ABSTRACT

Educational Data Mining(EDM) is an emerging discipline, concerned with developing methods for exploring the unique types of data that come from educational settings, and using those methods to better understand students, and the settings which they learning. New computer-supported interactive learning methods and tools, intelligent tutoring systems, simulations, games have opened up opportunities to collect and analyze student data, to discover patterns and trends in those data, and to make new discoveries and test hypotheses about how students learn. Data collected from online learning systems can be aggregated over large numbers of students and can contain many variables that data mining algorithms can explore for model building. Focus is upon on developing new tools and algorithms for discovering data patterns by applying methods and techniques from statistics, machine learning, and data mining to analyze data collected during teaching and learning. Similarly in this project the analysis has be done based on the employee and alumni data collected from various sources and advanced algorithms have been used to draw patterns and predict suited career to a computer science undergraduate based on his abilities, interests and opportunities. As students are going through their academics and pursuing their interested courses, it is very important for them to assess their capabilities and identify their interests so that they will get to know in which career area their interests and capabilities are going to put them in. This will help them in improving their performance and motivating their interests so that they will be directed towards their targeted career and get settled in that. Also recruiters while recruiting the candidates after assessing them in all different aspects, these kind of career recommender systems help them in deciding in which job role the candidate should be kept in based on his/her performance and other evaluations. This paper mainly concentrates on the career area prediction of computer science domain candidates.

CHAPTER - 1

1.1 INTRODUCTION

Competition today is heavily multiplying day by day. Especially it is too heavy in present day’s technical world. So as to compete and reach the goal students need to be planned and organized from initial stages of their education. So, it is very important to constantly evaluate their performance, identify their interests and evaluate how close they are to their goal and assess whether they are in the right path that directs towards their targeted. This helps them in improving themselves, motivating themselves to a better career path if their capabilities are not up to the mark to reach their goal and pre evaluate them-selves before going to the career peek point.

Not only that recruiters while recruiting people into their companies evaluate candidates on different parameters and draw a final conclusion to select an employee or not and if selected, finds a best suited role and career area for him. There are many types of roles like Database administrator, Business Process Analyst, Developer, Testing Manager, Network Manager, Data scientist and so on. All these roles require some pre-requisite knowledge in them to be placed in them. So, recruiters analyze these skills, talents and interests and place the candidate in the right job role suited for them. These kind of prediction systems make their recruitment tasks very easy because as the inputs are given, recommendation is done based on inputs.

Already these types of various career recommendation systems and job role recommendation, prediction systems are being used in various private performance evaluation portals like Co-Cubes, AMCAT. They only take factors like technical abilities and psychometry of students into consideration. These portals asses the students technically and suggest the students and companies job roles suited on their performance. But here various factors including abilities of students in sports, academics and their hobbies, interests, competitions, skills and knowledge are also taken into consideration. Considering all the factors the total number of parameters that were taken into consideration as inputs are 39. And the final job roles are fixed to15in number. As the input parameters and final classes of output are large in number typical programming and normal algorithms cannot give the best possible output classification and prediction. So advanced machine learning algorithms like SVM, Random Forest decision tree, One Hot encoding, XG boost are used.

Machine Learning is a technique where the machines are trained in such a way that it gains the ability to respond to a particular input or scenario based on the previous inputs it has learnt. Simply it the giving computers the ability to learn by using statistical techniques. Machine learning helps the computers to act without explicitly being programmed. This aims at reducing the human intervention in the machine depend-able problems and scenarios. This helps in solving very complex tasks and problems very easily and without involving much human labor. Various applications of machine learning include NLP, classification, prediction, image recognition, medical diagnosis, algorithm building, self-driving cars and much more. In this paper classification and prediction are being done. Let us see what is classification and prediction. Majority of problems in machine learning can be solved using supervised and unsupervised learning. If the final class labels are previously know and all the other data items are to be assigned with one of the available class labels, then it is call supervised. And if the final output classes and sets are not known and it is done by identifying the similarity between data point and their characteristics and finally they are made into groups based on these characteristics then it is called un-supervised. Classification falls under supervised. Input parameters are given and based on their properties a predefined class label is assigned. There are other alternatives like clustering and regression. Based on the type of problem the apt model is chosen.

However here algorithms like SVM, One Hot encoding, Decision tree and XG-boost are used. After training and testing the data with these we take into consideration the most accurate results given algorithm for our further processing. So, initial task done is predicting the output using all algorithms proposed above and later analyzing the results and there on continued with the most accurate algorithm. So finally, this paper deals with various advanced machine learning algorithms that involves classification and prediction and are used to improve the accuracy for better prediction, reliability and analyzing these algorithms performance

1.2 EDUCATIONAL DATAMINING

Educational data mining (EDM) describes a research field concerned with the application of data mining, machine learning and statistics to information generated from educational settings like universities and intelligent tutoring systems. At a high level, the field seeks to develop and improve methods for exploring this data, which often has multiple levels of meaningful hierarchy, in order to discover new insights about how people learn in the context of such settings. In doing so, EDM has contributed to theories of learning investigated by researchers in educational psychology and the learning sciences. The field is closely tied to that of learning analytics, and the two have been compared and contrasted.

Educational data mining refers to techniques, tools, and research designed for automatically extracting meaning from large repositories of data generated by correlated to people's learning activities in educational settings. Quite often, this data is extensive, fine-grained, and precise. For example, several learning management systems (LMSs) track information such as when each student accessed each learning object, how many times they access edit, and how many minutes the learning object was displayed on the user's computer screen. As another example, intelligent tutoring systems record data every time a learner submits a solution to a problem; they may collect the time of the submission, whether or not the solution matches the expected solution, the amount of time that has passed since the last submission, the order in which solution components were entered into the interface, etc. The precision of this data is such that even a fairly short session with a computer-based learning environment may produce a large amount of process data for analysis.

In other cases, the data is less fine-grained. For example, a student's university transcript may contain a temporally ordered list of courses taken by the student, the grade that the student earned in each course, and when the student selected or changed his or her academic major. EDM leverages both types of data to discover meaningful information about different types of learners and how they learn, the structure of domain knowledge, and the effect of instructional strategies embedded within various learning environments. These analyses provide new information that would be difficult to discern by looking at the raw data. For example, analyzing data from an LMS may reveal a relationship between the learning objects that a student accessed during the course and their final course goal. Similarly analyzing student transcript data may reveal a relationship between a student's grade in a course and their decision to change their academic major. Such information provides insight into the design of learning environments, which allows students, teachers, school administrators, and educational policy makers to make informed decisions about how to interact with, provide, and manage educational resources.

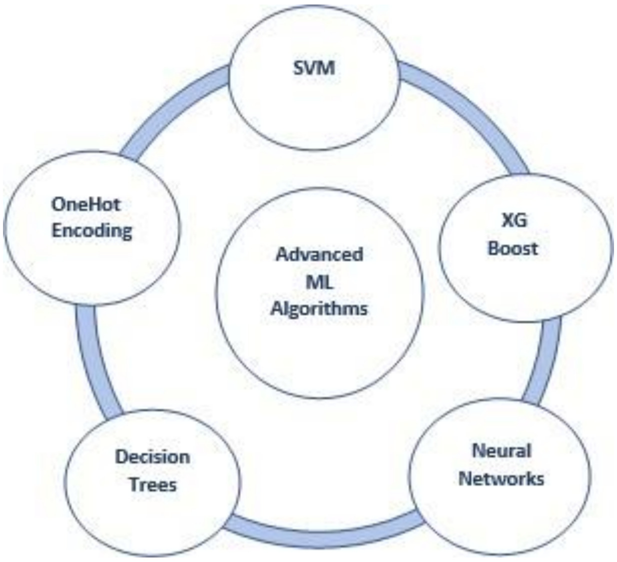


Fig 1: Overview of various Advanced Machine Learning Algorithms Used.

1.3 ARTIFICIAL INTELLIGENCE

AI (artificial intelligence) is the simulation of human intelligence processes by machines, especially computer systems. These processes include learning (the acquisition of information and rules for using the information), reasoning (using the rules to reach approximate or definite conclusions) and self-correction. Particular applications of AI include expert systems, speech recognition and machine vision.

AI was coined by John McCarthy, an American computer scientist, in 1956 at The Dartmouth Conference where the discipline was born. Today, it is an umbrella term that encompasses everything from robotic process automation to actual robotics. It has gained prominence recently due, in part, to big data, or the increase in speed, size and variety of data businesses are now collecting. AI can perform tasks such as identifying patterns in the data more efficiently than humans, enabling businesses to gain more insight out of their data.

It is a branch of computer science that aims to create intelligent machines. It has become an essential part of the technology industry.

Research associated with artificial intelligence is highly technical and specialized. The core problems of artificial intelligence include programming computers for certain traits such as:

- Knowledge.
- Reasoning.
- Problem solving.
- Perception.
- Prediction.
- Learning.
- Planning.
- Ability to manipulate and move objects.

Knowledge engineering is a core part of AI research. Machines can often act and react like humans only if they have abundant information relating to the world. Artificial intelligence must have access to objects, categories, properties and relations between all of them to implement knowledge engineering. Initiating common sense, reasoning and problem-solving power in machines is a difficult and tedious task.

Machine learning is also a core part of AI. Learning without any kind of supervision requires an ability to identify patterns in streams of inputs, whereas learning with adequate supervision involves classification and numerical regressions. Classification determines the category an object belongs to and regression deals with obtaining a set of numerical input or output examples, there by discovering functions enabling the generation of suitable outputs from respective inputs. Mathematical analysis of machine learning algorithms and their performance is a well-defined branch of theoretical computer science often referred to as computational learning theory.

Machine perception deals with the capability to use sensory inputs to deduce the different aspects of the world, while computer vision is the power to analyze visual inputs with a few sub-problems such as facial, object and gesture recognition. Robotics is also a major field related to AI. Robots require intelligence to handle tasks such as object manipulation and navigation, along with sub-problems of localization, motion planning and mapping.

1.4 MACHINE LEARNING

Machine learning is a subfield of artificial intelligence (AI). The goal of machine learning generally is to understand the structure of data and fit that data into models that can be understood and utilized by people. Although machine learning is a field within computer science, it differs from traditional computational approaches. In traditional computing, algorithms are sets of explicitly programmed instructions used by computers to calculate or problem solve. Machine learning algorithms instead allow for computers to train on data inputs and use statistical analysis in order to output values that fall within a specific range. Because of this, machine learning facilitates computers in building models from sample data in order to automate decision-making processes based on data inputs.

Any technology user today has benefitted from machine learning. Facial recognition technology allows social media platforms to help users tag and share photos of friends. Optical character recognition (OCR) technology converts images of text into movable type. Recommendation engines, powered by machine learning, suggest what movies or televisions how to watch next based on user preferences. Self-driving cars that rely on machine learning to navigate may soon be available to consumers. Machine learning is a continuously developing field. Because of this, there are some considerations to keep in mind as you work with machine learning methodologies, or analyze the impact of machine learning processes.

Here in this thesis, we are providing basic info of the common machine learning methods of supervised and unsupervised learning, and common algorithmic approaches in machine learning, including the k-nearest neighbor algorithm, decision tree learning, and deep learning.

1.4.1 SUPERVISEDLEARNING

In machine learning, tasks are generally classified into broad categories. These categories are based on how learning is received or how feedback on the learning is given to the system developed. Two of the most widely adopted machine learning methods are supervised learning which trains algorithms based on example input and output data that is labeled by humans, and unsupervised learning which provides the algorithm with no labeled data in order to allow it to find structure within its input data. Let’s explore these methods in more detail.

