BACHELOR OF SCIENCE IN
COMPUTER SCIENCE AND ENGINEERING

Inspiring Excellence

Crop Prediction Based on Geographical and Climatic Data Using Machine Learning and Deep Learning

Authors
Al Amin Alif
Israt Farhana Shukanya
Tasnia Nobi Afee

Supervisor
Hossain Arif
Assistant Professor
Department of Computer Science and Engineering

A thesis submitted to the Department of CSE
in partial fulfillment of the requirements for the degree of
B.Sc. Engineering in CSE

Department of Computer Science and Engineering
BRAC University, Dhaka - 1212, Bangladesh

December 2018

Declaration

We hereby declare that this thesis is based on the results we obtained from our work. Due acknowledgement has been made in the text to all other material used. This thesis, neither in whole nor in part, has been previously submitted by anyone to any other university or institute for the award of any degree.

Authors:

Al Amin Alif
Student ID: 14101009

Israt Farhana Shukanya
Student ID: 14101186

Tasnia Nobi Afee
Student ID: 14301052

Supervisor:

Hossain Arif
Assistant Professor, Department of Computer Science and Engineering
BRAC University

December 2018

The thesis titled
"Crop Prediction Based on Geographical and Climatic Data Using Machine Learning and Deep Learning"
Submitted by:

Al Amin Alif Student
ID: 14101009
Israt Farhana Shukanya
Student ID: 14101186
Tasnia Nobi Afee
Student ID: 14301052

of Academic Year Fall 2018 has been found as satisfactory and accepted as partial fulfillment of the requirement for the Degree of Bachelor of Computer Science and Engineering

1. Hossain Arif
Assistant Professor
Department of Computer Science and Engineering
BRAC University

2. Md. Abdul Mottalib, PhD
Professor and Chairperson
Department of Computer Science and Engineering
BRAC University

Acknowledgements

Firstly, we would like to thank the Almighty for giving us the ability to successfully conduct and complete our research to the fulfillment of our eventual target. Secondly, we would like to thank our supervisor Hossain Arif sir for his constant support, fair evaluation, feedback, guidance and contribution for this research. We always found him helpful every time we tried seeking his help. The resources he made available for us, the platform he brought upon us, and the excellent supervision he gave us was a strong factor in our success in completing this research. We extend our gratitude to our parents and friends, who constantly helped us with inspiration and motivation, and also valuable suggestions. We would also like to acknowledge our fellow researchers who also played a crucial part in the development of our research due to the fact that we learned a lot from the resources and advice they showered us with. Last, but definitely not the least, we would like to thank BRAC University for providing us this brilliant opportunity to conduct a research of this sort and for giving us a chance to complete our Bachelor Degree from here.

Abstract

Agriculture is the basic source of food supply in all the countries of the world—whether underdeveloped, developing or developed. Besides providing food, this sector has contributions to almost every other sector of a country. According to the Bangladesh Bureau of Statistics (BBS), 2017, about 17 % of the country's Gross Domestic Product (GDP) is a contribution of the agricultural sector, and it employs more than 45% of the total labor force. In light of the decreasing crop production and shortage of food across the world, one of the crucial criteria of agriculture now-a-days is selecting the right crop for the right piece of land at the right time. Therefore, in our research we have proposed a method which would help suggest the most suitable crop(s) for a specific land based on the analysis of the data of previous years on certain affecting parameters using machine learning. In our work, we have implemented Random Forest Classifier, Gaussian Naïve Bayes, Logistic Regression, Support Vector Machine, k-Nearest Neighbor, and Artificial Neural Network for crop selection. We have trained these algorithms with the training data and later these were tested with test dataset. We then compared the performances of all the tested methods to arrive at the best outcome.

Keywords: Crop Selection, Machine Learning Algorithms, Artificial Neural Network.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

1.1 Overview

For a country, one of the most crucial aspects of its development circles around its capacity to produce food. For decades, agriculture has been associated with the production of essential food crops. The rate of urbanization at present is by-far the most superior aim of our civilization. In doing this, we are ignorantly diminishing our capacity for agriculture; especially in terms of land and fertility. As the amount of land will not be increasing in this era of urbanization and globalization, we will have to focus on making the most of what we have. Due to this issue, we have to device newer ways to farm arable lands and extract the absolute most from these limited land resources. In this age of technology and data-science, if implemented properly, the agricultural sector may also be greatly affected. It is true that a farmer is the best decider of crop selection and crop cultivation. However, machine learning techniques can be applied in this field for far greater precision and stability of selection. In this research, we have attempted to come up with a few techniques that will lead us to choose suitable crops based on specific state, specific district, season, and some other environmental aspects.

1.2 Motivation

On this research, we primarily focused on creating a platform that can be implemented on a national scale. We experienced our primary motivation from some worldwide statistics, Bangladeshi national statistics, and personal experience as well. According to World Food Programme, one-in-nine people around the world, even on this day go to sleep on an empty stomach every night. That is about 821 million people. In our country, which is a developing

country, the scenario is far worse. According to BRAC the agricultural land in Bangladesh is shrinking by 1% annually, while the population is growing by 1.2% due to rising sea-level, extreme weather patterns, frequent flooding, and loss of arable land. All these lead to a growing and sustained threat to food security. Food security is a striking global issue these days. The crisis is increasing with absolute severity. Ways to avoid such calamity should be looked up with great emergency. In this era of technology, a sector as important as the agricultural sector should not go without utilizing the perks of it. Machine learning is used worldwide these days with great efficiency in sectors such as forecasting, pattern recognition, fraud/fault detection, prediction, virtual assistance, image processing, robotics, artificial intelligence and so on and so forth. The agricultural sector has had milestone work done on it through weather forecasting, crop disease prediction, yield prediction, etc. However, what we propose, which is crop prediction itself, has not yet been implemented on a mass scale.

1.3 Thesis Outline

The rest of the thesis is organized as follows:

- Chapter 2: The necessary background information regarding the proposed approach.
- Chapter 3: The algorithms used for classification.
- Chapter 4: The proposed model of the research including discussion about data collection, data pre-processing, data splitting, algorithm fitting,testing accuracy, data post-processing and ANN.
- Chapter 5: The experimental results with graphical representation.
- Chapter 6: The conclusion and future work.

Chapter 2

Background Analysis

2.1 Literature Review

Machine learning is the branch of computer science which is used to build algorithms which exhibit self-learning property i.e. learning which is done by the machine itself hence the term "Machine Learning". It is viewed as one of the significant areas under Artificial Intelligence. To show intelligence machine needs to interpret and analyze the input. After analyzing it the result data apart from simply following the instructions on that data. This is the thing that machine learning algorithms do. Machine learning centers on the development of computer programs that can get data and utilize it to learn for themselves. The way toward learning starts with perceptions on information, for example, direct experience, or instruction, in order to look for patterns in data. It helps to make better decisions in the future based on the examples that we give. The essential point is to permit the computer learn automatically without human intercession or help and regulate actions consequently. It is a major field in computer science which is being utilized in various forms of progressive technological development programs all around the world. It is regarded as the future of engineering and Artificial Intelligence [13]. In another research, the author K. Kaur utilizes Machine learning in different applications in Indian agriculture [6]. In this paper the various applications of machine learning techniques in agriculture have been listed such as Crop Selection and Crop Yield Prediction, Weather Forecasting, Smart Irrigation System, Crop Disease Prediction, Deciding the Minimum Support Price. These techniques will enhance the productivity of fields along with a reduction in the input efforts of the farmers. Along with the advances in machines and technologies used in farming, useful and accurate information about different matters also plays a significant role in it [5]. Machine learning provides many effective algorithms which can identify the input and output relationship in crop selection and yield prediction. To this issue of crop selection using machine learning some researches use

Markov logic of crop rotations for early crop mapping. A Markov logic network (MLN) is a probabilistic logic which applies the ideas of a Markov network to first-order logic, enabling uncertain inference. The obtained results show that the proposed approach is able to predict the crop type of each field, before the beginning of the crop season, with an accuracy as high as 60%, which is better than the results obtained with the previous approaches based on remote sensing imagery [14].

The authors S. Ying-xue, X. Huan, and Y. Li-jiao in their research work describes a crop model as a computer program that mathematically describes and models the principles of harvest growth and can be utilized to quantitatively and dynamically clarify the procedure of product development, improvement, yield, and response to ecological change[16]. Crop models can be classified as harvest factual models and crop simulation models (or crop growth models) in light of the fundamental numerical technique for the displaying. Furthermore, in another paper, the authors used Auto-regressive Integrate Moving Average (ARIMA) and The Support Vector Machines (SVMs)[17]. Various examinations were performed on the advancement of SVM and the most accuracy show by utilizing statistical criteria was additionally selected. The proposed model of Support Vector Machine (SVM) can conjecture nonlinear or linear forecasting function upon kernel function . These are a few of the ideas which have been additionally connected to increase better estimating of farming utilizing machine learning. An UchooBoost Algorithm is utilized for testing with exactness farming. It is supervised learning based on algorithm. The best characteristic of UChooBoost is that it can be applied for an extended data expression and works on compounding hypotheses which leads to improve algorithm performance [9]. Agriculture planning performs a widespread role in financial boom and food security of agro-based country. Selection of crop(s) is a crucial trouble for agriculture planning. It relies upon different parameters which includes production rate, market price and government approaches. Using statistics methods or machine learning techniques numerous scientists research prediction of yield rate of crop, prediction of weather, soil classification and crop classification. If there is in excess of one alternative to plant a product at any given moment utilizing constrained land asset, at that point determination of harvest is a riddle. However, this paper proposed a method named Crop Selection Method (CSM) can be applied to solve this crop selection problem and maximize net yield rate of crop over season and subsequently achieve maximum economic growth of the country. Crop selection method refers to a method of selecting crop(s) over a specific season depending upon various environmental as well as economic factors for the maximum benefit. These factors are precipitation levels, average temperature, soil type, market prices and demand, prevailing farm conditions, crop or varietal adaptability, resistance to pests and diseases, farming system, available technology etc. Right choice in the

selection of crop or crops to be grown, particularly perennial types, will eventually convert into a successful cultivating venture [11]. This task can be completed using Classification algorithms of WEKA. Waikato Environment for Knowledge Analysis (Weka) is a suite of machine learning software written in Java, developed at the University of Waikato, New Zealand. It is free software licensed under the GNU General Public License . In some researches, WEKA Classifiers and regression methods are used to precisely predict the most suitable crop(s) to be grown in a particular season [11]. There are many more features such as humidity, soil nutrition value, pH etc., which are included in the training dataset.

2.2 Background Study

Crop selection is a process which will lead us to specify a particular crop or a group of crops that is the most suitable based on specific pieces of land, on a specific geographical area, on a particular time of the year. All the factors considered by us are directly involved in the yield of the crops that we are predicting. There can be different factors affecting the production of a crop such as soil condition, pH, nitrogen, phosphate, potassium, organic carbon, calcium, manganese, copper, iron, depth, rainfall, temperature, humidity, price etc. As mentioned before, we have implemented various machine learning algorithms on this research for crop prediction. Machine learning is a field of artificial intelligence that uses statistical techniques to give computer systems the ability to "learn" from dataset without being explicitly programmed. Machine learning is closely related to computational statistics, which also focuses on prediction making through the use of computers [12]. The algorithms that we used were also highly efficient for this particular task; hence the usage. Random Forest Classifier is a statistical algorithm that is used to cluster points of data in functional groups. When the dataset is large and/or there are many variables, it becomes difficult to cluster, because not all the variables can be taken into account[2]. Therefore, the algorithm can also give a certain chance that a data point belongs in a certain group. For Gaussian Naïve Bayes, when dealing with continuous data, a typical assumption is that the continuous values associated with each class are distributed according to a Gaussian distribution. Naïve Bayes Classifiers are a family of simple "probabilistic classifiers" based on applying Bayes theorem with strong independence assumption between the features. Logistic Regression is a classification method that generalizes logistic regression to multi-class problems, i.e. with more than two possible discrete outcomes. That is, it is a model that is used to predict the probabilities of the different possible outcomes of a categorically distributed dependent variable, given a set of independent variables. Support vector machine (SVM) is a supervised learning model with associated learning algorithms that analyze data used for classification

and regression analysis. Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier[4][16]. k-Nearest Neighbors algorithm (k-NN) is a non-parametric method used for classification and regression[1]. In both cases, the input consists of the k closest training examples in the feature space. The output depends on whether k-NN is used for classification or regression. Artificial neural networks (ANN) or connectionist systems are computing systems vaguely inspired by the biological neural networks that constitute animal brains. The neural network itself is not an algorithm, but rather a framework for many different machine learning algorithms to work together and process complex data inputs. Such systems "learn" to perform tasks by considering examples, generally without being programmed with any task-specific rules [12].

