as sns

from google.colab import files

files.upload()

dataNagpur=pd.read_excel("Temperature_pandas.xlsx",sheet_name="Nagpur")

dataPune=pd.read_excel("Temperature_pandas.xlsx",sheet_name="Pune")

dataNashik=pd.read_excel("Temperature_pandas.xlsx",sheet_name="Nashik")

dataAurangabad=pd.read_excel("Temperature_pandas.xlsx",sheet_name="Auranga
bad")

dataAmaravati=pd.read_excel("Temperature_pandas.xlsx",sheet_name="Amravati")

dataSolapur=pd.read_excel("Temperature_pandas.xlsx",sheet_name="Solapur")

dataYavatmal=pd.read_excel("Temperature_pandas.xlsx",sheet_name="Yavatmal")

dataLatur=pd.read_excel("Temperature_pandas.xlsx",sheet_name="Latur")
```

```

dataNagpur=pd.read_excel("Rainfall.xlsx",sheet_name="Nagpur")

dataPune=pd.read_excel("Rainfall.xlsx",sheet_name="Pune")

dataNashik=pd.read_excel("Rainfall.xlsx",sheet_name="Nashik")

dataAurangabad=pd.read_excel("Rainfall.xlsx",sheet_name="Aurangabad")

dataAmaravati=pd.read_excel("Rainfall.xlsx",sheet_name="Amravati")

dataSolapur=pd.read_excel("Rainfall.xlsx",sheet_name="Solapur")

dataYavatmal=pd.read_excel("Rainfall.xlsx",sheet_name="Yavatmal")

dataLatur=pd.read_excel("Rainfall.xlsx",sheet_name="Latur")

datak=dataNagpur.append(dataPune)

datak=datak.append(dataNashik)

datak=datak.append(dataAurangabad)

datak=datak.append(dataAmaravati)

datak=datak.append(dataSolapur)

datak=datak.append(dataYavatmal)

datak=datak.append(dataLatur)


dataf=((datak-datak.min())/(datak.max()-datak.min()))*1


dataNagpur=dataf.iloc[0:16,:]

dataPune=dataf.iloc[16:32,:]

dataNashik=dataf.iloc[32:48,:]

```

```
dataAurangabad=dataf.iloc[48:64,:]
```

```
dataAmaravati=dataf.iloc[64:80,:]
```

```
dataSolapur=dataf.iloc[80:96,:]
```

```
dataYavatmal=dataf.iloc[96:112,:]
```

```
dataLatur=dataf.iloc[112:128,:]
```

```
dataNagpurTransposed=dataNagpur.T
```

```
dataPuneTransposed=dataPune.T
```

```
dataNashikTransposed=dataNashik.T
```

```
dataAurangabadTransposed=dataAurangabad.T
```

```
dataAmaravatiTransposed=dataAmaravati.T
```

```
dataSolapurTransposed=dataSolapur.T
```

```
dataYavatmalTransposed=dataYavatmal.T
```

```
dataLaturTransposed=dataLatur.T
```

```
dataNagpurTransposed['mean']=dataNagpurTransposed.mean(axis=1)
```

```
dataPuneTransposed['mean']=dataPuneTransposed.mean(axis=1)
```

```
dataNashikTransposed['mean']=dataNashikTransposed.mean(axis=1)
```

```
dataAurangabadTransposed['mean']=dataAurangabadTransposed.mean(axis=1)
```

```
dataAmaravatiTransposed['mean']=dataAmaravatiTransposed.mean(axis=1)
```

```
dataSolapurTransposed['mean']=dataSolapurTransposed.mean(axis=1)
```

```
dataYavatmalTransposed['mean']=dataYavatmalTransposed.mean(axis=1)

dataLaturTransposed['mean']=dataLaturTransposed.mean(axis=1)

columns=[2001,2002,2003,2004,2005,2006,2007,2008,2009,2010,2011,2012,2013,
2014,2015,2016,'mean']

index=[0,1,2,3,4,5,6,7,8,9,10,11]