The majority of practical machine learning uses supervised learning. Supervised learning is where you have input variables (x) and an output variable (Y) and you use an algorithm to learn the mapping function from the input to the output Y = f(X). The goal is to approximate the mapping functions o well that when you have new input data(x) that you can predict the output variables(Y) for that data. Techniques of Supervised Machine Learning algorithms include linear and logistic regression, multi-class classification, Decision Trees and support vector machines. Supervised learning requires that the data used to train the algorithm is already labeled with correct answers. For example, a classification algorithm will learn to identify animals after being trained on a data set of images that are properly labeled with the species of the animal and some identifying characteristics. Supervised learning problems can be further grouped into Regression and Classification problems. Both problems have as goal the construction of a succinct model that can predict the value of the dependent attribute from the attribute variables. The difference between the two tasks is the fact that the dependent attribute is numerical for regression and categorical for classification.

1.4.1.1CLASSIFICATION

As the name suggests, Classification is the task of "classifying things" into sub-categories. But, by a machine. If that doesn't sound like much, imagine your computer being able to differentiate between you and a stranger. Between a potato and a tomato. Between an A grade and a F. In Machine Learning and Statistics, Classification is the problem of identifying to which of a set of categories (sub populations), a new observation belongs to, on the basis of a training set of data containing observations and whose categories membership is known.

Types of Classification:

Classification is of two types:

1. **Binary Classification:** When we have to categorize given data into 2 distinct classes. Example– On the basis of given health conditions of a person, we have to determine whether the person has a certain disease or not.
2. **Multi-class Classification:** The number of classes is more than 2. For Example- Based on data about different species of flowers, we have to determine which specie does our observation belong t**o-**

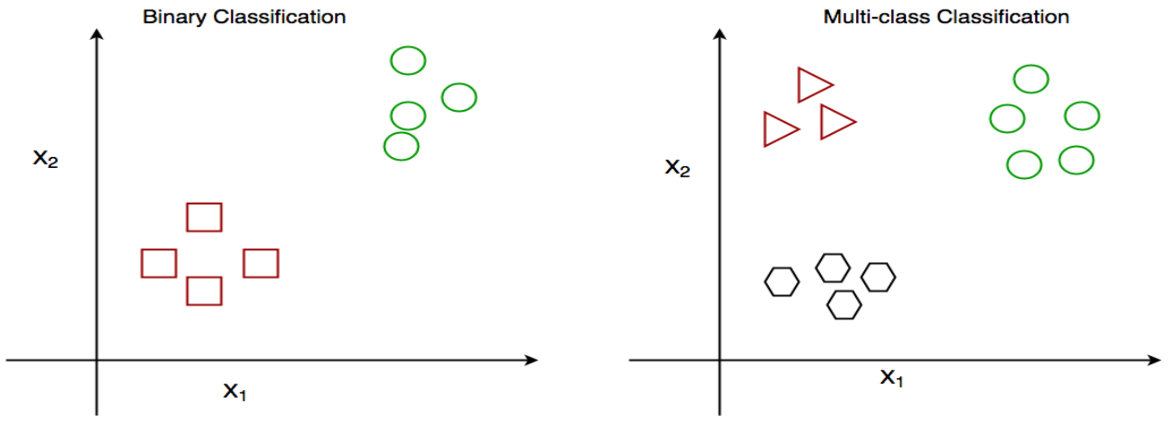


Fig2: Binary and Multi-class Classification. Here x1 and x2 are our variables upon which the class is predicted.

Suppose we have to predict whether a given patient has a certain disease or not, on the basis of 3 variables, called features.

Which means there are two possible out comes:

1. The patient has the said disease. Basically a result labeled “Yes” or “True”.
2. The patient is disease free. A result labeled “No” or “False”.
3. This is a binary classification problem. We have a set of observations called training data set, which comprises of sample data with actual classification results. We train a model, called Classifier on this dataset, and use that model to predict whether a certain patient will have the disease or not.

The outcome, thus now depends upon:

i. How well these features are able to “map” to the outcome.
ii. The quality of our data set. By quality I refer to statistical and Mathematical qualities.
iii. How well our Classifier generalizes this relationship between the features and the outcome.
iv. The values of the x1 and x2.

Following is the generalized block diagram of the classification task.

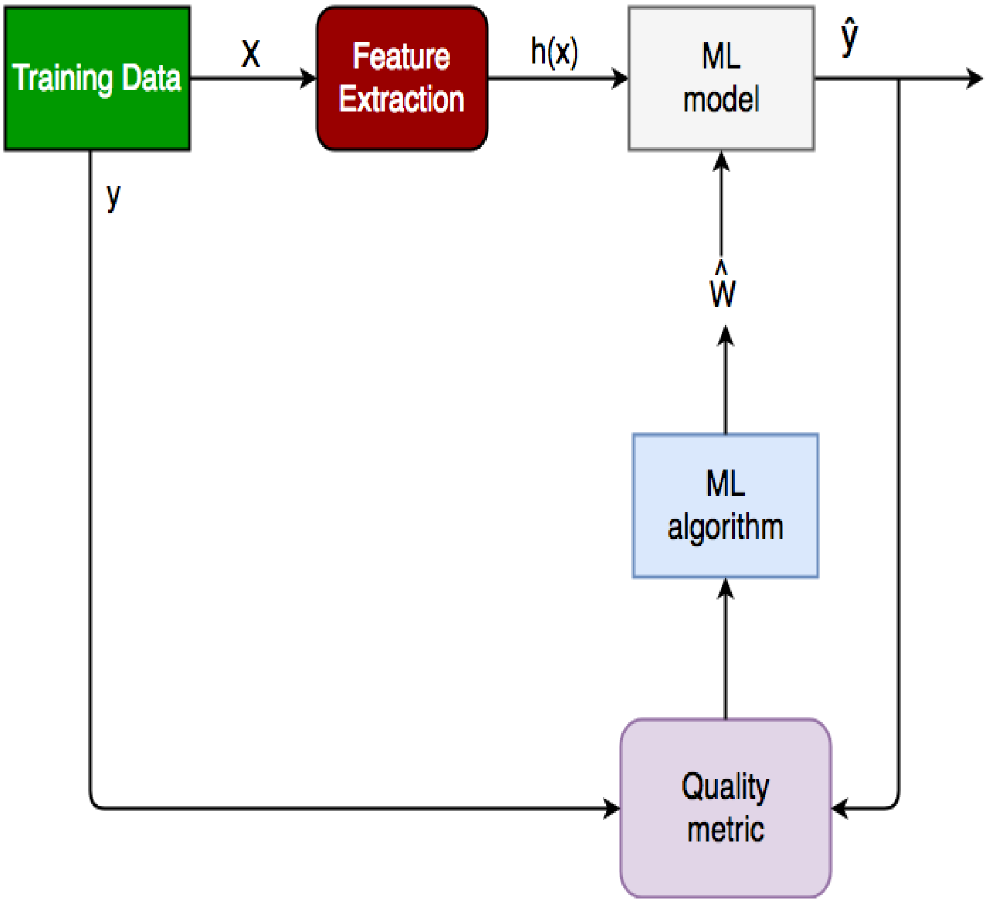


Fig 3: Generalized Classification Block Diagram.

1. X: pre-classified data, in the form of a N*M matrix. N is the no. of observations and M is the number of features
2. Y: An N-d vector corresponding to predicted classes for each of the N observations.
3. Feature Extraction: Extracting valuable information from input X using a series of transforms.
4. ML Model: The “Classifier” we’ll train.
5. Y’: Labels predicted by the Classifier.
6. Quality Metric: Metric used for measuring the performance of the model.
7. ML Algorithm: The algorithm that is used to update weights w’, which update the model and “learns” iteratively.

Types of Classifiers (Algorithms) :

There are various types of classifiers. Some of them are:

- Linear Classifiers: Logistic Regression.
- Tree Based Classifiers: Decision Tree Classifier.
- Support Vector Machines.
- Artificial Neural Networks.
- Bayesian Regression.
- Gaussian Naive Baye's Classifiers.
- Stochastic Gradient Descent (SGD) Classifier.
- Ensemble Methods: Random Forests, Ada Boost, Bagging Classifier, Voting Classifier, Extra Trees Classifier.

Practical Applications of Classification:

Google's self-driving car uses deep learning enabled classification techniques which enables it to detect and classify obstacles.

Spam E-mail filtering is one of the most widespread and well recognized uses of Classification techniques.

Detecting Health Problems, Facial Recognition, Speech Recognition, Object Detection, sentiment analysis all use classification at their core.

1.4.1.2REGRESSION

A regression problem is when the output variable is a real or continuous value, such as "salary" or "weight". Many different models can be used, the simplest is the linear regression. It tries to fit data with the best hyper-plane which goes through the points.

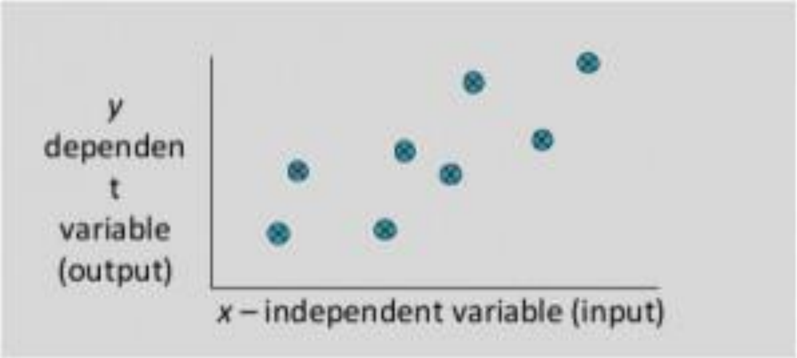


Fig 4: Linear Regression Model

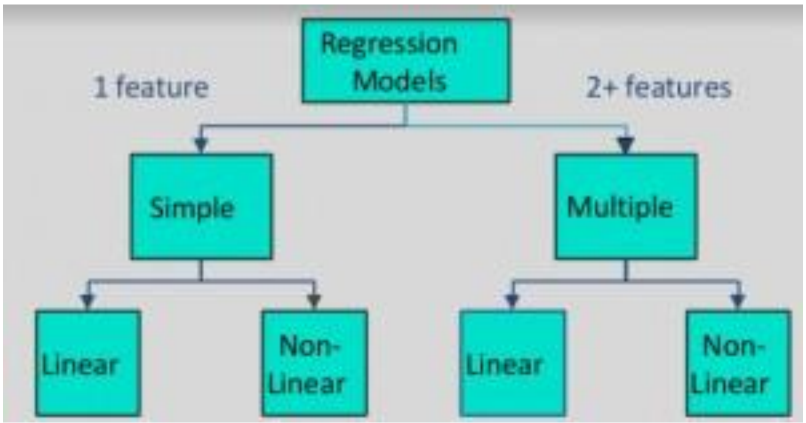


Fig 5: Types of Regression Models

1.4.2 UNSUPERVISED LEARNING

Unsupervised learning is where we only have input data (X) and no corresponding output variables. The goal for unsupervised learning is to model the underlying structure or distribution in the data in order to learn more about the data. These are called unsupervised learning because unlike supervised learning above there is no correct answer and there is no teacher. Algorithms are left to their own devises to discover and present the interesting structure in the data. Unsupervised learning problems can be further grouped into clustering and association problems.

1.4.2.1CLUSTERING

It is basically a type of unsupervised learning method. An unsupervised learning method is a method in which we draw references from data sets consisting of input data without labeled responses. Generally, it is used as a process to find meaningful structure, explanatory underlying processes, generative features, and groupings inherent in a set of examples. Clustering is the task of dividing the population or data points into a number of groups such that data points in the same groups are more similar to other data points in the same group and dissimilar to the data points in other groups. It is basically a collection of objects on the basis of similarity and dissimilarity between them.