2.3 Supervised Learning

Data mining is one of the most significant application of machine learning. Human intelligence, although extremely powerful, is error prone in many instances. People learn and apply that knowledge based on what they remember from what they had learned, or simply make a bargain judgment. Machine learning, more specifically data mining, does the same task in the same way. However, the machine does not have the tendency to forget what it had learned. Although, unlike human beings, it cannot make that crunch judgment call, it can still predict outcomes on a certain accuracy level. All this is possible due to the existence of supervised learning[10]. Supervised machine learning is the search for algorithms that reason from externally supplied instances to produce general hypotheses, which then make predictions about future instances. In other words, the goal of supervised learning is to build a concise model of the distribution of class labels in terms of predictor features. The resulting classifier is then used to assign class labels to the testing instances where the values of the predictor features are known, but the value of the class label is unknown. It simply is the task of learning a function that maps an input to an output based on input-output pairs of examples. A supervised learning algorithm analyzes the training data and produces an inferred function, which can be used for mapping new examples. An optimal scenario will allow for the algorithm to correctly determine the class labels for unseen instances. This requires the learning algorithm to generalize from the training data to unseen situations in a "reasonable" way. A flow chart of supervised learning is given below in Fig. 2.1.

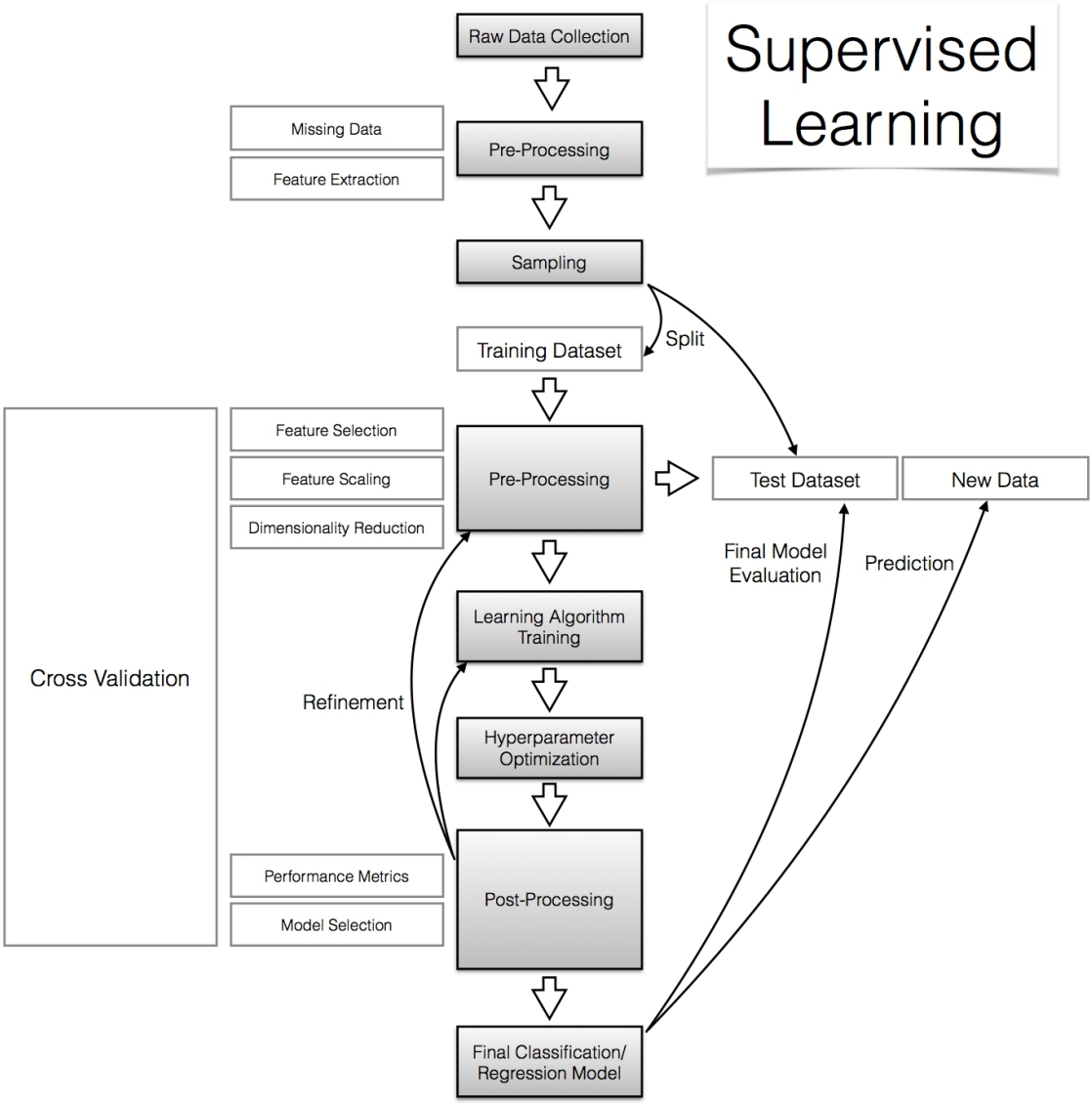

Fig. 2.1 Flow Chart of Supervised Learning

Supervised learning models have some advantages over the unsupervised approach, but they also have limitations. The systems are more likely to make judgments that humans can relate to, for example, because humans have provided the basis for decisions. However, in the case of a retrieval-based method, supervised learning systems have trouble dealing with new information. If a system with categories for crops and fruits is presented with a flower, for example, it would have to be incorrectly lumped in one category or the other. If the AI system was generative, however, it may not know what the flower is but would be able to recognize it as belonging to a separate category.

2.4 Data Classification

Data Mining has three major components: Classification or Clustering, Association Rules, and Sequence Analysis. By simple definition, classification and clustering analyze a set of data and generate a set of grouping rules which can be used to classify future data. Classification is used to classify each item in a set of data into one of predefined set of classes or groups. The data analysis task of classification is where a model or classifier is constructed to predict categorical labels or the class label attributes. Classification is a data mining function that assigns items in a collection to target categories or classes. The goal of classification is to accurately predict the target class for each case in the data. For example, a classification model could be used to identify classification is used to classify each item in a set of data into one of predefined set of classes or groups. The data analysis task classification is where a model or classifier is constructed to predict categorical labels (the class label attributes)[1]. Classification is a data mining function that assigns items in a collection to target categories or classes. The goal of classification is to accurately predict the target class for each case in the data. For example, a classification model could be used to identify a particular crop as to the possibility of its production given a set of features, as per our research. A classification task begins with a data set in which the class assignments are known. For example, a classification model that predicts crop selection could be developed based on observed data for many crop selections over a period of time. In addition to the crop data, the data might track a number of features. A specific crop would be the target, the other attributes would be the predictors, and the data for each crop would constitute a case. Classifications are discrete and do not imply order. Continuous, floating-point values would indicate a numerical, rather than a categorical, target. A predictive model with a numerical target uses a regression algorithm, not a classification algorithm. The simplest type of classification problem is binary classification. In binary classification, the target attribute has only two possible values: for example, rice or no rice. Multi-class targets have more than two values: for example, low, medium, high, or unknown possibility of rice production. In the model build (training) process, a classification algorithm finds relationships between the values of the predictors and the values of the target[7]. Different classification algorithms use different techniques for finding relationships. These relationships are summarized in a model, which can then be applied to a different data set in which the class assignments are unknown.

Chapter 3

Algorithms for Proposed Model

3.1 Logistic Regression

Methods involving regression are essential to any data analysis models which attempt to describe the association between a response variable and any number of predictor variables. Situations involving discrete variables constantly arise. For instance, the dataset we have implemented has an outcome involving the presence or absence of a particular crop, given a set of features. Logistic regression analysis extends the techniques of multiple regression analysis to investigate and inquire situations in which the outcome is categorical, which is, taking on multiple values [15]. This is a very basic branch of data science. Although the name suggests a regression technique, logistic regression is a statistical classification model which deals with categorical dependent variables. Classification is decision. To make an optimal decision, we need to assess the utility function, which implies that we need to account for the uncertainty in the outcome, i.e. a probability. Logistic regression is emphatically not a classification algorithm on its own. It is only a classification algorithm in combination with a decision rule that makes dichotomous the predicted probabilities of the outcome. This is one of the very first algorithm any machine learning practitioner attempts when faced with a classification problem. The basic mechanism and output of this algorithm is similar to many other machine learning algorithms. It is the appropriate regression analysis to conduct when the dependent variable is dichotomous or binary. Like all regression analyses, the logistic regression is a predictive analysis. Logistic regression is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval or ratio-level independent variables.

This algorithm works with binary data, where either the event happens, represented by "1", or the event does not happen, represented by 0. So given some feature "X", it tries to find out whether some event "y" happens or not. So "y" can either be "0" or "1". In the case

where the event happens, “y” is given the value “1”. If the event does not happen, then “y” is given the value of “0”. For example, if “y” represents whether a particular crop among a huge variety of crops, then “y” will be “1” if the crop does grow or “y” will be “0” if it does not. This is known as Binomial Logistic Regression. There is also another form of Logistic Regression which uses multiple values for the variable “y”. This form of Logistic Regression is known as Multinomial Logistic Regression. Fig. 3.1 shows a simple flow chart representation of logistic regression.

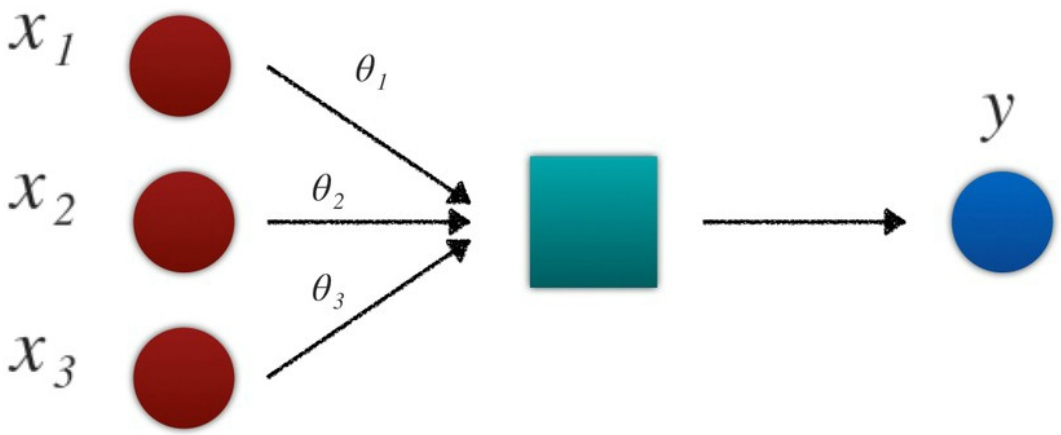

Fig. 3.1 Flow Chart of Logistic Regression

Logistic Regression uses the logistic function to find a model that fits with the data points. The function gives an “S” shaped curve to model the data. The curve is restricted between “0” and “1”, so it is easy to apply when “y” is binary. Logistic Regression can then model events better than linear regression, as it shows the probability for “y” being “1” for a given “x” value. Logistic Regression is used in statistics and machine learning to predict values of an input from previous test data. Scatter plot classification of data by Logistic Regression is shown in Fig. 3.2