dataNagpurTransposed.columns=columns

dataPuneTransposed.columns=columns

dataNashikTransposed.columns=columns

dataAurangabadTransposed.columns=columns

dataAmaravatiTransposed.columns=columns

dataSolapurTransposed.columns=columns

dataYavatmalTransposed.columns=columns

dataLaturTransposed.columns=columns


dataNagpurTransposed.index=index

dataPuneTransposed.index=index

dataNashikTransposed.index=index

dataAurangabadTransposed.index=index

dataAmaravatiTransposed.index=index

dataSolapurTransposed.index=index

dataYavatmalTransposed.index=index

dataLaturTransposed.index=index
```



```

writer = pd.ExcelWriter('changed_Rainfall.xlsx', engine='xlsxwriter')

dataNagpurTransposed.to_excel(writer, sheet_name='Nagpur')

dataPuneTransposed.to_excel(writer, sheet_name='Pune')

dataNashikTransposed.to_excel(writer, sheet_name='Nashik')

dataAurangabadTransposed.to_excel(writer, sheet_name='Aurangabad')

dataAmaravatiTransposed.to_excel(writer, sheet_name='Amaravati')

dataSolapurTransposed.to_excel(writer, sheet_name='Solapur')

dataYavatmalTransposed.to_excel(writer, sheet_name='Yavatmal')

dataLaturTransposed.to_excel(writer, sheet_name='Latur')

writer.save()

print(dataAmaravatiTransposed)


from sklearn.model_selection import train_test_split

x_train,x_test,y_train,y_test = train_test_split(df3.iloc[:,0:-1],df3.iloc[:,-1:])

from sklearn.naive_bayes import GaussianNB

id1=GaussianNB()

id1.fit(x_train, y_train)

predicted1= id1.predict(x_test)

accuracy1=accuracy_score(y_test,predicted1)

```

```
precision1=precision_score(y_test,predicted1,average='macro')
```

```
recall1=recall_score(y_test,predicted1,average='macro')
```

```
cm1=confusion_matrix(y_test,predicted1)
```

```
dfcm1=pd.DataFrame(cm1)
```

```
plt.figure(1)
```

```
sns.heatmap(dfcm1,annot=True)
```

```
plt.title(' Confusion Matrix, Accuracy=%f, Precision=%f,  
Recall=%f'%(accuracy1,precision1,recall1))
```

```
plt.show()
```

```
from sklearn import tree
```

```
id2 = tree.DecisionTreeClassifier(criterion = 'gini')
```

```
id2.fit(x_train, y_train)
```

```
predicted2= id2.predict(x_test)
```

```
accuracy2=accuracy_score(y_test,predicted2)
```

```
precision2=precision_score(y_test,predicted2,average='macro')
```

```
recall2=recall_score(y_test,predicted2,average='macro')
```

```
cm2=confusion_matrix(y_test,predicted2)
```

```
dfcm2=pd.DataFrame(cm2)
```

```
plt.figure(1)
```

```
sns.heatmap(dfcm2,annot=True)
```

```
plt.title(' Confusion Matrix, Accuracy=%f, Precision=%f,  
Recall=%f'%(accuracy2,precision2,recall2))  
  
plt.show()
```

```
from sklearn.ensemble import RandomForestClassifier
```

```
id3 = RandomForestClassifier()  
  
id3.fit(x_train, y_train)  
  
predicted3= id3.predict(x_test)  
  
accuracy3=accuracy_score(y_test,predicted1)  
  
precision3=precision_score(y_test,predicted1,average='macro')  
  
recall3=recall_score(y_test,predicted1,average='macro')  
  
cm3=confusion_matrix(y_test,predicted3)  
  
dfcm3=pd.DataFrame(cm3)  
  
plt.figure(1)  
  
sns.heatmap(dfcm3,annot=True)  
  
plt.title(' Confusion Matrix, Accuracy=%f, Precision=%f,  
Recall=%f'%(accuracy3,precision3,recall3))  
  
plt.show()
```