For example, the data points in the graph below clustered together can be classified into one single group. We can distinguish the clusters, and we can identify that there are 3 clusters in the below picture.

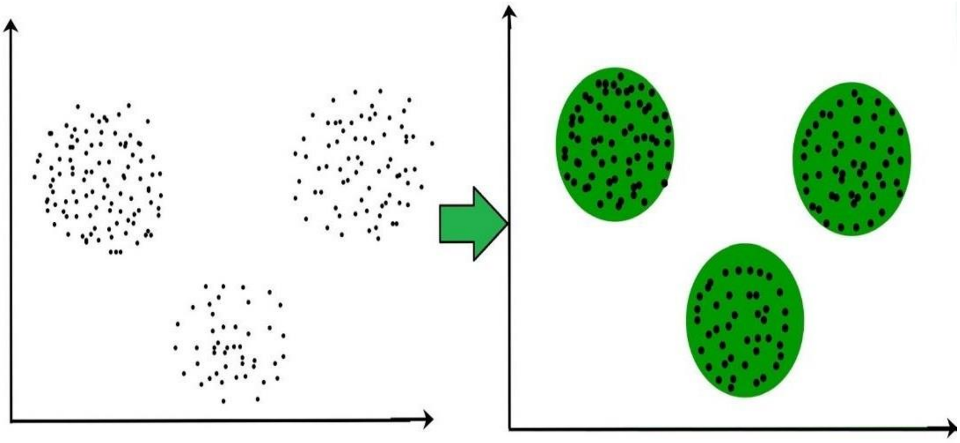


Fig 6: Clustering overview

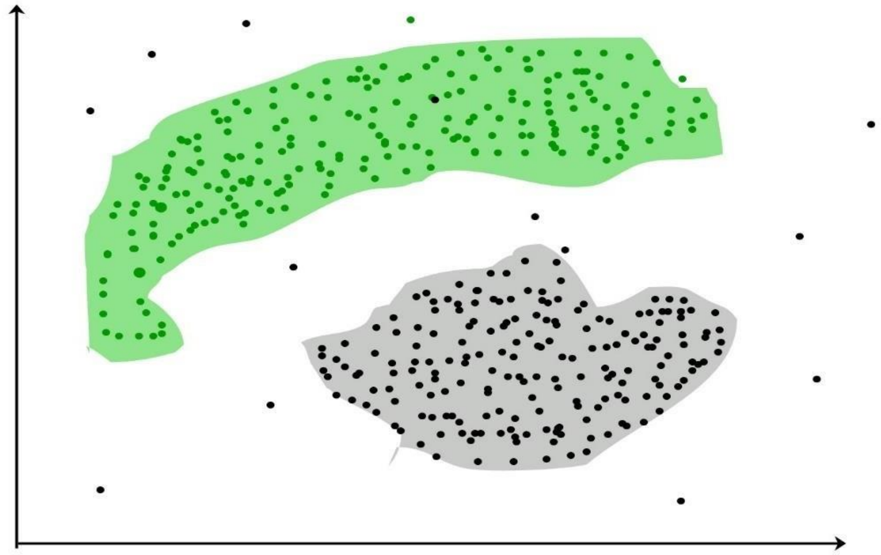


Fig 7: Clustering Example

These data points are clustered by using the basic concept that the data point lies within the given constraint from the cluster center. Various distance methods and techniques are used for calculation of the outliers.

Clustering is very much important as it determines the intrinsic grouping among the unlabeled data present. There are no criteria for a good clustering. It depends on the user, what is the criteria they may use which satisfy their need. For instance, we could be interested in finding representatives for homogeneous groups (data reduction), in finding "natural clusters" and describe their unknown properties ("natural" data types), in finding useful and suitable groupings ("useful" data classes) or in finding unusual data objects (outlier detection). This algorithm must make some assumptions which constitute the similarity of points and each assumption make different and equally valid clusters.

Clustering Methods:

1. Density-Based Methods: These methods consider the clusters as the dense region having some similarity and different from the lower dense region of the space. These methods have good accuracy and ability to merge two clusters. Example *DBSCAN* (Density-Based Spatial Clustering of Applications with Noise) , OPTICS (Ordering Points to Identify Clustering Structure)etc.

2. Hierarchical Based Methods: The clusters formed in this method forms a tree type structure based on the hierarchy. New clusters are formed using the previously formed one. It is divided into two category

- Agglomerative (bottom up approach)
- Divisive (top down approach).

3. Partitioning Methods**:** These methods partition the objects into k clusters and each partition forms one cluster. This method is used to optimize an objective criterion similarity function such as when the distance is a major parameter example K-means, CLARANS (Clustering Large Applications based upon randomized Search) etc.

4. Grid-based Methods**:** In this method the data space is formulated into a finite number of cells that form a grid-like structure. All the clustering operation done on these grids are fast and independent of the number of data objects example STING (Statistical Information Grid), wave cluster, CLIQUE (Clustering In Quest) etc.

Clustering Algorithms:

- K- Means Clustering.
- Mean-Shift Clustering for a single sliding window.
- The entire process of Mean-Shift Clustering.
- DBSCAN Smiley Face Clustering.
- EM Clustering using GMMs.
- Agglomerative Hierarchical Clustering.

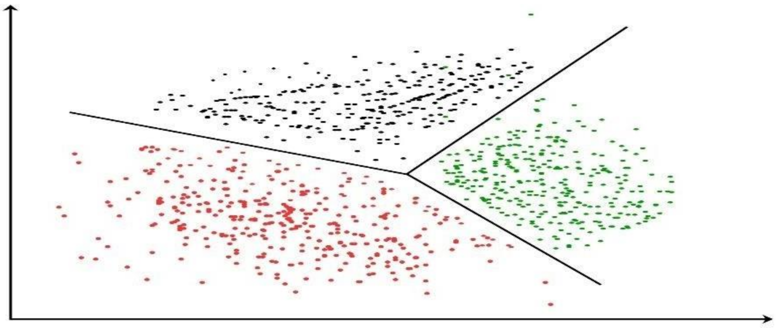


Fig 8: k-means clustering

1.5 ALGORITHMS USED

1.5.1 DECISION TREES

Tree based learning algorithms are considered to be one of the best and mostly used supervised learning methods. Tree based methods empower predictive models with high accuracy, stability and ease of interpretation. Unlike line are models, they map non- linear relationships quite well. They are adaptable at solving any kind of problem at hand (classification or regression).

Methods like decision trees, random forest, gradient boosting are being popularly used in all kinds of data science problems. Hence, for every analyst it's important to learn these algorithms and use them for modeling. Decision tree is a type of supervised learning algorithm (having a pre-defined target variable) that is mostly used in classification problems. It works for both categorical and continuous input and output variables. In this technique, we split the population or sample into two or more homogeneous sets (or sub-populations) based on most significant splitter/differentiator in input variables.

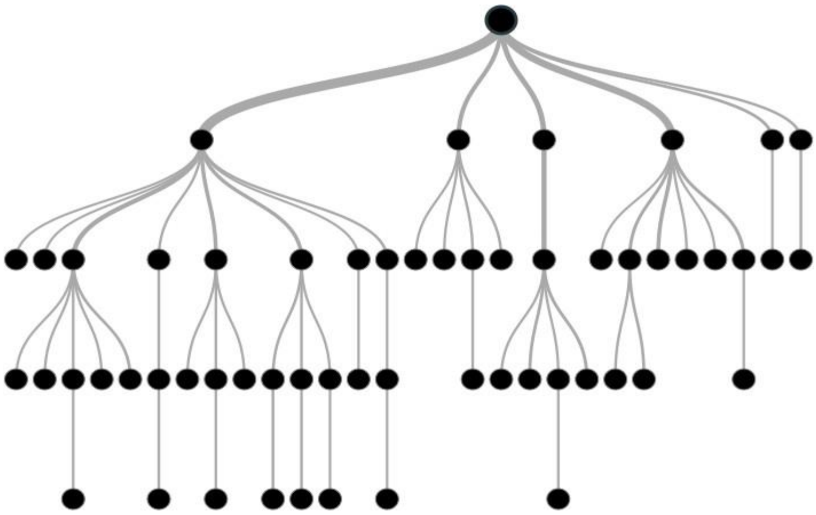


Fig 9: A Typical Decision Tree Structure

For instance, let’s say we have a sample of 30 students with three variables Gender (Boy/ Girl), Class (IX/ X) and Height (5 to 6 ft). 15 out of these 30 play cricket in leisure time. Now, I want to create a model to predict who will play cricket during leisure period? In this problem, we need to segregate students who play cricket in their leisure time based on highly significant input variable among all three.

This is where decision tree helps, it will segregate the students based on all values of three variable and identify the variable, which creates the best homogeneous sets of students (which are heterogeneous to each other). In the snapshot below, you can see that variable Gender is able to identify best homogeneous sets compared to the other two variables.

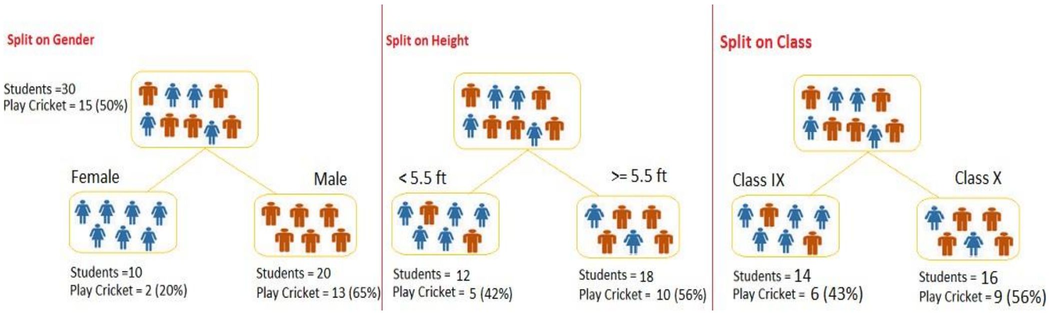


Fig 10: Example of a Tree classification

As mentioned above, decision tree identifies the most significant variable and it’s value that gives best homogeneous sets of population. Now the question which arises is, how does it identify the variable and the split? To do this, decision tree uses various algorithms, which we will shall discuss in the following section.

Types of Decision Trees:

Types of decision tree is based on the type of target variable we have. It can be of two types:

1. Categorical Variable Decision Tree**:** Decision Tree which has categorical target variable then it called as categorical variable decision tree. Example:-In above scenario of student problem, where the target variable was “Student will play cricket or not” i.e. YES or NO.
2. Continuous Variable Decision Tree**:** Decision Tree has continuous target variable then it is called as Continuous Variable Decision Tree.

For instance, let's say we have a problem to predict whether a customer will pay his renewal premium with an insurance company (yes/ no). Here we know that income of customer is a significant variable but insurance company does not have income details for all customers. Now, as we know this is an important variable, then we can build a decision tree to predict customer income based on occupation, product and various other variables. In this case, we are predicting values for continuous variable.

Important Terminology related to Decision Trees:

Here is the info about the basic terminology used with Decision trees:

- Root Node: It represents entire population or sample and this further gets divided into two or more homogeneous sets.
- Splitting: It is a process of dividing a node into two or more sub-nodes.
- Decision Node: When a sub-node splits into further sub-nodes, then it is called decision node.
- Leaf/ Terminal Node: Nodes do not split is called Leaf or Terminal node.