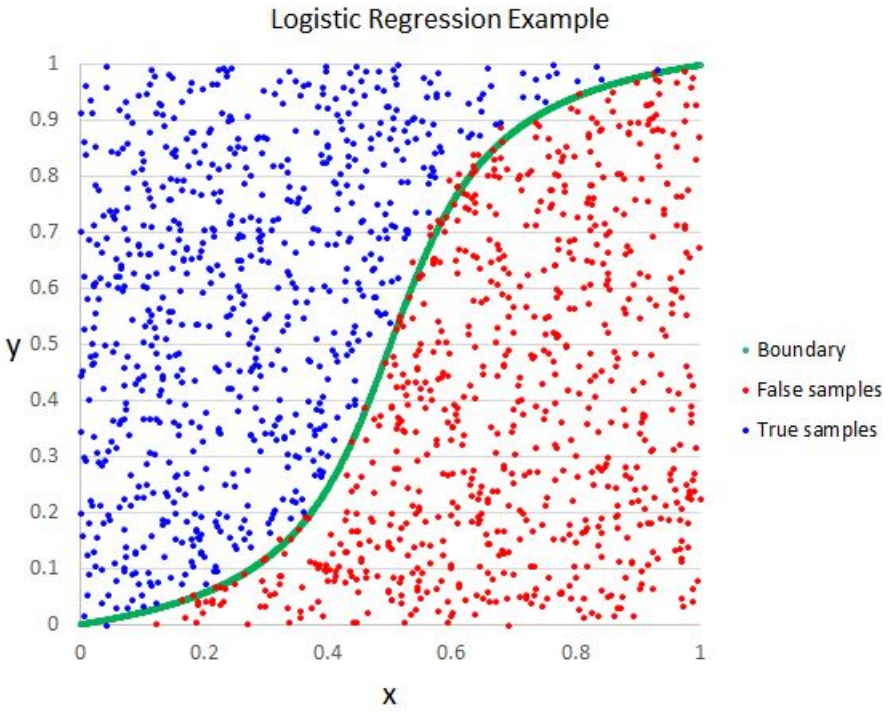

Fig. 3.2 Scatter Plot Example of Logistic Regression

3.2 Random Forest

Random forests are a combination of tree predictors such that each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest. The generalization error for forests converges to a limit as the number of trees in the forest becomes large. The generalization error of a forest of tree classifiers depends on the strength of the individual trees in the forest and the correlation between them. Using a random selection of features to split each node yields error rates that compare favorably to Adaboost, but are more robust with respect to noise [3]. Internal estimates monitor error, strength, and correlation and these are used to show the response to increasing the number of features used in the splitting. Internal estimates are also used to measure variable importance. These ideas are also applicable to regression. Random Forest is a highly flexible and easy to use machine learning algorithm that produces, even without hyper-parameter tuning, a great result most of the time. It is also one of the most used algorithms, because of its simplicity and the fact that it can be used for both classification and regression tasks. In this research, we have used the classification aspect of this algorithm based on our necessity.

Using this random forest classifier, we have successfully achieved a very high accuracy upon implementation on our dataset.

Random forest is a supervised learning algorithm. As the name suggests, this algorithm creates a forest and using precision techniques, makes it random. The "forest" it builds, is an ensemble of Decision Trees, which are mostly trained with the "bagging" method. The general idea of the bagging method is that a combination of learning models increases the overall result. To say it in simple words: Random forest builds multiple decision trees and merges them together to get a more accurate and stable prediction[2]. One big advantage of random forest is, that it can be used for both classification and regression problems, which form the majority of current machine learning systems. We have implemented random forest in classification, since classification is sometimes considered the building block of machine learning, and the simple fact that it was necessary for us to use that aspect of the algorithm based on the type of research we have up taken. Fig. 3.3 shows random forest depiction.

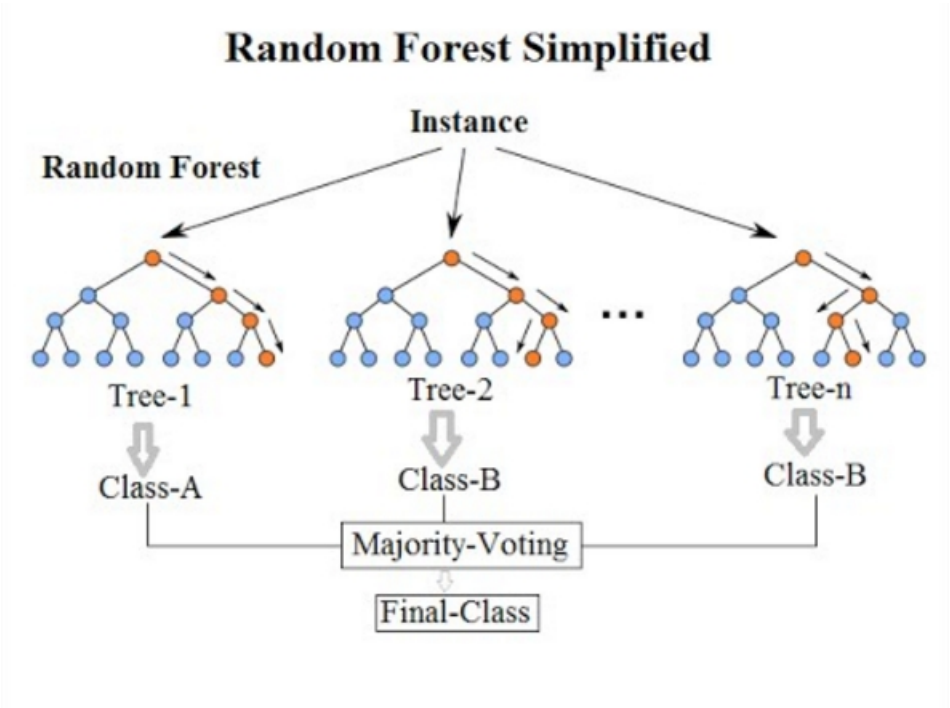

Fig. 3.3 Random Forest Depiction

Random Forest has nearly the same hyper parameters as a decision tree or a bagging classifier. Fortunately, we do not have to combine a decision tree with a bagging classifier and can just easily use the classifier-class of Random Forest. As mentioned previously, with Random Forest, we can also deal with Regression tasks by using the Random Forest Regression. Random Forest adds additional randomness to the model, while growing the

trees. Instead of searching for the most important feature while splitting a node, it searches for the best feature among a random subset of features. This results in a wide diversity that generally results in a better model. Therefore, in Random Forest, only a random subset of the features is taken into consideration by the algorithm for splitting a node. We can even make trees more random, by additionally using random thresholds for each feature rather than searching for the best possible thresholds (like a normal decision tree does). Another successful reason for choosing this algorithm is that it is very easy to measure the relative importance of each feature on the prediction with great precision and logic. Scikit-learn provides a great tool for this, that measures the importance of a feature by looking at how much the tree nodes, which use that feature, reduce impurity across all trees in the forest. It computes this score automatically for each feature after training and scales the results, so that the sum of all importance is equal to 1. One of the key reason for using the random forest classifier, and not something simpler and more straight-forward such as a decision tree, is the fact that we wanted compact output levels. We have a data intensive dataset, which might have hampered the precision and efficiency of a decision tree. If we input a training dataset with features and labels into a decision tree, it will formulate some set of rules, which will be used to make the predictions. In comparison, the Random Forest algorithm randomly selects observations and features to build several decision trees and then averages the results. It is more like using the decision tree algorithm, but over and over again; hence, increasing the viability of the prediction. Deep decision trees might suffer from over fitting. Random Forest prevents over fitting most of the time, by creating random subsets of the features and building smaller trees using these subsets. Afterwards, it combines the sub-trees. Note that this does not work every time and that it also makes the computation slower, depending on the number of trees the random forest builds.

3.3 Naïve Bayes

Naive Bayes is one of the most efficient and effective inductive learning algorithms for machine learning and data mining. It is a classifying algorithm that uses the Bayes theorem to calculate the probability and then classify data. It is an intuitive supervised learning approach that simplifies the calculation by assuming that the probability of each attribute is independent of all other attributes. This is why the classifier is called naïve. Although this is a strong assumption it takes into account, but the method is rather very fast and effective. The main key insight of Bayes' theorem is that the probability of an event can be adjusted as new data is introduced. A Naïve Bayes model is easy to build and it has no complicated iterative parameters estimation which makes it particularly useful for very large datasets [8].

As mentioned before, Naïve Bayes uses Bayes theorem and from the probability perspective, Bayes' theorem states the following relationship,

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{3.1}$$

Here A and B are two events and where A is class variable and B is a dependent feature vector (of size n). As a result,

$$B = (b_1, b_2, b_3, \ldots\ldots\ldots b_n) \tag{3.2}$$

And,

- P(A|B) is the conditional probability that event A occurs , given that B has occurred. This is also known as the posterior probability. (A) and P(B) are the probabilities of A and B respectively without regard of each other. (B|A) is the conditional probability that event B occurs , given that A has occurred.

Now, from the machine learning perspective, it can be assumed that A is the response variable and B is the input attribute. So according to the equation,

P(A|B) : The conditional probability of response variable belonging to a particular value, given the input attributes. This is also known as the posterior probability.

P(A) : The prior probability of the response variable.

P(B) : The probability of training data or the evidence.

P(B|A) : This is known as the likelihood of the training data.

Therefore, the above equation can be rewritten as

$$Posterior = \frac{Prior * likelihood}{evidence} \tag{3.3}$$

The evidence can be spited into the independent parts. Now, if any two events A and B are independent, then,

$$P(A,B) = P(A)P(B) \tag{3.4}$$

Hence, the equation become,

$$P(A|b_1\ldots\ldots\ldots b_n) = \frac{P(A)P(b_1\ldots\ldots\ldots b_n|A)}{(P(b_1\ldots\ldots..b_n))} \tag{3.5}$$

This can be expressed as,

$$P(A|b_1 \ldots\ldots\ldots b_n) = \frac{1}{z} P(A) \prod_{i=1}^{n} P(b_i|A) \tag{3.6}$$

Here, z is factor dependent of b_1 though b_n,

In spite the fact that the Naïve Bayes models are very simple, it often works surprisingly well in many real-world situations, famously document classification and spam filtering. Moreover, it is a decent method and is widely used because it outperforms more sophisticated classification methods. Depending on the dataset, different Naïve Bayed model is applied to classify the dataset. There are three available models in the Sklearn python library:

- Gaussian: It assumes that continuous features follow a normal distribution.
- Multinomial: It is useful if the features are discrete.
- Bernoulli: The binomial model is useful if the feature vectors are binary.

For our research we have used Gaussian Naïve Bayes Classifier model. In Gaussian Naive Bayes, continuous values associated with each feature are assumed to be distributed according to a Gaussian distribution or normal distribution.

3.4 Support Vector Machine (SVM)

"Support Vector Machine" (SVM) is a supervised machine learning algorithm which is a very useful technique for data classification. However, this learning algorithm can also be used for regression challenges. A classification task usually involves separating data into training and testing sets. Each instance in the training set contains one "target value" (i.e. the class labels) and several "attributes" (i.e. the features or observed variables). The goal of SVM is to produce a model (based on the training data) which predicts the target values of the test data given only the test data attributes [4].

3.4.1 Linear SVM

Support Vector Machine classifier plots each data item with the value of each feature as a point in an n-dimensional space (where n is number of features) being the value of a particular coordinate. SVM maps data to a high-dimensional feature space so that data points can be categorized, even when the data are not otherwise linearly separable. Then, it performs classification by finding the hyper-plane that differentiates the two classes very well.

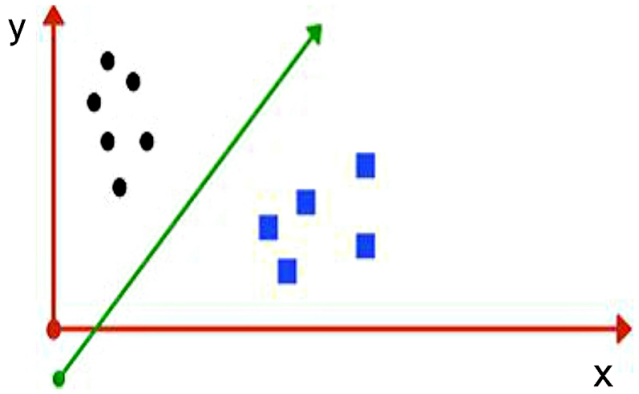

Fig. 3.4 Classifications of Two Classes Using Boundary

When the data can be linearly separated in two dimensions, as shown in Fig. 3.4, any machine learning algorithm tries to find a boundary that divides the data in such a way that the mis-classification error can be minimized. Nevertheless, there can be several boundaries that correctly divide the data points as shown below in Fig. 3.5 .