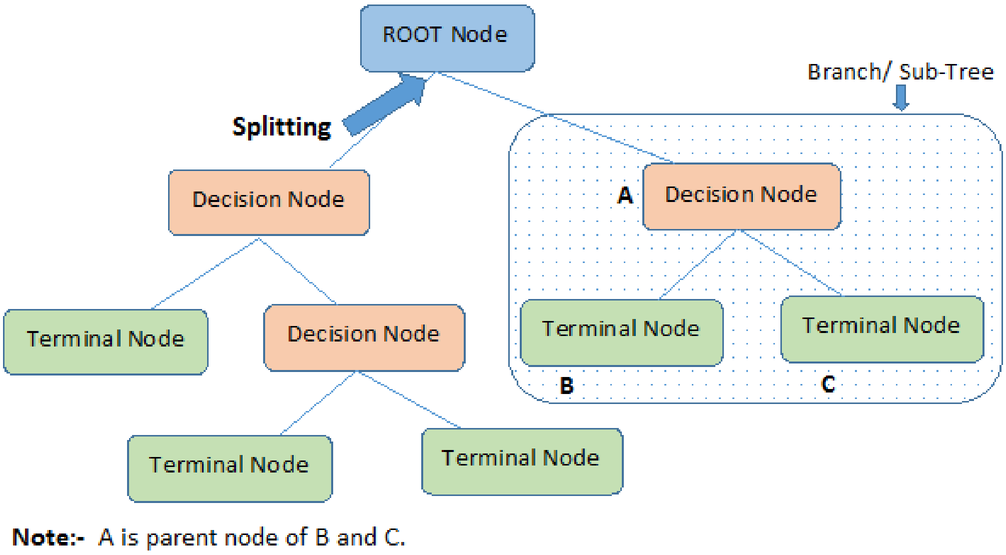


Fig 11: Process undergone in a decision tree

- Pruning: When we remove sub-nodes of a decision node, this process is called pruning. You can say opposite process of splitting.
- Branch / Sub-Tree: A sub section of entire tree is called branch or sub-tree.
- Parent and Child Node: A node, which is divided into sub-nodes is called parent node of sub-nodes where as sub-nodes are the child of parent node.

Advantages

- Easy to Understand: Decision tree output is very easy to understand even for people from non-analytical background. It does not require any statistical knowledge to read and interpret them. Its graphical representation is very intuitive and users can easily relate their hypothesis.
- Useful in Data exploration**:** Decision tree is one of the fastest ways to identify most significant variables and relation between two or more variables. With the help of decision trees, we can create new variables / features that has better power to predict target variable. It can also be used in data exploration stage. For example, we are working on a problem where we have information available in hundreds of variables, there decision tree will help to identify most significant variable.
- Less data cleaning required**:** It requires less data cleaning compared to some other modeling techniques. It is not influenced by outliers and missing values to a fair degree.
- Data type is not a constraint: It can handle both numerical and categorical variables.
- Non Parametric Method: Decision tree is considered to be a non-parametric method. This means that decision trees have no assumptions about the space distribution and the classifier structure.

Now, let us know how tree knows where to split the branch. The decision of making strategic splits heavily affects a tree’s accuracy. The decision criteria is different for classification and regression trees. Decision trees use multiple algorithms to decide to split a node in two or more sub-nodes. The creation of sub-nodes increases the homogeneity of resultant sub-nodes. In other words, we can say that purity of the node increases with respect to the target variable. Decision tree splits the nodes on all available variables and then selects the split which results in most homogeneous sub- nodes. The algorithms election is also based on type of target variables. Let’s look at the four most commonly used algorithms in decision tree:

Gini Index

Gini index says, if we select two items from a population at random then they must be of same class and probability for this is 1 if population is pure.

1. It works with categorical target variable “Success” or “Failure”.
2. It performs only Binary splits
3. Higher the value of Gini higher the homogeneity.
4. CART (Classification and Regression Tree) uses Gini method to create binary splits.

Steps to Calculate Gini for a split

1. Calculate Gini for sub-nodes, using formula sum of square of probability for success and failure (p^2+q^2).
2. Calculate Gini for split using weighted Gini score of each node of that split

Referring to example used above, where we want to segregate the students based on target variable (playing cricket or not). In the snapshot below, we split the population using two input variables Gender and Class. Now, I want to identify which split is producing more homogeneous sub-nodes using Gini index.

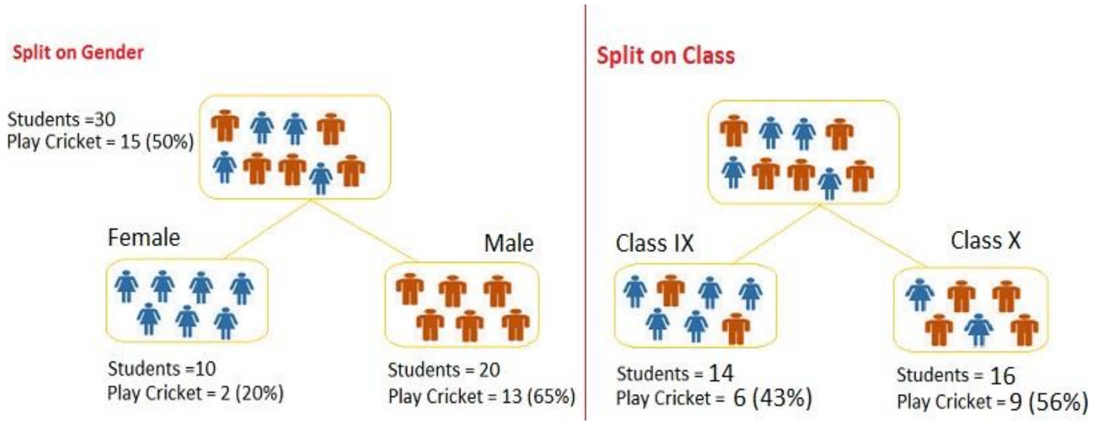


Fig 12: Example for tree classification

Split on Gender:

1. Calculate, Gini for sub-node Female = (0.2) *(0.2) +(0.8) *(0.8) = 0.68 2. Gini for sub-node Male = (0.65) *(0.65) +(0.35) *(0.35) = 0.55

3. Calculate weighted Gini for Split Gender = (10/30)*0.68+(20/30)*0.55 =0.59

Similar for Split on Class:

1. Gini for sub-node Class IX = (0.43)*(0.43)+(0.57)*(0.57)=0.51
2. Gini for sub-node Class X = (0.56)*(0.56)+(0.44)*(0.44)=0.51
3. Calculate weighted Gini for Split Class = (14/30)*0.51+(16/30)*0.51 = 0.51

Above, you can see that Gini score for *Split on Gender* is higher than *Split on Class,* hence, the node split will take place on Gender.

Chi-Square

It is an algorithm to find out the statistical significance between the differences between sub-nodes and parent node. We measure it by sum of squares of standardized differences between observed and expected frequencies of target variable.

1. It works with categorical target variable “Success” or “Failure”.
2. It can perform two or more splits.
3. Higher the value of Chi-Square higher the statistical significance of differences between sub-node and Parent node.
4. Chi-Square of each node is calculated using formula,
5. Chi-square = ((Actual – Expected)^2 /Expected)^1/2
6. It generates tree called CHAID (Chi-square Automatic Interaction Detector)

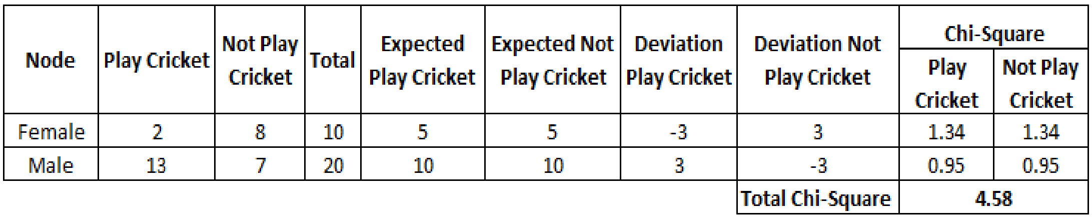
Steps to Calculate Chi-square for a split:

1. Calculate Chi-square for individual node by calculating the deviation for Success and Failure both
2. Calculated Chi-square of Split using Sum of all Chi-square of success and Failure of each node of the split

   Now, Let’s work with above example that we have used to calculate Gini.

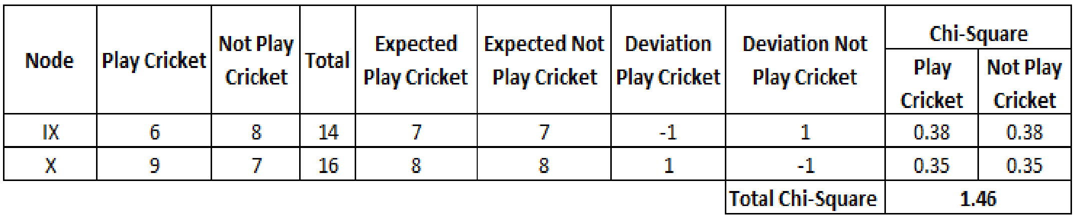
Split on Gender:

1. First we are populating for node Female, Populate the actual value for “Play Cricket” and “Not Play Cricket”, here these are 2 and 8 respectively.
2. Calculate expected value for “Play Cricket” and “Not Play Cricket”, here it would be 5 for both because parent node has probability of 50% and we have applied same probability on Female count(10)
3. Calculate deviations by using formula, Actual – Expected. It is for “Play Cricket” (2 – 5 = -3) and for “Not play cricket” (8 – 5 =3).
4. Calculate Chi-square of node for “Play Cricket” and “Not Play Cricket” using formula with formula = ((Actual–Expected)^2/Expected)^1/2. You can refer below table for calculation.



Node	Play Cricket	Not Play Cricket	Total	Expected Play Cricket	Expected Not Play Cricket	Deviation Play Cricket	Deviation Not Play Cricket	Chi-Square	
								Play Cricket	Not Play Cricket
Female	2	8	10	5	5	-3	3	1.34	1.34
Male	13	7	20	10	10	3	-3	0.95	0.95
							Total Chi-Square	4.58	

Table 1: Explaining example for Gini Index based tree classification Split on Class



Node	Play Cricket	Not Play Cricket	Total	Expected Play Cricket	Expected Not Play Cricket	Deviation Play Cricket	Deviation Not Play Cricket	Chi-Square	
								Play Cricket	Not Play Cricket
IX	6	8	14	7	7	-1	1	0.38	0.38
X	9	7	16	8	8	1	-1	0.35	0.35
							Total Chi-Square	1.46	

Table 2: Explaining split on class classification tree

1. Follow similar steps for calculating Chi-square value for Male node.
2. Now add all Chi-square values to calculate Chi-square for split Gender.

Performing similar steps of calculation for split on Class and we will come up with below table. Above, we can see that Chi-square also identify the Gender split is more significant compare to Class.

Information Gain:

We can easily say by looking at below figure which node can describe further process easily, that is C because it requires less information as all values are similar. On the other hand, B requires more information to describe it and A requires the maximum information. In other words, we can say that C is a Pure node, B is less Impure, and A is more impure.

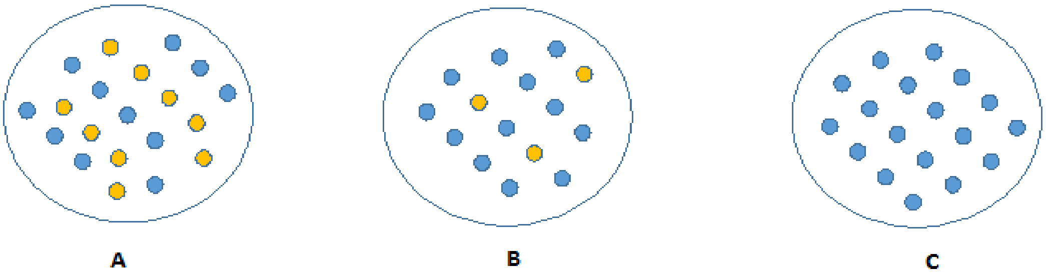


Fig 13: Example of best classification

Now, we can build a conclusion that less impure node requires less information to describe it. And, more impure node requires more information. Information theory is a measure to define this degree of disorganization in a system known as Entropy. If the sample is completely homogeneous, then the entropy is zero and if the sample is an equally divided (50% – 50%), it has entropy of one.

Entropy can be calculated using formula

$$\text{Entropy} = -p \log_2 p - q \log_2 q$$

Here p and q is probability of success and failure respectively in that node. Entropy is alsousedwithcategoricaltargetvariable.Itchoosesthesplitwhichhaslowestentropy compared to parent node and other splits. The lesser the entropy, the better itis.

Steps to calculate entropy for a split:

1. Calculate entropy of parent node
2. Calculate entropy of each individual node of split and calculate weighted average of all sub-nodes available in split.

Example: Let’s use this method to identify best split for student example.

1. Entropy for parent node=-(15/30)log2(15/30)–(15/30)log2(15/30)=1.Here 1 shows that it is a impure node.
2. Entropy for Female node = -(2/10) log2 (2/10) – (8/10) log2 (8/10) = 0.72 and for male node, -(13/20) log2 (13/20) – (7/20) log2 (7/20) = 0.93
3. Entropy for split Gender = Weighted entropy of sub-nodes = (10/30)*0.72 + (20/30)*0.93 =0.86
4. Entropy for Class IX node, -(6/14) log2 (6/14) – (8/14) log2 (8/14) = 0.99 and for Class X node, -(9/16) log2 (9/16) – (7/16) log2 (7/16) = 0.99.
5. Entropy for split Class = (14/30)*0.99 + (16/30)*0.99 = 0.99

Above, we can see that entropy for Split on Gender is the lowest among all, so the tree will split on gender. We can derive information gain from entropy as 1-Entropy.

1.5.2 SUPPORT VECTORMACHINE:

Support Vector Machine (SVM) is a supervised machine learning algorithm which can be used for both classification or regression challenges. However, it is mostly used in classification problems. In this algorithm, we plot each data item as a point in n- dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiate the two classes very well.

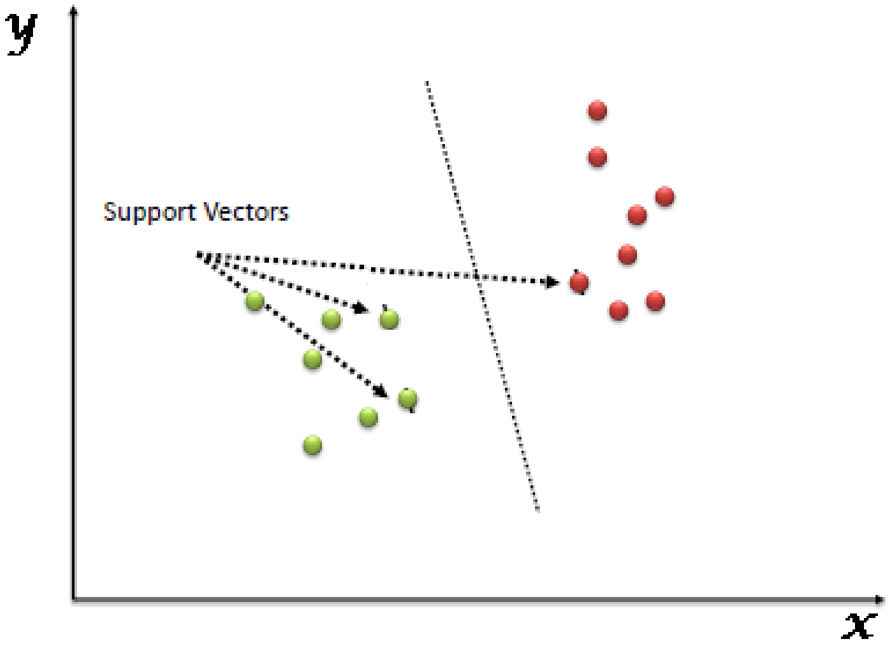


Fig 14: Support vector machine

Support Vectors are simply the co-ordinates of individual observation. Support Vector Machine is a frontier which best segregates the two classes (hyper-plane/ line).

Working of an SVM:

- Identify the right hyper-plane (Scenario-1): Here, we have three hyper-planes (A, B and C). Now, identify the right hyper-plane to classify star and circle.

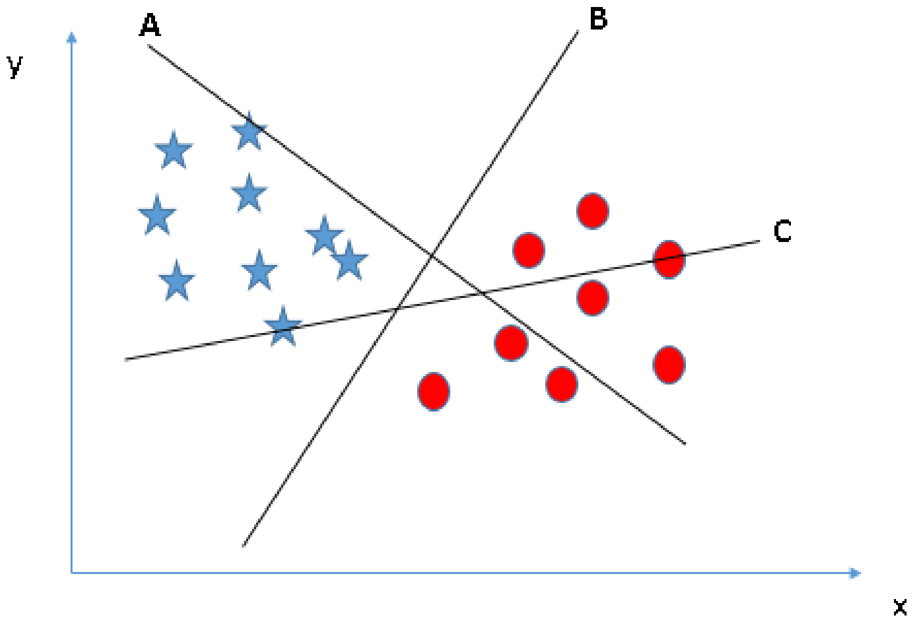
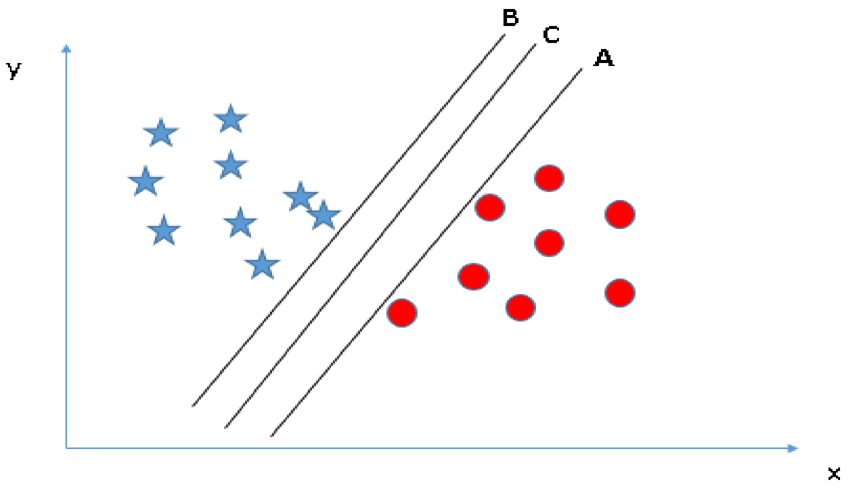


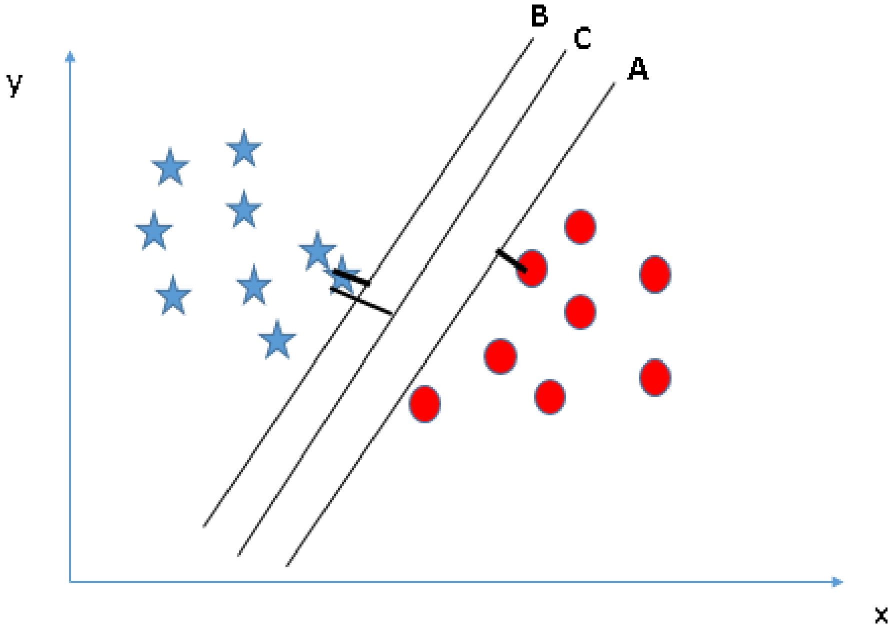
Fig 15: Steps involved in Working of an SVM

In this scenario, hyper-plane “B” has excellently performed this job.

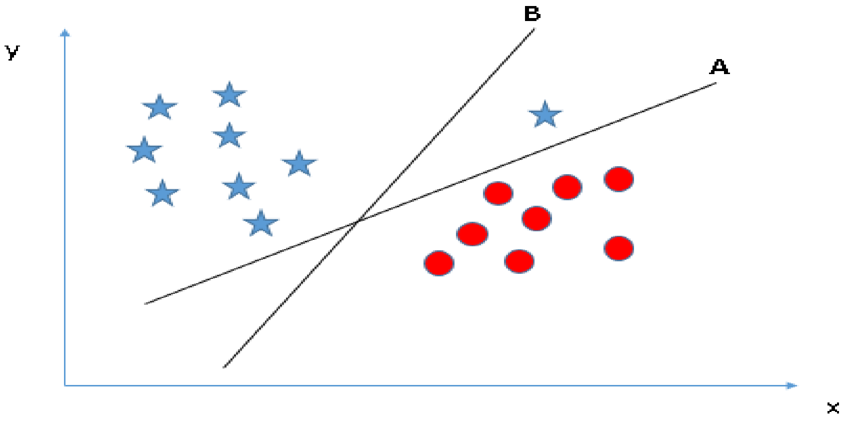
- Identify the right hyper-plane (Scenario-2): Here, we have three hyper-planes (A, B and C) and all are segregating the classes well. Now, How can we identify the right hyper-plane?