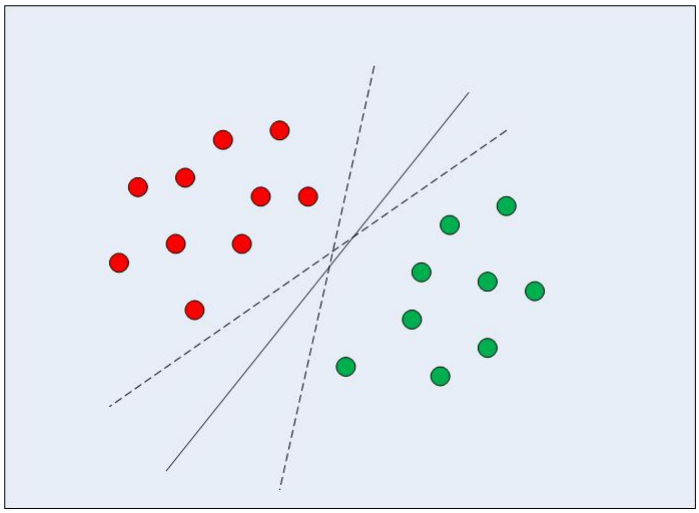

Fig. 3.5 Multiple Decision Boundaries

SVM is different from the other classifiers in the way that it chooses the decision boundary that maximizes the distance from the nearest data points of all the classes. This boundary has the maximum margin from the nearest points of the training class as well as the test class. . As a result, SVM classifier does not only find a boundary; it finds the most optimal decision boundary. This boundary resulting from SVM is called the maximum margin classifier, or the maximum margin hyper plane. The nearest points from the hyper plane that maximize

the distance between the decision boundaries are called support vectors. SVM margin is shown below in Fig. 3.6

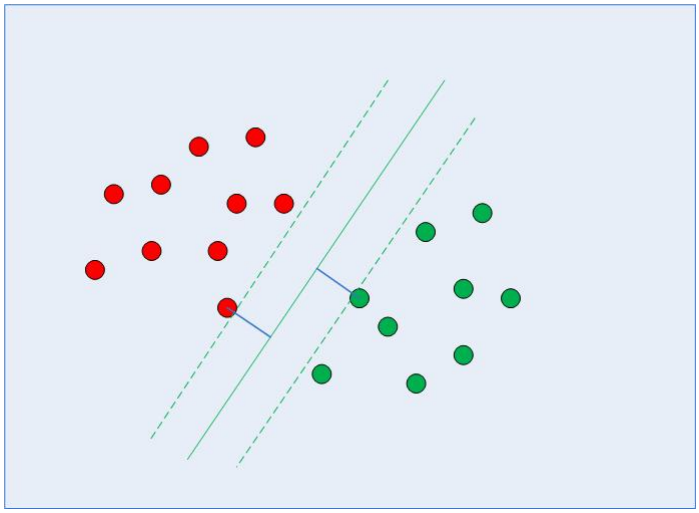

Fig. 3.6 Maximum Hyper-Plane with Support Vectors

3.4.2 Kernel SVM

In real world, data is almost never as clean as shown in the previous figures. In the case of non-linearly separable data, such as the one shown below in Fig. 3.7, a straight line cannot be used as a decision boundary.

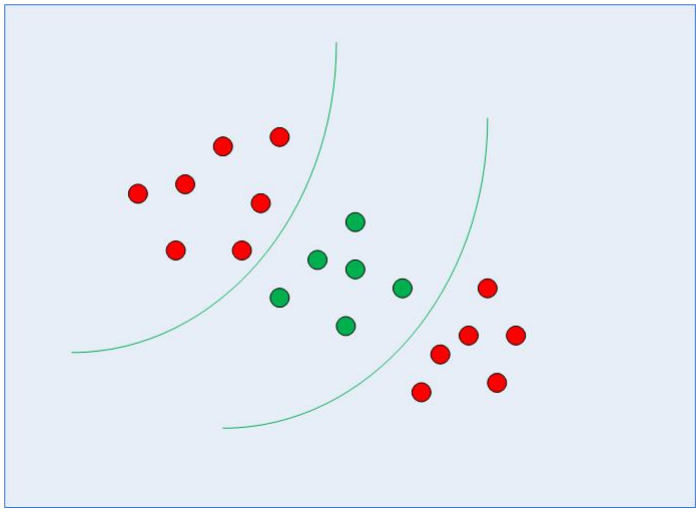

Fig. 3.7 Non-Linearly Separable Data

In case of non-linearly separable data, the simple SVM algorithm cannot be used. Rather, a modified version of SVM, called Kernel SVM, is used. Basically, the kernel SVM projects

the non-linearly separable data lower dimensions to linearly separable data in higher dimensions in such a way that data points belonging to different classes are allocated to different dimensions. As the dataset is considered in higher dimensions, the hyper-plane will not be a line this time; it will be as shown in Fig. 3.8 below

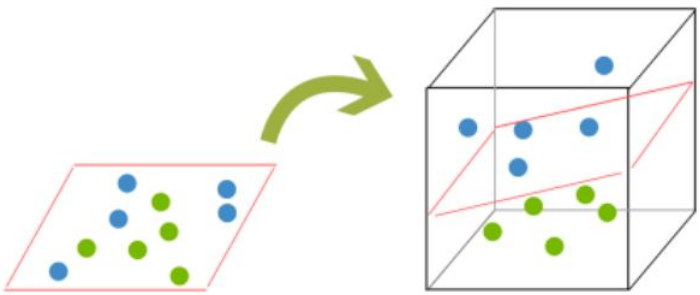

Fig. 3.8 Kernel SVM Classification

There are 3 types of kernel SVM available in the Scikit-learn library of python and these are:

- Polynomial
- Radial basis function (RBF)
- Sigmoid

For our research we have implemented linear and RBF or Gaussian Kernel support vector classifier using Scikit-learn library.

3.5 K-Nearest Neighbors

K-Nearest Neighbors (KNN) algorithm is one of the simplest, easy to understand, versatile and one of the topmost machine learning algorithms. KNN is a non-parametric supervised learning algorithm. Additionally, it is an instance-based learning or a lazy algorithm.When a query to the database is made, the algorithm uses the training instances to spit out an answer. That is why, for KNN the training phase is very fast compared to other classifier algorithms. However, the testing phase becomes slower and costlier, that is in terms of time and memory. KNNs purpose is to use a database in which the data points are separated into several classes

to predict the classification of a new sample point. Informally, this means that for a labeled dataset consisting of training observations (x, y) and the algorithm is used to capture the relationship between x and y. KNN Algorithm is based on feature similarity: How closely out-of-sample features resemble our training set determines how we classify a given data point. An example of KNN classification is shown below in Fig. 3.9:

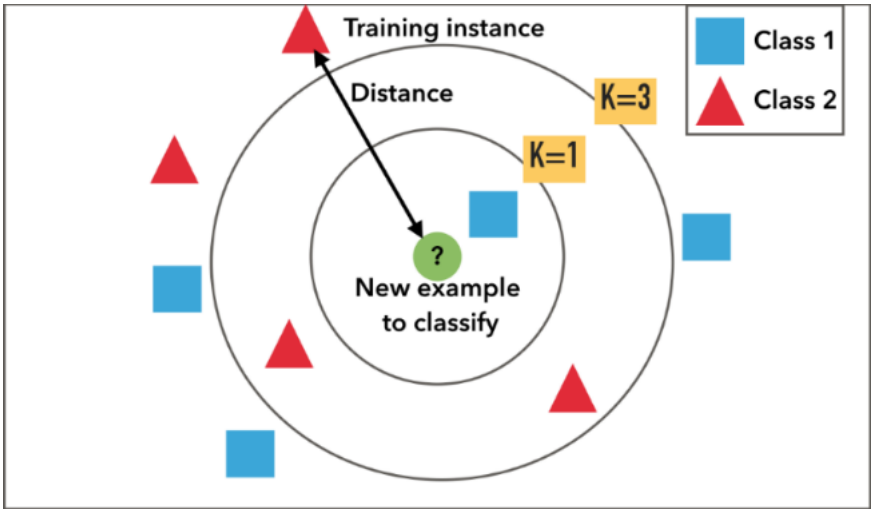

Fig. 3.9 K-Nearest Neighbors Classification

Here, the test sample (inside circle) has to be classified either to the first class of blue squares or to the second class of red triangles. If k = 3 (outside circle) it is assigned to the second class because there are 2 triangles and only 1 square inside the inner circle. If, k = 5 it is assigned to the first class (3 squares vs. 2 triangles outside the outer circle).

In KNN, K is considered as the number of the nearest neighbors. The number of neighbors is the core deciding factor for the classification. The algorithm is known as the nearest neighbor algorithm when K=1. In the classification setting, the K-nearest neighbor algorithm essentially boils down to forming a majority vote between the K most similar instances to a given “unseen” observation. Similarity is defined according to a distance metric between two data points. The “Euclidean distance” is a popular choice. But other measures can be better suited for a particular setting that includes the distance Manhattan, Minkowski and Hamming. The equations are shown in the Fig. 3.10 below:

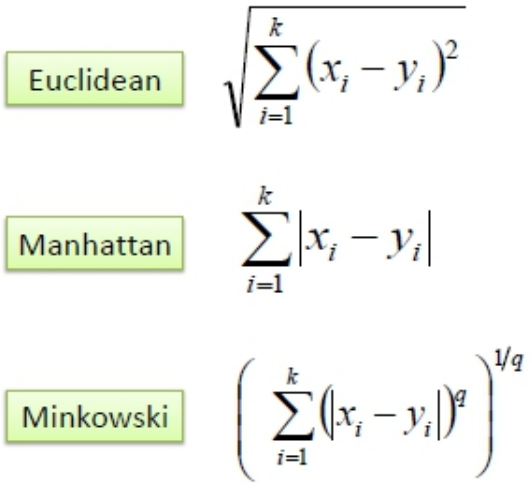

Fig. 3.10 Formula for Distance Metric Calculation

More formally, given a positive integer K, an unseen observation x and a similarity metric d, KNN classifier performs the following three steps:

• It runs through the whole dataset computing d between x and each training observation. We'll call the K points in the training data that are closest to x the set A. Note that K is usually odd to prevent tie situations.

• It then estimates the conditional probability for each class, that is, the fraction of points in A with that given class label.

• Finally, the input x is assigned to the class with the largest probability.

The number of neighbors (K) is a hyper-parameter that needs to be chosen at the time of model building. Research has shown that there is no optimal number of neighbors which suits all kind of data sets. Each dataset is different and needs to fulfill its own requirements. In most cases, it is better to choose it as an odd number if the number of classes is even. When the value of K is small, the region of a given prediction is being restrained. And the classifier is being forced to be "more blind" to the overall distribution. A small value for K provides the most flexible fit, which will have low bias but high variance. Graphically, the decision boundary will be more jagged, which is shown in the Fig. 3.11 below:

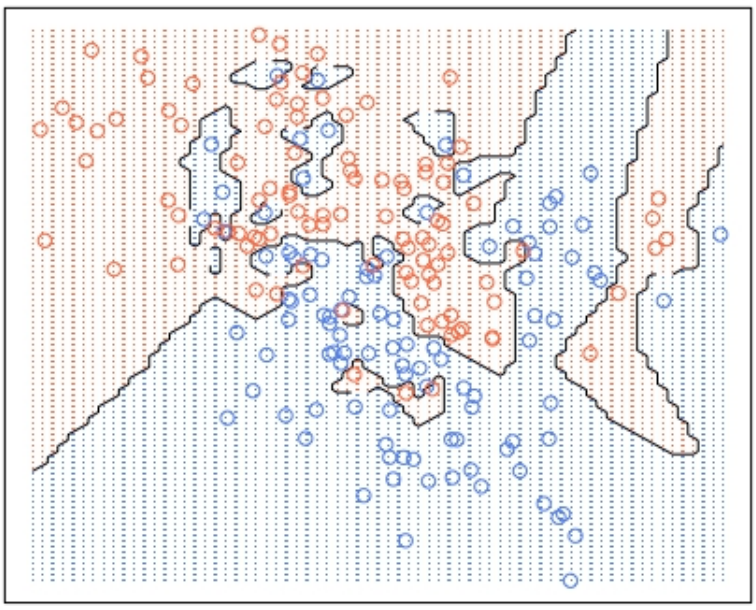

Fig. 3.11 Data Classification When K=1

On the other hand, a higher valued K averages more voters in each prediction. Hence it is more resilient to outliers. Larger values of K will have smoother decision boundaries which mean lower variance but increased bias, shown below in the Fig. 3.12 below:

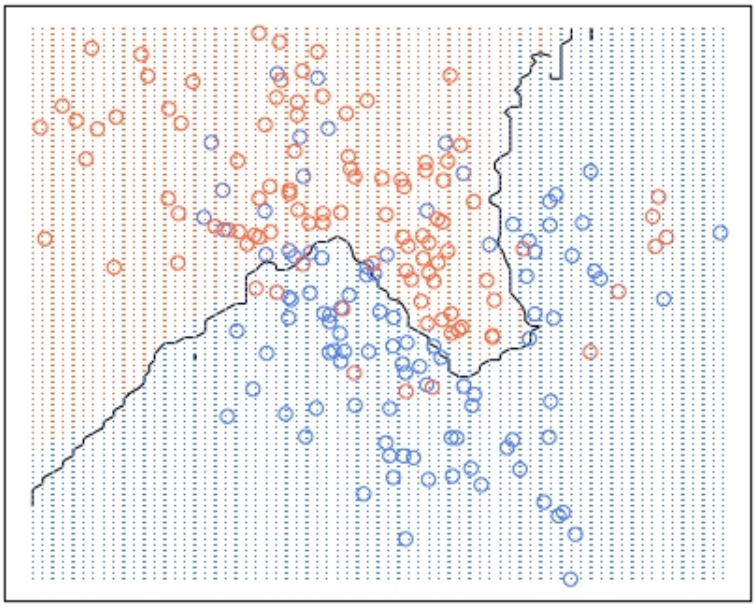

Fig. 3.12 Data Classification When K = 20

However, the training error rate and the validation error rate are two parameters need to

be accessed on different K-value. Following is the curve for the training error rate curve with varying value of K for sample dataset in Fig. 3.13

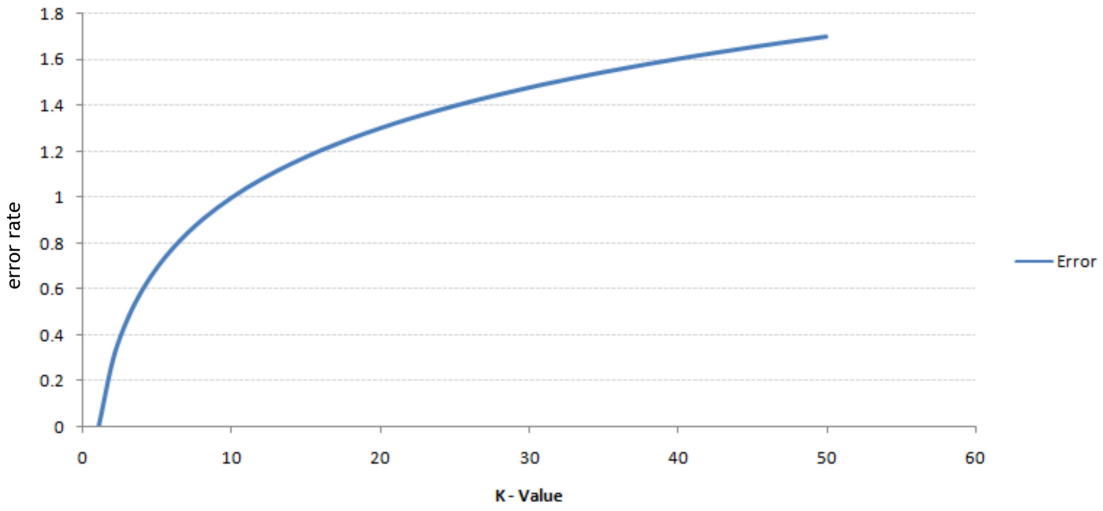

Fig. 3.13 Training Error Curve

Here the error rate at K=1 is always zero for the training sample. This is because the closest point to any training data point is itself. As a result, the prediction is always accurate with K=1. If validation error curve would have been similar, the choice for the value of K would have been 1. Following is the validation error curve with varying value of K in Fig. 3.14:

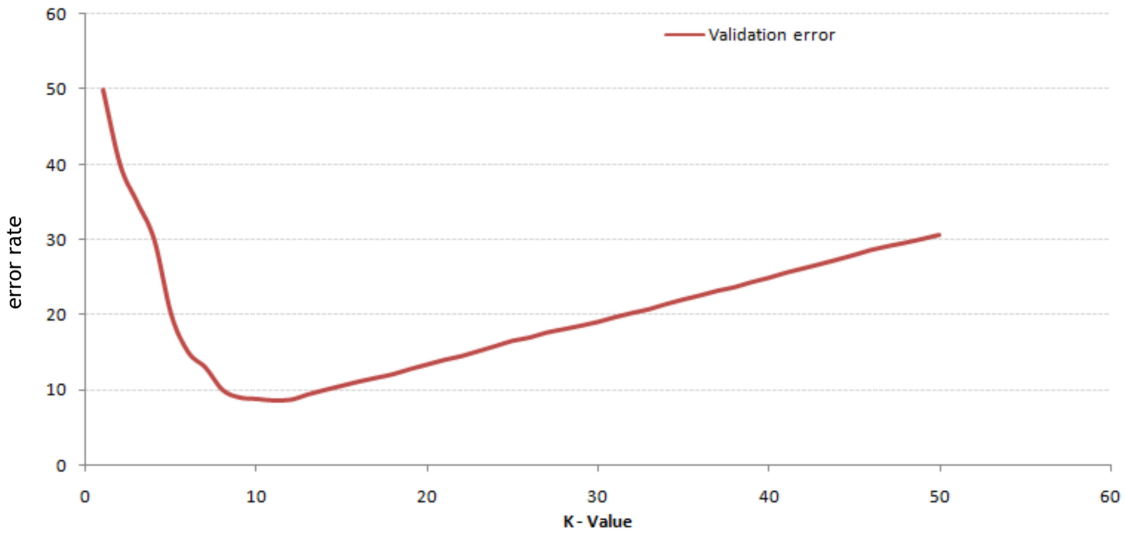

Fig. 3.14 Validation Error Curve

This graph clarifies that at K=1, the boundaries were being over-fitted. So the error rate initially decreases and reaches minima. After the minima point, it then increases with increasing K. To get the optimal value of K, the training and validation can be segregated from the initial dataset. Then by plotting the validation error curve, it is easier to get the optimal value of K. This value of K should be used for all predictions.

3.6 Artificial Neural Network

The use of neural networks in our research is to bring a final decision on the performance of all the previously mentioned algorithms that we have used. Neural networks work in a different way than other regular machine learning algorithms. Hence, these systems do not fall into the machine learning category. Neural networks are therefore an integral part of deep learning due to their mechanism of learning and predicting. Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN) are prominent neural networks in this field. However we have opted to adopt Artificial Neural Network for our design and prediction. CNN are deep neural networks that are primarily used for classifying images, clustering them and performing object recognition within scenes. They are algorithms that can identify faces, individuals, street signs, kidneys, rabbits and many other aspects of visual data. RNN are a powerful set of neural network algorithms especially useful for processing sequential data such as sound, time series (sensor) data or written natural language. A version of recurrent networks was used by Deep Mind in their work playing video games with autonomous agents.

None of the above suit our purpose, hence the selection of Artificial Neural Network. An Artificial Neural Network or ANN is an information processing paradigm that is inspired by the way biological nervous systems, such as the brain, process information. The key element of this paradigm is the novel structure of the information processing system. It is composed of a large number of highly interconnected processing elements called neurons working in unison to solve specific problems. Similar to human beings, artificial neural networks learn by example. An ANN is configured for a specific application, such as pattern recognition or data classification, through a learning process. The data classification characteristic of ANN remains one of the key reasons for us to pursue ANN for our research. Learning in biological systems involves adjustments to the synaptic connections that exist between the neurons. This is true of ANNs as well[12]. Artificial neural networks are one of the main tools used in machine learning. As the "neural" part of their name suggests, they are brain-inspired systems which are intended to replicate the way that we humans learn. Neural networks consist of input and output layers, as well as (in most cases) a hidden layer consisting of units that transform the input into something that the output layer can use. They are excellent tools for finding patterns which are far too complex or numerous for a human programmer to extract and teach the machine to recognize. Neural networks have back propagation. This allows networks to adjust their hidden layers of neurons in situations where the outcome does not match what the creator is hoping for. The different layers of a multi-layer network extract different features until it can recognize what it is looking for. Multi-layers of ANN is shown in Fig. 3.15

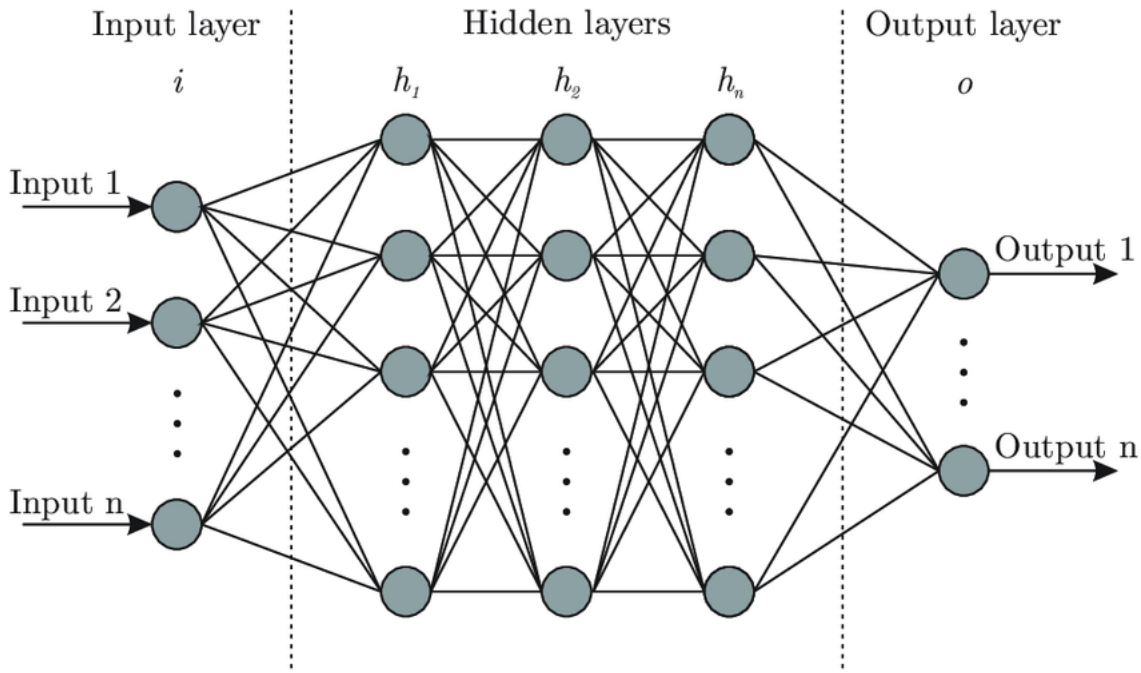

Fig. 3.15 Multi-layers of ANN

This is a field of supervised learning. The whole concept of training and testing emerge due to this. We shower the neural network with numerous features and correct outcomes. This is where it learns. With enough examples of feature-outcome pairs, the calculations and values stored at each neuron and synapse are slowly adjusted. This is the very basic of back propagation. An artificial neural network is given a multitude of examples and then it tries to get the same answer as the example given. When it is wrong, an error is calculated and the values at each neuron and synapse are propagated backwards through the ANN for the next time. This process takes a lot of examples. For real world applications, the number of examples can be in the millions. The number of neurons and layers to be taken is also essential. The deeper the neural network, the better refined the outcome would presumably be. Although it takes a lot of computational power, having more layers is always beneficial as it directly impacts the accuracy of the result. Layers are just sets of neurons. We have an input layer which is the data we provide to the ANN. We have the hidden layers. Lastly, we have the output layer, which is where the finished computations of the network are placed for us to use. A sample of the layers are shown below in fig

Layers themselves are just sets of neurons. Having a single hidden layer does not have any particular disadvantage. However, for better accuracy, addition of new neurons to the single hidden layer will actually just create a linear mapping from each input to the output. In other words, a certain input would always map to a certain output. This leads to zero flexibility and really could only handle inputs learned before. Hence, predictive capabilities

of the network will be greatly underutilized. Deep learning, which is when we have more than one hidden layer, is suitable for classification problems such as ours.

Each neuron has a set of weights that need to be maintained. One weight for each input connection and an additional weight for the bias. We store additional properties for a neuron during training, which will be used as a dictionary to represent each neuron and store properties by names such as ‘weights‘ for the weights.