- Here, maximizing the distances between nearest data point (either class) and hyper-plane will help us to decide the right hyper-plane. This distance is called as Margin. Let's look at the below snapshot:

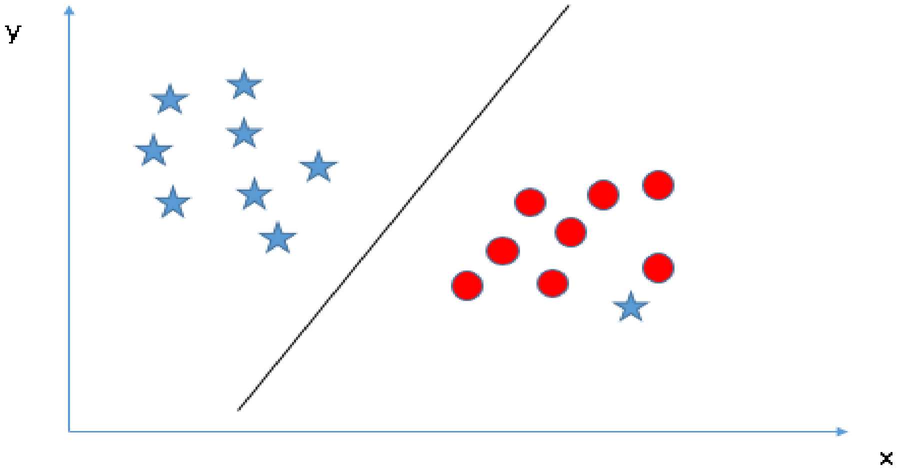


- Above, you can see that the margin for hyper-plane C is high as compared to both A and B. Hence, we name the right hyper-plane as C. Another lightning reason for selecting the hyper-plane with higher margin is robustness. If we select a hyper-plane having low margin then there is high chance of miss- classification.
- Identify the right hyper-plane (Scenario-3)**:** Use the rules as discussed in previous section to identify the right hyper-plane



Some of we may have selected the hyper-plane B as it has higher margin compared to A. But, here is the catch, SVM selects the hyper-plane which classifies the classes accurately prior to maximizing margin. Here, hyper-plane B has a classification error and A has classified all correctly. Therefore, the right hyper-plane is A.

- Can we classify two classes (Scenario-4): Below, I am unable to segregate the two classes using a straight line, as one of star lies in the territory of other (circle) class as an outlier.
- As I have already mentioned, one star at other end is like an outlier for star class. SVM has a feature to ignore outliers and find the hyper-plane that has maximum margin. Hence, we can say, SVM is robust to outliers.



- Find the hyper-plane to segregate to classes (Scenario-5): In the scenario below, we can’t have linear hyper-plane between the two classes, so how does SVM classify these two classes? Till now, we have only looked at the linear hyper-plane.
- SVM can solve this problem. Easily! It solves this problem by introducing additional feature. Here, we will add a new feature z=x^2+y^2. Now, let’s plot the data points on axis x and z:

- In above plot, points to considerate:
- All values for z would be positive always because z is the squared sum of both x and y
- In the original plot, red circles appear close to the origin of x and y axes, leading to lower value of z and star relatively away from the origin result to higher value of z.

- In SVM, it is easy to have a linear hyper-plane between these two classes. But, another burning question which arises is, should we need to add this feature manually to have a hyper-plane. No, SVM has a technique called the kernel trick. These are functions which takes low dimensional input space and transform it to a higher dimensional space i.e. it converts not separable problem to separable problem, these functions are called kernels. It is mostly useful in non-linear separation problem. Simply put, it does some extremely complex data transformations, then find out the process to separate the data based on the labels or outputs you've defined.

1.5.3 ONE HOT ENCODING:

One Hot Encoding is a technique by which categorical values present in the data collected are converted into numerical or other ordinal format so that they can be provided to machine learning algorithms and get better results of prediction. Simply One Hot encoding transforms categorical values into a form that best fits as input to feed to various machine learning algorithms. This algorithm works fine with almost all machine learning algorithms. Few algorithms like random forest handle categorical values very well. In such cases One Hot encoding is not required. Process of One Hot encoding may seem difficult but most modern day machine learning algorithms take care of that. The process is easily explained here: For example in a data if there are values like yes and no., integer encoder assigns values to them like 1 and 0.This process can be followed as long as we continue the fixed values for yes as 1 and no as 0. As long as we assign or allocate these fixed numbers to these particular labels this is called as integer encoding. But here consistency is very important because if we invert the encoding later, we should get back the labels correctly from those integer values especially in the case of prediction. Next step is creating a vector for each integer value. Let us suppose this vector is binary and has a length of 2 for the two possible integer values. The 'yes' label encoded as 1 will then be represented with vector [1,1] where the zeroth index is given the value 1.Similarly 'no' label encoded as '0' will be represented like [0,0] which represents the first index is represented with value 0.

For example [pillow, rat, fight, rat] becomes [0,1,2,1]. This is here imparting an ordinal property to the variable, i.e. pillow & lt; ret & lt; fight. As this is ordinally characteristic and is usually not required and desired and so One Hot encoding is required for correct representation of distinct elements of available. It makes representation of categorical variables to be more expressive.

1.5.4 XGBOOST

XG-Boost is an open-source software library which provides the gradient boosting framework for C++, Java, Python, R, and Julia. It works on Linux, Windows, and Mac-OS. From the project description, it aims to provide a "Scalable, Portable and Distributed Gradient Boosting (GBM, GBRT, GBDT) Library". Other than running on a single machine, it also supports the distributed processing frameworks Apache Hadoop, Apache Spark, and Apache Flink. It has gained much popularity and attention recently as it was the algorithm of choice for many winning teams of a number of machine learning competitions.

XG-Boost initially started as are search project by Tianqi Chen as part of the Distributed (Deep) Machine Learning Community (DMLC) group. Initially, it began as a terminal application which could be configured using a lib svm configuration file. After winning the Higgs Machine Learning Challenge, it became well known in the ML competition circles. Soon after, the Python and R packages were built and now it has packages for many other languages like Julia, Scala, Java, etc. This brought the library to more developers and became popular among the Kaggle community where it has been used for a large number of competitions.

It soon became used with multiple other packages making it easier to use in the respective communities. It now has integrations with scikit-learn for Python users, and also with the caret package for R users. It can also be integrated into Data Flow frameworks like Apache Spark, Apache Hadoop, and Apache Flink using the abstracted Rabbit and XG-Boost 4J. The working of XG-Boost has also been published by Tianqi Chen and Carlos Guestrin.

XG-Boost denotes eXtreme Gradient Boosting. XG-Boost is implementation of gradient boosting algorithms. It is available in many forms like tool, library at cetera. It mainly focuses on model performance and computational time. It greatly reduces the time and greatly lifts the performance of the model. Its implementation has the features of scikit-learn and R implementations and also have a newly added features like regularization. Regularized gradient boosting means gradient boosting with both L1 and L2 type regularizations. The main best features that the implementation of the algorithm provides are: Automatic handling of missing values with sparse aware implementation, and it provides block structure to promote parallel construction of tree and continued training which supports further boost an already fitted model on the fresh data. Gradient boosting is a technique where new models are made that can predict the errors or remains of previous models and then added together to make the final prediction. they use gradient descendent algorithms to reduce loss during adding of new models. They support both classification and regression type of challenges. In the training part generally an objective function is defined. Define an objective function and try to optimize it.

$obj=i=1\ \Sigma l(yi,yi\hat{}(t))+\ i=1\ \Sigma\ \Omega(fi)$

CHAPTER - 2

2.1 SOFTWARES USED

2.1.1 ANACONDA

Anaconda is a free and open source distribution of the Python and R programming languages for data science and machine learning related applications (large-scale data processing, predictive analytics, scientific computing), that aims to simplify package management and deployment. Package versions are managed by the package management system conda. Anaconda Distribution is used by over 6million users, and it includes more than 250 popular data science packages suitable for Windows, Linux, and Mac-OS.

Python is a high-level programming language devised by Guido van Rossum & first released in 1991. It’s the most popular coding language used by software developers to build, control, manage and for testing. It is also an interpreter which executes Python programs. The python interpreter is called python.exe on Windows.

Python Packages

Packages or additional libraries help in scientific computing and computational modeling. In Python, the packages are not the part of the Python standard library. Few major packages are

- numpy (NUMeric Python): matrices and linear algebra
- scipy (SCIentific Python): many numerical routines
- matplotlib: (PLOTtingLIBrary) creating plots of data
- sympy (SYMbolic Python): symbolic computation
- pytest (Python TESTing): a code testing frame work

Together with a list of Python packages, tools like editors, Python distributions include the Python interpreter. Anaconda is one of several Python distributions. Anaconda is a new distribution of the Python and R data science package. It was formerly known as Continuum Analytics. Anaconda has more than 100 new packages.

This work environment, Anaconda is used for scientific computing, data science, statistical analysis, and machine learning. The latest version of Anaconda 5.0.1 is released in October 2017.

The released version 5.0.1 addresses some minor bugs and adds useful features, such as updated R language support. All of these features weren’t available in the original release.

This package manager is also an environment manager, a Python distribution, and a collection of open source packages and contains more than 1000 R and Python Data Science Packages.

Anaconda can help with

- Installing Python on multiple platforms
- Separating out different environment
- Dealing with not having correct privileges and
- Getting up and running with specific packages and libraries

2.1.2 IPYTHON NOTEBOOKS

IPython is a command shell for interactive computing in multiple programming languages, originally developed for the Python programming language, that offers introspection, rich media, shell syntax, tab completion, and history. IPython provides the following features:

- Interactive shells (terminal and Qt-based).
- A browser-based notebook interface with support for code, text, mathematical expressions, inline plots and other media.
- Support for interactive data visualization and use of GUI toolkits.
- Flexible, embeddable interpreters to load into one's own projects.
- Tools for parallel computing.

Parallel computing in Ipython:

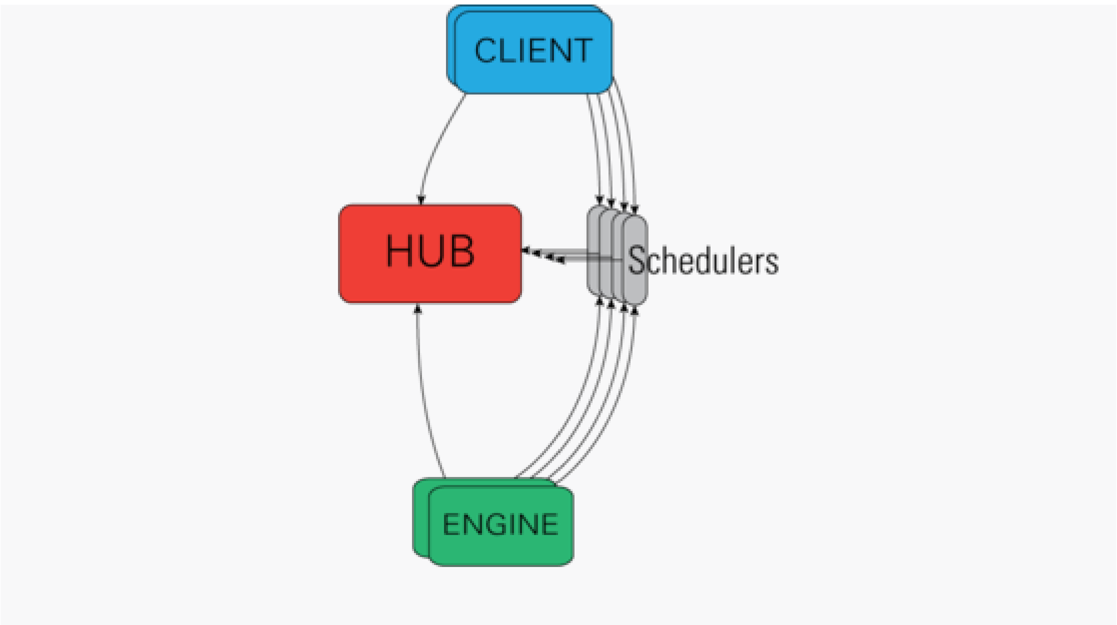


Fig 16: Architectural View of IPython's parallel machinery

IPython is based on an architecture that provides parallel and distributed computing.

IPython enables parallel applications to be developed, executed, debugged and monitored interactively. Hence, the I (Interactive) in IPython. This architecture abstracts out parallelism, which enables IPython to support many different styles of parallelism including:

- Single program, multiple data (SPMD) parallelism.
- Multiple program, multiple data (MIMD) parallelism.
- Message passing using MPI.
- Task parallelism.
- Data parallelism.
- Combinations of these approaches.
- Custom user defined approaches.

With the release of IPython 4.0, the parallel computing capabilities have been made optional and released under the ipy parallel python package.

IPython frequently draw from SciPy stack libraries like NumPy and SciPy, often installed alongside from one of many Scientific Python distributions. IPython provide integration some library of the SciPy stack like matplotlib, like inline graph when in used with the Jupyter notebook. Python libraries can implement IPython specific hooks to customize object Rich object display. SymPyfor example implement rendering of Mathematical Expression as rendered LaTeX when used within IPython context.

Other features:

IPython also allows non-blocking interaction with Tkinter, PyGTK, PyQt/PySide and wxPython (the standard Python shell only allows interaction with Tkinter). IPython can interactively manage parallel computing clusters using asynchronous status call-backs and/or MPI. IPython can also be used as a systems hell replacement. Its default behavior is largely similar to Unixshells, but it allows customization and the flexibility of executing code in alive Python environment. Using IPython as a shell replacement is less common and it is now recommended to use Xonsh which provide most of the IPython feature with better shell integrations.

2.1.3 JUPYTER NOTEBOOK

It is a web based interface that allows for rapid prototyping and sharing of data-related projects. It works with many kernels (this is the name given to the code env that it can run), including but not limited to Python and R (even though its more famous and suited for Python).Previously it used to be called IPython Notebook but has been renamed and moved to the Jupyter project. Jupyter is an open source project aiming at creating a better work experience for (data) scientists.

Unique features:

- Supports all imports and exports.
- Great for sharing and collaborative work.
- Supports variety of data types within the same window such as text, code, graphs, videos, pictures.
- Great for Visualizations.
- Enables parallel computing.
- Presentation feature in the jupyter and ipython notebook where you can make the presentation directly from your notebook. One such extension is “RISE”.

2.1.4 SPYDER

Spyder (formerly Pydee) is an open source cross-platform integrated development environment (IDE) for scientific programming in the Python language. Spyder integrates NumPy, SciPy, Matplotlib and IPython, as well as other open source software. It is released under the MIT license.

Spyder is extensible with plugins, includes support for interactive tools for data inspection and embeds Python-specific code quality assurance and introspection instruments, such as Pyflakes, Pylinth and Rope. It is available cross-platform through Anaconda, on Windows with WinPython and Python (x,y), on Mac-OS through MacPorts, and on major Linux distributions such as ArchLinux, Debian, Fedora, Gentoo Linux, openSUSE and Ubuntu.

Spyder makes use of Qt either through the binding PyQtor PySide. This flexibility is reached through a small abstraction layer called QtPy.

Features include:

- Editor with syntax highlighting and introspection for code completion.
- Support for multiple Python consoles (including IPython).
- The ability to explore and edit variables from a GUI.

2.1.5 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. Weka contains a collection of visualization tools and algorithms for data analysis and predictive modeling, together with graphical user interfaces for easy access to these functions. The original non-Java version of Weka was a Tcl/Tkfront-end to (mostly third-party) modeling algorithms implemented in other programming languages, plus data pre-processing utilities in C, and a Make file-based system for running machine learning experiments. This original version was primarily designed as a tool for analyzing data from agricultural domains, but the more recentfully Java- based version (Weka 3), for which development started in 1997, is now used in many different application areas, in particular for educational purposes and research. Advantages of Weka include:

- Free availability under the GNU General Public License.
- Portability, since it is fully implemented in the Java programming language and thus runs on almost any modern computing platform.
- A comprehensive collection of data preprocessing and modeling techniques.
- Ease of use due to its graphical user interfaces.

Weka supports several standard data mining tasks, more specifically, data preprocessing, clustering, classification, regression, visualization, and feature selection. All of Weka's techniques are predicated on the assumption that the data is available as one flat file or relation, where each data point is described by a fixed number of attributes (normally, numeric or nominal attributes, but some other attribute types are also supported). Weka provides access to SQL databases using Java Database Connectivity and can process the result returned by a database query. Weka provides access to deep learning with Deep learning. It is not capable of multi-relational data mining, but there is separate software for converting a collection of linked database tables into a single table that is suitable for processing using Weka. Another important area that is currently not covered by the algorithms included in the Weka distribution is sequence modeling.

Weka's main user interface is the Explorer, but essentially the same functionality can be accessed through the component-based Knowledge Flow interface and from the command line. There is also the Experimenter, which allows the systematic comparison of the predictive performance of Weka's machine learning algorithms on a collection of datasets. The Explorer interface features several panels providing access to the main components of the work bench:

- The Pre-process panel has facilities for importing data from a database, a comma- separated values (CSV) file, etc., and for pre-processing this data using a so- called filtering algorithm. These filters can be used to transform the data (e.g., turning numeric attributes into discrete ones) and make it possible to delete instances and attributes according to specific criteria.
- The Classify panel enables applying classification and regression algorithms (indiscriminately called classifiers in Weka) to the resulting dataset, to estimate the accuracy of the resulting predictive model, and to visualize erroneous predictions, receiver operating characteristic (ROC) curves, etc., or the model itself (if the model is amenable to visualization like, e.g., a decision tree).

- The Associate panel provides access to association rule learners that attempt to identify all important interrelationships between attributes in the data.
- The Cluster panel gives access to the clustering techniques in Weka, e.g., the simple k-means algorithm. There is also an implementation of the expectation maximization algorithm for learning a mixture of normal distributions.
- The Select attributes panel provides algorithms for identifying the most predictive attributes in a dataset.
- The Visualize panel shows a scatter plot matrix, where individual scatter plots can be selected and enlarged, and analyzed further using various selection operators.

  Weka has a large number of regression and classification tools. Native packages are the ones included in the executable Weka software, while other non-native ones can be downloaded and used within R.Weka environment. Among the native packages, the most famous tool is the M5p model tree package. The full list of tools is available here. Some of the regression tools are:

- M5Rules (M5' algorithm presented in terms of mathematical function without a tree).
- Decision Stump (same as M5' but with a single number output in each node).
- M5P (splitting domain into successive binary regions and then fit linear models to each tree node).
- Random Forest (several model trees combined).
- Rep Tree (several model trees combined).
- ZeroR (the average value of outputs).
- Decision Rules (splits data into several regions based on a single independent variable and provides a single output value for each range).
- Linear Regression.
- SMOreg (support vector regression).
- Simple Linear Regression (uses an intercept and only 1 input variable for multi variate data).
- Multi-Layer Perceptron (neural network).
- Gaussian Processes.

CHAPTER - 3

3.1 INSTALLATION PROCEDURES

The primary requirement of machine learning projects is data. To earn datasets needed for these kind of projects data is collected in three ways.

a) Randomly generated data: Basically, 25 percent of the data used in this project is generated randomly and this data does not fit well into algorithms as it does not follow any pattern. To get better accuracy with this type of data try to establish variable relationships among variables so that a hidden pattern is created.

To generate this kind of data, use any language and generate a script giving all the input requirements and values that are to be filled in the data and how many number of requirements that are needed. In this project there is a file named data generation. java and this file has to be compiled. To use to this and generate a dataset an application called eclipse is need to be installed.

For this first java is needed to be installed on the pc. To install java run the following commands:

Introduction

Java and the JVM (Java's virtual machine) are widely used and required for many kinds of software. This article will guide you through the process of installing and managing different versions of Java using apt-get.

Prerequisites

To follow this tutorial, you will need:

- One Ubuntu 16.04server.
- A sudo non-root user, which you can set up by following the Ubuntu 16.04 initial server setup guide.

Installing the Default JRE/JDK

The easiest option for installing Java is using the version packaged with Ubuntu. Specifically, this will install Open JDK 8, the latest and recommended version.

First, update the package index.

sudo apt-getupdate

Next, install Java. Specifically, this command will install the Java Runtime Environment (JRE).

- sudo apt-get install default-jre

There is another default Java installation called the JDK (Java Development Kit).The JDK is usually only needed if you are going to compile Java programs or if the software that will use Java specifically require sit.

The JDK does contain the JRE, so there are no disadvantages if you install the JDK instead of the JRE, except for the larger file size.

You can install the JDK with the following command:

- sudo apt-get install default-jdk

Installing the Oracle JDK

If you want to install the Oracle JDK, which is the official version distributed by Oracle, you will need to follow a few more steps.

First, add Oracle's PPA, then update your package repository.

- sudo add-apt-repositoryppa:webupd8team/java
- sudo apt-getupdate

Then, depending on the version you want to install, execute one of the following commands:

Oracle JDK 8

This is the latest stable version of Java at time of writing, and the recommended version to install. You can do so using the following command:

- sudo apt-get install oracle-java8-installer

Oracle JDK 9

This is a developer preview and the general release is scheduled for March 2017. It's not recommended that you use this version because there may still be security issues and bugs. There is more information about Java 9 on the official JDK 9 website.

To install JDK 9, use the following command:

- sudo apt-get install oracle-java9-installer

Managing Java

There can be multiple Java installations on one server. You can configure which version is the default for use in the command line by using update-alternatives, which manages which symbolic links are used for different commands.

- sudo update-alternatives --configjava

The output will look something like the following. In this case, this is what the output will look like with all Java versions mentioned above installed.

```
                                        Output
There are 5 choices for the alternative java (providing /usr/bin/java).


  Selection    Path                                            PriorityStatus
.............................................................................-
* 0           /usr/lib/jvm/java-8-openjdk-amd64/jre/bin/java          1081     automode
  1          /usr/lib/jvm/java-6-oracle/jre/bin/java            1        manual mode
  2          /usr/lib/jvm/java-7-oracle/jre/bin/java            2        manual mode
  3
  4          /usr/lib/jvm/java-8-openjdk-amd64/jre/bin/java          1081       manual mode
  5          /usr/lib/jvm/java-8-oracle/jre/bin/java            3        manual mode
             /usr/lib/jvm/java-9-oracle/bin/java               4        manual mode



Press <enter> to keep the current choice [*], or type selection number:
```

You can now choose the number to use as a default. This can also be done for other Java commands, such as the compiler (javac), the documentation generator (javadoc), the JAR signing tool (jarsigner), and more. You can use the following command, filling in the command you want to customize.

- sudo update-alternatives --configcommand

  Setting the JAVA_HOME Environment Variable

  Many programs, such as Java servers, use the JAVA_HOME environment variable to determine the Java installation location. To set this environment variable, we will first need to find out where Java is installed. You can do this by executing the same command as in the previous section:

- sudo update-alternatives --configjava

  Copy the path from your preferred installation and then open /etc/environment using nano or your favorite text editor.

- sudo nano/etc/environment

  At the end of this file, add the following line, making sure to replace the highlighted path with your own copied path.

  JAVA_HOME="/usr/lib/jvm/java-8-oracle" Save and exit the file, and reload it.