A network is organized into layers. The input layer is really just a row from our training dataset. The first real layer is the hidden layer. This is followed by the output layer that has one neuron for each class value. We organize layers as arrays of dictionaries and treat the whole network as an array of layers. It is good practice to initialize the network weights to small random numbers such as 0 or 1. We calculate an output from a neural network by propagating an input signal through each layer until the output layer outputs its values. This is called forward-propagation. It is the technique that is needed to generate predictions during training that will need to be corrected, and it is the method required after the network is trained to make predictions on new data Making predictions with a trained neural network is simple and straightforward. Forward-propagating an input pattern to get an output is all that is required to create a prediction. We use the output values themselves directly as the probability of a pattern belonging to each output class. The predictions derived from an artificial neural network is an output that has been thoroughly processed and cross-checked by the network. This gives the outcome greater viability. We have valued our results with artificial neural network on our dataset on a different level simply due to this viability.

Chapter 4

Design and Implementation

Multiple pathways can be taken for a research of this sort. The field of machine learning is vast enough to accommodate a great number of ways this type of prediction work can be done. However, this particular work that we have tried to accomplish has never been done before in the manner we have chosen to do it. As mentioned before in the "Literature Review" section of this paper, we have come across a few works in this field that have similar target as us. However, we have taken a data mining and deep learning approach that had not been done before.

4.1 Review

For our model, we have implemented all the algorithms and data processing codes using the Python programmable language. The Integrated Development Environment (IDE) that we have used is Anaconda Spyder, which is a highly efficient Python favored IDE for data science related programming. Occasionally, we ran our codes on the Jupyter Notebook during our initial data processing phases. Our model consists of all the aforementioned algorithms which has led us to propose a comparison between all of them, based on performance. There are 9 steps that we have followed in our model:

- Dataset collection.
- Data visualization.
- Data pre-processing in the form of data cleaning and feature extraction.
- Data pre-processing in the form of feature selection, feature scaling and dimensionality reduction.

- Data splitting into train and test sets.
- Fitting the algorithm.
- Parameter tuning (only for Artificial Neural Network).
- Testing the accuracy of the model.
- Data post-processing in the form of performance metrics.

A block diagram of our model is given below in Fig. 4.1:

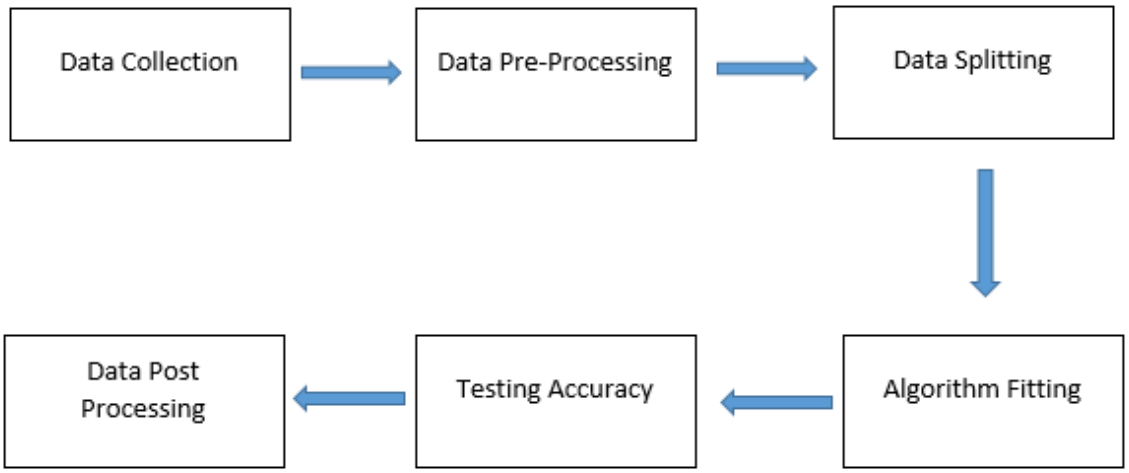

Fig. 4.1 Block Diagram of our Model

4.2 Data Collection

For a research of this sort, it is crucial to have an available dataset to work upon. It is very difficult to find legible and reliable datasets of this sort. It took us a lot of time and effort to find one suitable for us. The dataset that we finally found contained all the accurate features that we really wanted. The features were perfect for a research of this sort. There were a total of 12 columns and around 250,000 rows. The columns were “State Name”, “District Name”, “Crop Year”, “Season”, “Area”, “Rainfall”, “Humidity”, “Temperature”, “Previous Year’s Rainfall”, “Previous Year’s Humidity”, “Previous Year’s Temperature” and “Crop”. 4 of the columns contained data which were in string notation. The rest of the columns contained data which were numerical. The “State Name” column had the names of a number of important states in the country. The “District Name” column had names of districts in those states. The “Crop Year” column had the crop years for the past 19 years. “Season”

contained 4 different seasons. “Area” had area data in meters squared. “Area”, “Rainfall”, “Humidity”, “Temperature”, “Previous Year’s Rainfall”, “Previous Year’s Humidity”, and “Previous Year’s Temperature” all contained data respective of the names just mentioned. Finally the “Crop” column, our most important column, contained 135 different crops which were grown in the places mentioned in the respective columns. All this data is for India, and were taken from the Indian Government Agricultural website upon email request. A plot representation of our data for "Crop" vs "Area" is shown in Fig. 4.2.

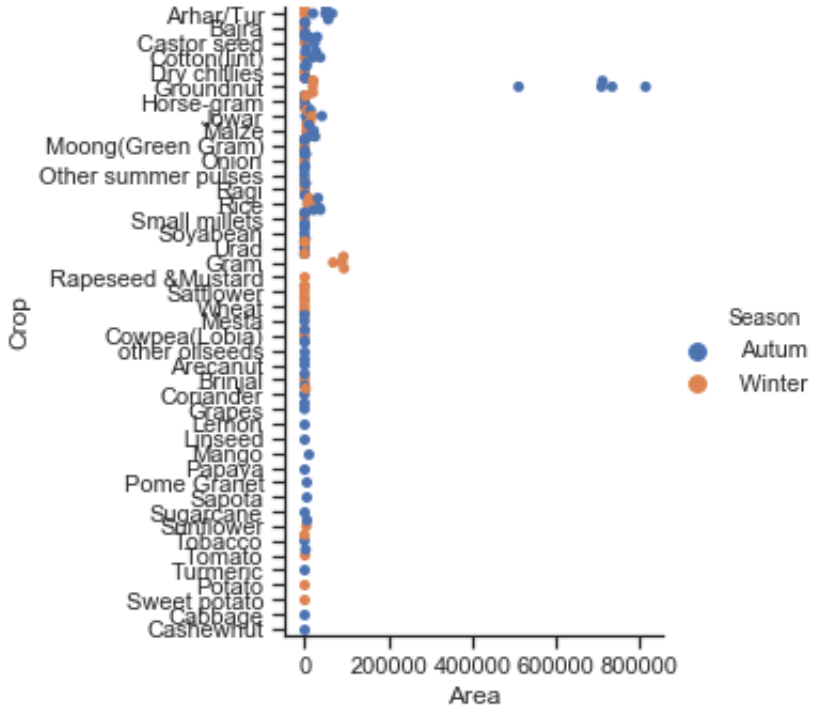

Fig. 4.2 Graphical Representation of Crop vs Area Data for Different Seasons

A research in the field of machine learning, especially deep learning within machine learning, requires heavy usage of computational power. In our research, we had to use Keras and Tensorflow as a back-end engine which actually supports the Scikit-learn library. Scikit-learn is basically run by Keras. Furthermore, as our dataset contained a massive number of rows, algorithms such as the Artificial Neural Network algorithm, required high computational power. A distribution plot representation of sample data is shown in Fig. 4.3.

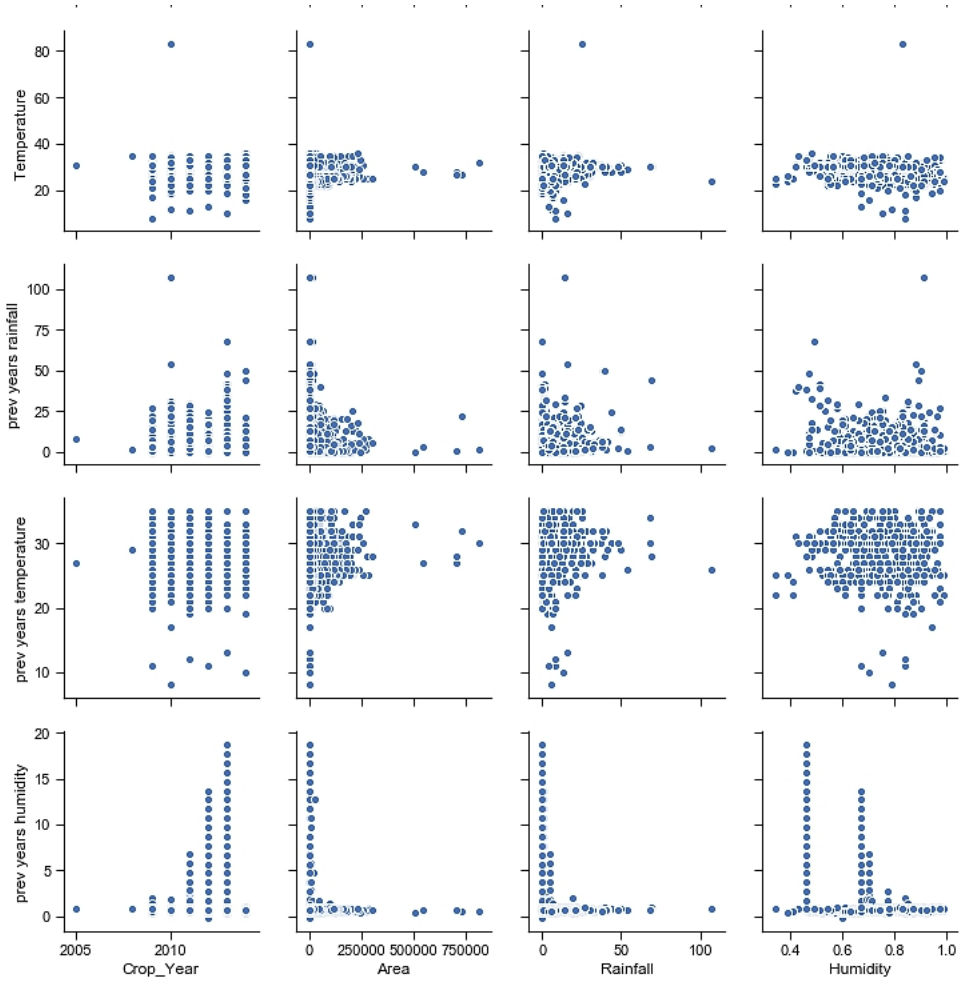

Fig. 4.3 Data Distribution Plot for Some of the Features

Next we split our dataset into a train set and a test set. We then trained our algorithms with the scikit-learn library.

4.3 Data Pre-Processing

One of the primary tasks we completed was to convert all of our string data into numerical data. In order to do this, we converted all the string features into dummy variables. This greatly increased our column number. We then cleaned our data in a very singular pattern. We had a small portion of null values in the production column. Due to the miniature amount, we dropped the fields off of the dataset. This did not affect much, as the amount was minimal. We also performed feature extraction on the dataset. Some of our data were categorical and some were continuous numeric. This type of mixed data always causes problems to

the algorithms. Hence, we performed standard feature scaling on all of the data to bring them into a common scale. Feature scaling is extremely important due to the fact that, most algorithms feature a lot of internal calculation. Additionally, feature selection was done to reduce over fitting issues.

4.4 Data Splitting

Data splitting is the process of splitting the dataset into training and testing data. This process is very useful for any machine learning process as the main idea of machine learning depends on training and testing data and finding the accuracy of the machine given result. In our research, we divided our dataset because we trained our algorithms on the test dataset where the particular crop had its data in there. Here the algorithms were trained using that data. We deduced that data having 1 to be yes, and 0 to be no. The algorithms were trained and we will apply the trained algorithm to our test set and measured the accuracy of the machine. The datasets are usually divided into an 80:20 ratio. However, for our model the dataset was split into both 80:20 and 60:40 ratio. Thus, the 80%/60% of the selected data was chosen as training set and the remaining 20%/40% was test set. There are many built in python tool-kits for splitting data such as, 'pandas', 'keras', 'scikit-learn' etc. although we used "scikit-learn" for the machine learning approaches in this research because of its built-in libraries.

4.5 Algorithm Fitting

The most crucial part of the model was to fit the algorithm with the data. All the algorithms were easily fitted as the programming of this part was comparatively easy. Simple method callings were all that were required. The algorithms, upon being implemented, processed all the data using all the internal calculations. Data frames were created and could be viewed in the variable explorer. Because of the fact that the data had been split into training and testing datasets, the algorithm could start the core process: learning. The machine learned from the train set. This learning was to be used later on while predicting from the test set. Fitting is similar to training. For example, let us assume that we have measured the production for a group of crops and decided that your model would be a normal distribution. Then determining the mean and variance of the normal distribution that best explains our observed data is called fitting: we are determining the parameters' mean $\mu\mu$ and variance $\sigma\sigma$. Suppose we have an algorithm that estimates $\mu\mu$ and $\sigma\sigma$ given our data. We can ask our algorithm to run "n" iterations where each iteration takes the same amount of time but

more iterations yield slightly more accurate estimates of the parameters $\mu\mu$ and $\sigma\sigma$. “n” is a value we supply that may influence the estimates and is called a hyper-parameter.

4.6 Testing Accuracy

To test the accuracy, we implemented different methods on different algorithms based on requirement. Some were direct accuracy-check method calls from scikit-learn libraries. While in some other algorithms, we implemented manual accuracy checks, again based on the algorithm itself. In Random Forest of instance, mean was calculated. In Artificial Neural Network, despite calling an accuracy-check method, all the accuracy from all the epochs were taken in for mean value of accuracy. The accuracy check is crucial in understanding the viability of the algorithms and also the research itself. A very low accuracy in all the algorithms would mean the entire research was a dead end. It would mean this method altogether is not viable for this research. A low accuracy in a few algorithms and a high accuracy in the others would mean the ones with the low accuracy are not efficient in this model, but the others are. We would have discarded the low accuracy yielding algorithms. However, in our case, all the algorithms yielded a very high accuracy. Although this issue did cause us a few problems later on when comparing the accuracy amongst all the algorithms, the fact that the accuracy is high on all, was a strong point in determining the methods to be viable in this field of research.An Effective Model is a model which basically predicts the testing data most accurately as compared to other models and hence, can be deployed successfully. Testing accuracy of k-NN is shown in the Fig 4.4

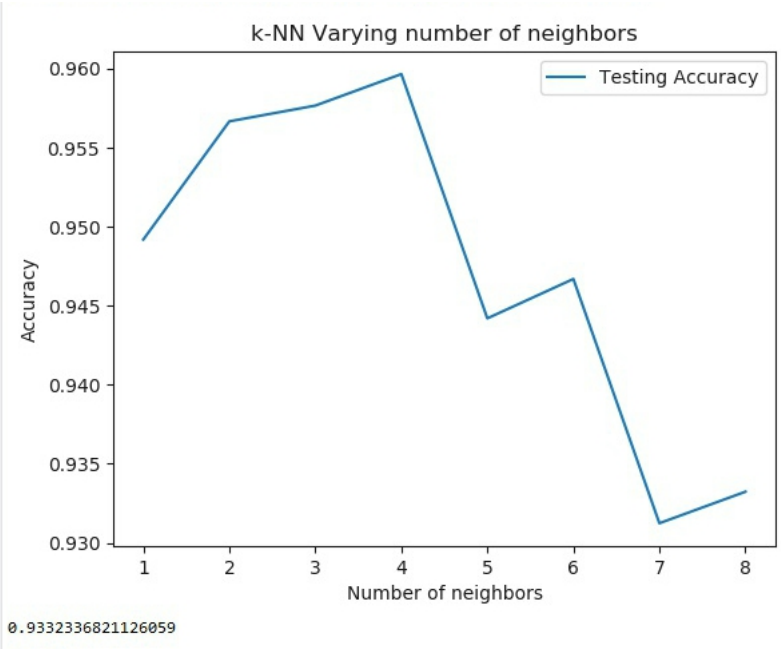

Fig. 4.4 Testing Accuracy for k-Nearest Neighbors for Specific Crop Possibility

4.7 Data Post-Processing

After all the accuracy have been taken into account, a few other data processes can still be implemented. This part of the model is not necessary for the primary target of the research, but we still used it for certain confirmation purposes. We implemented a method which would create a confusion matrix. A confusion matrix is a technique for summarizing the performance of a classification algorithm. Classification accuracy alone can be misleading if we have an unequal number of observations in each class or if we have more than two classes in our dataset. Calculating a confusion matrix can give us a better idea of what our classification model is getting right and what types of errors it is making. Fig. 4.5 shows the confusion matrix we got by implementing Logistic Regression for crop prediction.

```
[[  0   0   0 ...   0   0   0]
 [  0  33   0 ...   0   0   0]
 [  0   0  57 ...   0   0   0]
 ...
 [  0   0   0 ... 152   0   0]
 [  2   0   0 ...   0  54   0]
 [  0   5   0 ...   0   0   0]]
```

Fig. 4.5 Confusion Matrix for Crop Prediction

The methods to reduce any over fitting issues that were implemented earlier in the data pre-processing stage, can even be implemented here instead the other stage. However, it was extremely necessary for us to use those in the model, any form of over fitting or under fitting would actually render the whole model lack of any real use.

4.8 Artificial Neural Network (ANN)

We followed standard procedures for all the algorithms apart from the Artificial Neural Network (ANN). Starting from the data pre-processing, to algorithm fitting, to accuracy checking, and finally predicting the desired outcome, the methods and codes used nearly remained the same. The only major variation was in the part where we called the algorithm fitting functions. However, the deep learning model that we created for the ANN, was different to the other algorithms to a certain extent. We created the hidden layers and perceptions manually. Even after the predictions were being generated, we performed further checks in the form of k-Fold Cross Validation and GridSearchCV to ensure the highest viability of the accuracy obtained. The first step was to calculate the activation of one neuron given an input. The input was a row from our training dataset, as in the case of the hidden layer. It might also be the outputs from each neuron in the hidden layer, in the case of the output layer. Neuron activation was calculated as the weighted sum of the inputs. Much like linear regression.

$$activation = sum(weight\_i * input\_i) + bias \tag{4.1}$$

Where weight was a network weight, input was an input, “i” was the index of a weight or an input and bias was a special weight that had no input to multiply with.

Once a neuron was activated, we needed to transfer the activation to see what the neuron output actually was. Different transfer functions could be used. It was traditional to use the sigmoid activation function, but we could also use the tan h (hyperbolic tangent) function to transfer outputs. However, due to the recent popularity and increase in efficiency, the rectifier transfer function had been used by us in our deep learning model. The sigmoid activation function looks like an S shape; it is also called the logistic function. It can take any input value and produce a number between 0 and 1 on an S-curve. It is also a function of which we can easily calculate the derivative (slope) that we use later when back propagating error. We could transfer an activation function using the sigmoid function as follows.

$$output = 1/(1 + e^{-activation}) \tag{4.2}$$

Forward propagating an input was straightforward. We worked through each layer of our network calculating the outputs for each neuron. All of the outputs from one layer become inputs to the neurons on the next layer. The back propagation algorithm was named for the way in which weights are trained. Error was calculated between the expected outputs and the outputs forward propagated from the network. These errors were then propagated backward through the network from the output layer to the hidden layer, assigning blame for the error and updating weights as they go. Given an output value from a neuron, we needed to calculate its slope.

$$derivative = output * (1.0 - output) \tag{4.3}$$

The first step was to calculate the error for each output neuron, that gave us our error signal (input) to propagate backwards through the network. The error for a given neuron was calculated as follows.

$$error = (expected - output) * transfer\_derivative(output) \tag{4.4}$$

Where "expected" was the expected output value for the neuron, "output" was the output value for the neuron and transfer_derivative() calculated the slope of the neuron's output value, as shown above. This error calculation was used for neurons in the output layer. The expected value was the class value itself. In the hidden layer, things are a little more complicated. The error signal for a neuron in the hidden layer was calculated as the weighted error of each neuron in the output layer. The error traveled back along the weights of the output layer to the neurons in the hidden layer. The back-propagated error signal was accumulated and then used to determine the error for the neuron in the hidden layer, as follows.

$$error = (weight\_k * error\_j) * transfer\_derivative(output) \tag{4.5}$$

Where "error_j" was the error signal from the "j" th neuron in the output layer, "weight_k" was the weight that connects the "k"th neuron to the current neuron and output was the output for the current neuron. The network was trained using stochastic gradient descent.

This involves multiple iterations of exposing a training dataset to the network and for each row of data forward propagating the inputs, back propagating the error and updating the network weights. Once errors were calculated for each neuron in the network via the back propagation method above, they could be used to update weights. Network weights were updated as follows.

$$weight = weight + learning\_rate * error * input \tag{4.6}$$

Where “weight” was a given weight, “learning_rate” was a parameter that we specify, “error” was the error calculated by the back propagation procedure for the neuron and “input” was the input value that caused the error. The same procedure might have been used for updating the bias weight, except there was no input term, or input was the fixed value of 1.0. Learning rate controls how much to change the weight to correct for the error. For example, a value of 0.1 will update the weight 10% of the amount that it possibly could be updated. Small learning rates are preferred that cause slower learning over a large number of training iterations. This increases the likelihood of the network finding a good set of weights across all layers rather than the fastest set of weights that minimize error (called premature convergence). As mentioned above, the network was updated using stochastic gradient descent. This involved first looping for a fixed number of epochs and within each epoch updating the network for each row in the training dataset.

Because updates were made for each training pattern, this type of learning is called online learning. If errors were accumulated across an epoch before updating the weights, this would be called batch learning or batch gradient descent. Once all the aforementioned steps were completed, the network became eligible for training. The rest of the procedure followed a similar structure to the rest of the algorithms, as discussed above.

Chapter 5

Result Analysis

The core aim of our research was to establish a model that will efficiently predict a particular crop based on a set of features. During the course of this research, we have also successfully created a model that can predict the possibility of a particular crop to be able to be grown, given a set of features. In order to do this, as mentioned above, we have implemented 6 different machine learning algorithms. Amongst them, The SVM was done using 3 different aspects, and the KNN was done using 2. This ensured us the ability to bring a comparison between varieties of different algorithms. Our target was to establish the best performing algorithm for this field of work, based on our data.

5.1 Accuracy Analysis

As mentioned before, we have extracted accuracy from 9 different processes. Only the ANN was run on one instance only. Apart from that, all the other algorithms were tested on a variety of instances of the same dataset. We took 5 samples of our primary dataset. The samples had an increasing number of rows starting from 2000 and ending at 11000. We ran each algorithm on these samples in 2 separate train test splits. Initially, we used the 60%-40% train test ratio. Eventually, we changed that to 80%:20%. Both gave us fair results, but the latter gave us a better accuracy, which led us to finalize the model on that. Two separate outcomes were extracted from our algorithms with slight tweaking. We could both predict a particular crop and the percentage chance of a specific crop to be able to be grown, given a particular set of features. However, the former method fetched bad accuracy levels, which we found highly unconvincing. The highest accuracy we got was from the KNN (K=optimal) algorithm, which was 81.3%, on a set of 11000 features. In similar features, but with a specific crop as the dependent variable, all the algorithms gave strong accuracy levels. The lowest accuracy was given by Random Forest Classifier which was still 92.30%. The highest

accuracy was again given by the KNN (K=optimal) algorithm. The ANN was tested only once with 11000 features, and only for the specific crop model. We trained and tested the network up to 1000 epochs. The accuracy that we obtained was 96.95%. All the accuracy levels in this part of the thesis indicated to a strong viability of our research in this field. In the (Table. 5.1) and (Table. 5.2) below we have shown the comparison among all the classifiers we have implemented for specific crop possibility prediction and crop prediction respectively:

Table 5.1 Accuracy comparison for specific crop possibility

data	GNB	SVM	Linear SVM	RBF SVM	KNN (k=1)	KNN (k= optimal)	LR	RF	ANN
2000	80.61	95.07	92.97	94.38	90.44	95.50	95.22	95.07	
4000	84.26	94.95	96.35	94.53	93.26	95.65	96.07	95.16	
6000	89.04	97.28	97.09	97.84	96.16	96.44	95.78	95.78	
8000	85.39	95.27	95.59	95.74	95.59	96.81	95.31	93.71	
11000	95.5	97.0	97.2	96.23	96.38	97.70	97.38	92.30	96.95

Table 5.2 Accuracy comparison for crop prediction

data	GNB	SVM	Linear SVM	RBF SVM	KNN (k=1)	KNN (k= optimal)	LR	RF
2000	70.18	70.50	65.44	67.13	65.44	66.01	62.58	39.04
4000	70.88	49.92	49.64	58.34	38.84	48.80	57.50	44.37
6000	55.97	63.12	67.50	62.92	54.02	56.46	52.52	45.41
8000	58.79	63.52	58.90	64.79	57.33	60.0	43.73	45.84
11000	56.2	63.7	56.8	68.8	61.38	81.3	66.46	55.0

In Fig. 5.1 we have shown the graphical representation of accuracy comparison among all classifiers for crop prediction

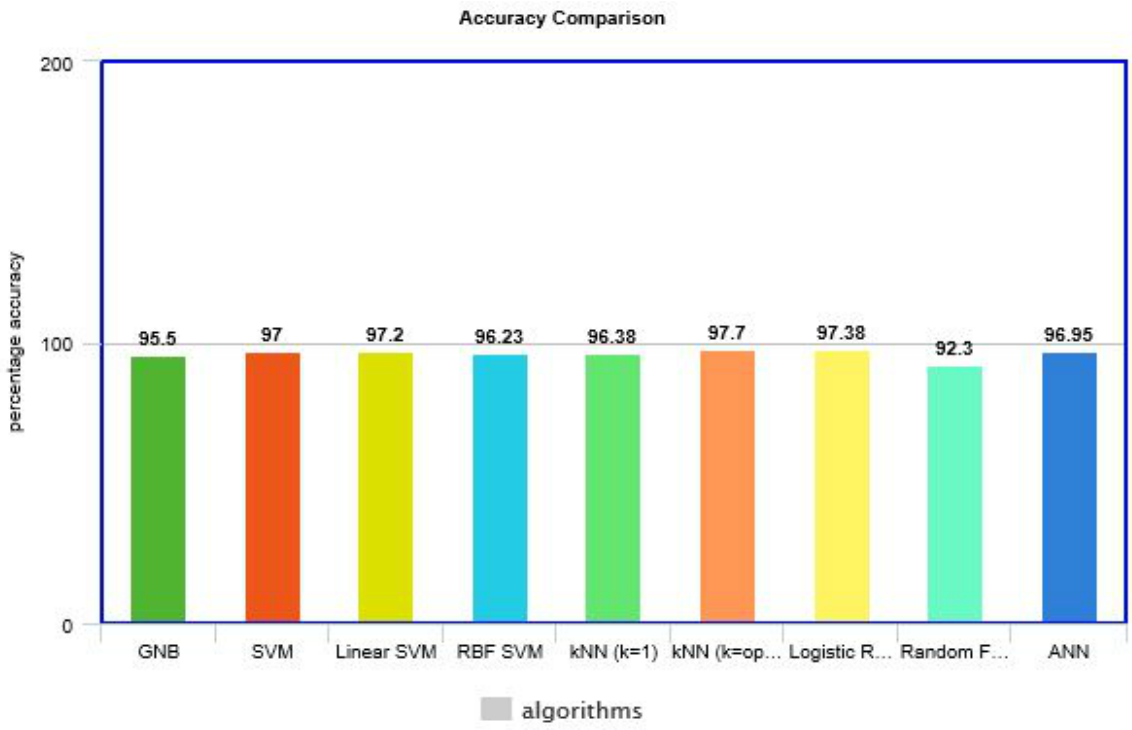

Fig. 5.1 Accuracy Comparison Among All Classifiers for Specific Crop Possibility

Additionally, in Fig. 5.2 we have shown the graphical representation of accuracy comparison among all classifiers for specific crop's possibility among all classifiers.

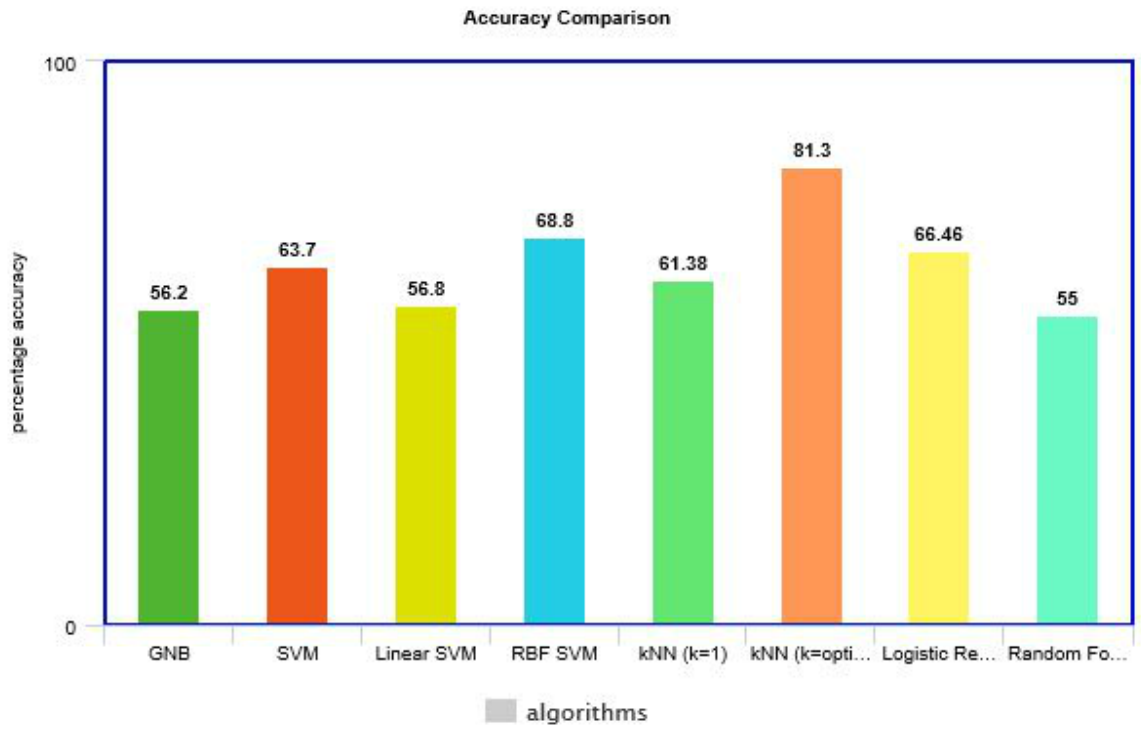

Fig. 5.2 Accuracy Comparison Among All Classifiers for Crop Prediction

From the graphs we can see that the accuracy highly vary for crop prediction. However, for specific crop possibility prediction the accuracy range is quite stable for all the algorithms.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

We believe this model can play a very essential part in today's world. Agriculture is a fundamental aspect of modern civilization. With increasing world hunger and economy breakdown, the proper selection of crop emerges as a massive factor in this. Our proposed model can predict the proper crop for a particular piece of land in a way that is very efficient. We have implemented 6 different types of machine learning and deep learning algorithms in this research. All the accuracy of the models were carefully obtained through various different methods, and compared with each other. Using multiple algorithms helped to understand which algorithm is more suitable for this system. The crops can be predicted based on a very suitable set of features included in the dataset used. From this research work, we found that, the cleaner the data, the better the accuracy of the result. The entire length of this research was very enjoyable, as we were able to work in the field of machine learning and deep learning. Some of the python library usage, algorithm fitting, and accuracy checking methods were very interesting in practicality. We trust all our algorithms and research work to efficiently work on any platform and any new type of data. The predictions made were solid and robust. Such strength in the model delights us, and we hope this keeps working over the years without issues.

6.2 Future Work

In the future, we hope that this model would be implemented with much more efficient dataset for a specific piece of land containing information such as different soil property, soil pH, different mineral percentage etc. So that no agricultural plot be wasted by harvesting a

less efficient crop. We want this model to be used worldwide for the further development of the agricultural sector. This can be a field of interest for both researchers and entrepreneurs. In this research, we have developed a model that will predict suitable crop or predict the viability of a specific crop for a particular plot of land. Our plan is to build on this research and improve this model; mostly by adding further algorithms based on other aspects of machine learning. For future work, we also want to build a platform for all the farmers who will be using this model, to share the predictions on their land with other farmers all over the land. This will let a farmer in one country to know the prospect of farming in another part of the world; down to a specific geographical unit. The accuracy values we obtained in the specific crop prediction part of our research was very poor to our target standards. In the future, we hope to implement ANN for crop prediction and check its viability on this.

In addition to that, we desire to take this model into the mobile phone platform. Android and iOS applications will be a part of it. Nowadays, even some the poorest farmers are seen to be using such devices. A mobile application can be of help to them as well. One of our most ambitious goals is to improve our model so that it can give a string of crop sequencing predictions. This will enable us to derive predictions for not just the next year, but of multiple years ahead. This time we have only implemented supervised learning. In the future, we plan to use unsupervised and reinforced learning as well. That will give a new dynamic to our research. We are looking for a better world in a sense that we hope our model to be able to successfully raise the standard of farming and agriculture. We hope, whatever shapes this model take, it will stay user friendly and properly welcomed by the potential users.

References

[1] Aggarwal, C. C. (2014). *Data classification: algorithms and applications*. CRC Press.

[2] Breiman, L. (2001). Random forests. *Machine learning*, 45(1):5–32.

[3] Freund, Y., Schapire, R. E., et al. (1996). Experiments with a new boosting algorithm. In *Icml*, volume 96, pages 148–156. Citeseer.

[4] Hsu, C.-W., Chang, C.-C., Lin, C.-J., et al. (2003). A practical guide to support vector classification.

[5] Jain, N., Kumar, A., Garud, S., Pradhan, V., and Kulkarni, P. (2017). Crop selection method based on various environmental factors using machine learning.

[6] Kaur, K. (2016). Machine learning: applications in indian agriculture. *Int. J. Adv. Res. Comput. Commun. Eng.(IJARCCE)*, 5(4).

[7] Kesavaraj, G. and Sukumaran, S. (2013). A study on classification techniques in data mining. In *Computing, Communications and Networking Technologies (ICCCNT), 2013 Fourth International Conference on*, pages 1–7. IEEE.

[8] Kim, S.-B., Han, K.-S., Rim, H.-C., and Myaeng, S. H. (2006). Some effective techniques for naive bayes text classification. *IEEE transactions on knowledge and data engineering*, 18(11):1457–1466.

[9] Kolesnikova, A., Song, C.-H., and Lee, W. D. (2009). Applying uchooboost algorithm in precision agriculture. In *Proceedings of the International Conference on Advances in Computing, Communication and Control*, pages 30–34. ACM.

[10] Kotsiantis, S. B., Zaharakis, I., and Pintelas, P. (2007). Supervised machine learning: A review of classification techniques. *Emerging artificial intelligence applications in computer engineering*, 160:3–24.

[11] Kumar, R., Singh, M., Kumar, P., and Singh, J. (2015). Crop selection method to maximize crop yield rate using machine learning technique. In *Smart Technologies and Management for Computing, Communication, Controls, Energy and Materials (ICSTM), 2015 International Conference on*, pages 138–145. IEEE.

[12] Maind, S. B., Wankar, P., et al. (2014). Research paper on basic of artificial neural network. *International Journal on Recent and Innovation Trends in Computing and Communication*, 2(1):96–100.

[13] Michalski, R. S., Carbonell, J. G., and Mitchell, T. M. (2013). *Machine learning: An artificial intelligence approach*. Springer Science & Business Media.

[14] Osman, J., Inglada, J., and Dejoux, J.-F. (2015). Assessment of a markov logic model of crop rotations for early crop mapping. *Computers and Electronics in Agriculture*, 113:234–243.

[15] Rush, S. (2001). Logistic regression: The standard method of analysis in medical research. Technical report, Technical Report Mathematics.

[16] Su, Y.-x., Xu, H., and Yan, L.-j. (2017). Support vector machine-based open crop model (sbocm): Case of rice production in china. *Saudi journal of biological sciences*, 24(3):537–547.

[17] Sujjaviriyasup, T. and Pitiruek, K. (2013). Agricultural product fore-casting using machine learning approach. *Int. Journal of Math. Analysis*, 7(38):1869–1875.