- source/etc/environment

  You can now test whether the environment variable has been set by executing the following command:

- echo $JAVA_HOME

  This will return the path you just set. Conclusion

  You have now installed Java and know how to manage different versions of it. You can now install software which runs on Java, such as Tomcat, Jetty, Glassfish, Cassandra, or Jenkins.

  And after installation of java is completed., install eclipse using the commands:

Sudo apt-get update apt-get install eclipse

Then after installation is completed open eclipse from menu and create a project, name a class and place the code of datageneration.java into the new class created in the eclipse. Run the project. A csv file is generated in the workspace folder of the eclipse.

b) Another way of getting the data is through linked in api. One needs to get access token from linked In and then persons person ID must be known. Then using the link https://api.linkedin.com/v2/people/(id:{person ID})

One can get the profile of another person. But getting the access token is difficult and a long process. Also if we want to grab profiles of lot of users then knowing person Id of every person from url is too difficult for that many number of users.

c) And the final method is to create a google form or something and filling it by alumni or employees.

Now after getting the data, to run the main code install anaconda in the pc. To install anaconda first, one must install conda. Commands for that are:

Sudo apt-get update apt-get install conda

To set path there are 2 ways:

Always when terminal is open execute this command

$ export PATH=~/anaconda3/bin:$PATH

And other way is enter the command

$ gedit ~/.bashsrh.sh

and at the end type the following command and save and exit

$ export PATH=~/anaconda3/bin:$PATH.

After completion of conda installation install anaconda. Commands for installation:

The way below utilizes bash scripts which is a faster way to install anaconda. This should work on Ubuntu 12.04 (precise), 14.04 (trusty), and 16.04 (xenial).

1. Open a new terminal.
2. Copy and paste the paste commands from either gist for python 3 below on the terminal.

```
#go to home directory
$ cd ~

#You can change what anaconda version you want at

# https://repo.continuum.io/archive/

$ wget https://repo.continuum.io/archive/Anaconda3-5.0.1-Linux-x86_64.sh

$ bash Anaconda3-5.0.1-Linux-x86_64.sh -b -p ~/anaconda

$ rm Anaconda3-5.0.1-Linux-x86_64.sh

$ echo 'export PATH="~/anaconda/bin:$PATH"' >> ~/.bashrc

# Refresh basically

$ source .bashrc

$ conda update conda

$ conda install anaconda-navigator
```

After successful installation to open the terminal and type anaconda-navigator. After the navigator is open launch jupyter notebook.

After opening click new file->python3 option to create a new python notebook.
Then from the file of project code. Ipynb take the code line by line and paste in jupyter and click run. Finally we can get the results and required output.

CHAPTER - 4

4.1 PROCESS FLOW ANDIMPLEMENTATION

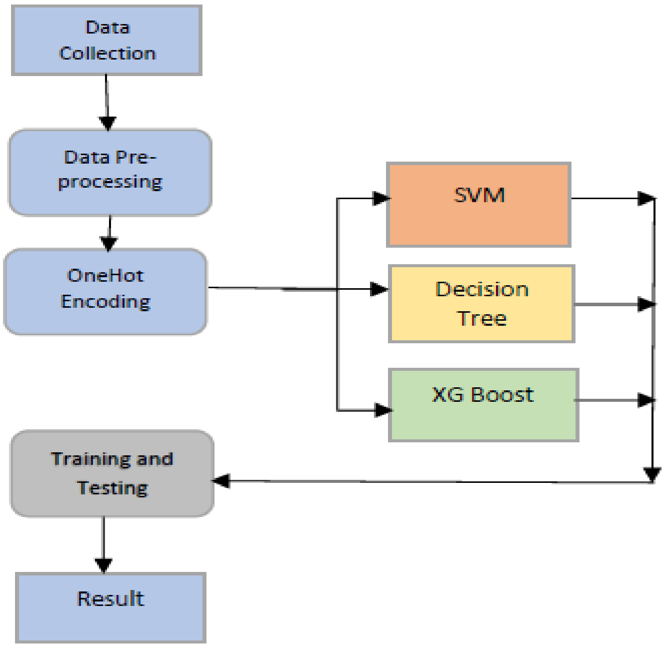


Fig 17: Process Flow

1. Data Collection:

Collection of data is one of the major and most important tasks of any machine learning projects. Because the input we feed to the algorithms is data. So, the algorithms efficiency and accuracy depend upon the correctness and quality of data collected. So as the data same will be the output. For student career prediction many parameters are required like students' academic scores in various subjects, specializations, programming and analytical capabilities, memory, personal details like relationship, interests, sports, competitions, hackathons, workshops, certifications, books interested and many more. As all these factors play vital role in deciding student's progress towards a career area, all these are taken in-to consideration. Data is collected in many ways. Some data is collected from employees working in different organizations, some amount of data is collected through LinkedIn api, some amount of data is randomly generated and other from college alumni database. Totally nearly 20 thousand records with 36 columns of data is collected.

2. Data Pre-processing:

Collecting the data is one task and making that data useful is an-other vital task. Data collected from various means will be in an unorganized format and there may be lot of null values, in-valid data values and unwanted data. Cleaning all these data and replacing them with appropriate or approximate data and removing null and missing data and replacing them with some fixed alternate values are the basic steps in preprocessing of data. Even data collected may contain completely garbage values. It may not be in exact format or way that is meant to be. All such cases must be verified and replaced with alternate values to make data meaning meaningful and useful for further processing. Data must be kept in a organized format.

3. Application of Algorithms:

The next step is algorithms are applied to data and results are noted and observed. The algorithms are applied in the fashion mention in the diagram to improve accuracy at each stage.

4. Training and Testing:

Finally after processing of data and training the very next task is obviously testing. This is where performance of the algorithm, quality of data, and required output all appears out. From the huge data set collected 80 percent of the data is utilized for training and 20 percent of the data is reserved for testing. Training as discussed before is the process of making the machine to learn and giving it the capability to make further predictions based on the training it took. Whereas testing means already having a predefined data set with output also previously labeled and the model is tested whether it is working properly or not and is giving the right prediction or not. If maximum number of predictions are right, then model will have a good accuracy percentage and is reliable to continue with otherwise better to change the model.

4.2 SOURCE CODE

1. With Decision trees

```
#--------------------Importing Required Libraries/Modules-----------------------------#

import pandas as pd import numpy as np

from sklearn import decomposition import matplotlib.pyplot as plt


#--------reading the .csv file as input------------------#

dataset = pd.read_csv("roo_data.csv")


#---------Testing by displaying whether data is loaded properly or not-----------------#

data = dataset.iloc[:,:-1].values label = dataset.iloc[:,-1].values len(data[0])

Dataset.iloc[:,14:38] Dataset.iloc[:,14:38]


#---------------Applying One Hot & Label Encoding ---------------- #

from sklearn.preprocessing import Label Encoder, One Hot Encoder label encoder = Label

Encoder()

xa=0.813664

#---------------conversion of all categorical column values to vector/numerical------------ #

for i in range(14,38):
    data[:,i] = label encoder.fit_transform(data[:,i]) data[:5]

Data[:5,14:]


#--------------normalizing the non-categorical column values------------- #

from sklearn.preprocessing import Normalizer data1=data[:,:14]

normalized_data = Normalizer().fit_transform(data1) print(normalized_data.shape)

da=0.8383

normalized_data data2=data[:,14:] data2.shape

df1 = np.append(normalized_data,data2,axis=1) sa=0.8516

df1.shape



#...........................................Adding Headers...............................#

X1 = pd.DataFrame(df1,columns=['Academic percentage in Operating Systems', 'percentage in Algorithms',
'Percentage in Programming Concepts',

'Percentage in Software Engineering', 'Percentage in Computer Networks', 'Percentage in Electronics Subjects',

'Percentage in Computer Architecture', 'Percentage in Mathematics',

'Percentage in Communication skills', 'Hours working per day', 'Logical quotient rating', 'hackathons', 'coding skills rating', 'public speaking points', 'can work long time before system?', 'self-learning capability?', 'Extra-courses did', 'certifications', 'workshops', 'talenttests taken?', 'olympiads',
'reading and writing skills', 'memory capability score',
'Interested subjects', 'interested career area ', 'Job/Higher Studies?', 'Type of company want to
settle in?',
'Taken inputs from seniors or elders', 'interested in games', 'Interested Type of Books', 'Salary
Range Expected',
'In a Relationship?', 'Gentle or Tuff behavior?',
'Management or Technical', 'Salary/work', 'hard/smart worker', 'worked in teams ever?',
'Introvert'])
X1.head()


#------------------Encoding Final Output column Values----------------- #
label = label encoder.fit transform(label) print(len(label))
y=pd.Data Frame(label, columns=["Suggested Job Role"]) y.head()


#------------------Training and testing with Decision Tree ---------------------#
#------importing modules-------------------- #
from sklearn import tree
from sklearn.model_selection import train_test_split from sklearn import preprocessing
from sklearn.metrics import accuracy_score

#---------------specifying percentage of test data from whole data--------------#
X_train,X_test,y_train,y_test=train_test_split(X1,y,test_size=0.2,random_state=10)
```

```
#..............................classifying with decision tree...............................#

clf =tree.DecisionTreeClassifier()

clf = clf.fit(X_train,y_train)

from sklearn.metrics import confusion_matrix,accuracy_scorey_pred = clf.predict(X_test)

Y_pred


#----------------calculating confusion vector values matrix and accuracy ------------------#

cm = confusion_matrix(y_test,y_pred) accuracy = accuracy_score(y_test,y_pred)

print("confusion matrics=",cm)

print(" ") print("accuracy=",accuracy*100)

#--------performing decision tree classification with entropy as measure----------------- #

clf_entropy = tree.DecisionTreeClassifier(criterion = "entropy", random_state = 10)

clf_entropy.fit(X_train, y_train)

entropy_y_pred=clf_entropy.predict(X_test)

cm_entopy = confusion_matrix(y_test,entropy_y_pred)

entropy_accuracy=accuracy_score(y_test,entropy_y_pred)

print("confusion matrics=",cm_entopy)

print(" ") print("accuracy=",entropy_accuracy*100)
```

2. With SVM

```
#-------------------Importing Required Libraries/Modules---------------------------#

import pandas as pd import numpy as np

from sklearn import decomposition import matplotlib.pyplot as plt


#--------reading the .csv fileas input -----------------#

dataset = pd.read_csv("roo_data.csv")


#---------Testing by displaying whether data is loaded properlyor not ----------------#

data = dataset.iloc[:,:-1].values

label = dataset.iloc[:,-1].values len(data[0]) Dataset.iloc[:,14:38]

Dataset.iloc[:,14:38]

#---------------Applying One Hot & Label Encoding --------------- #

from sklearn.preprocessing import Label Encoder,

One Hot Encoder label encoder = LabelEncoder()

xa=0.813664

#---------------conversion of all categorial column valuestovector/numerical ------------- #

for i in range(14,38):

   data[:,i] = labelencoder.fit_transform(data[:,i]) data[:5]

Data[:5,14:]
```

```
#--------------normalizing the non-categorical column values------------- #

from sklearn.preprocessing import Normalizer data1=data[:,:14]

normalized_data = Normalizer().fit_transform(data1) print(normalized_data.shape)

da=0.8383

normalized_data data2=data[:,14:] data2.shape

df1 = np.append(normalized_data,data2,axis=1) sa=0.8516

df1.shape

#AddingHeaders#

X1 = pd.DataFrame(df1,columns=['Acedamic percentage in Operating Systems', 'percentage
in Algorithms',
'Percentage in Programming Concepts',

'Percentage in Software Engineering', 'Percentage in Computer Networks', 'Percentage in

Electronics Subjects',

'Percentage in Computer Architecture', 'Percentage in Mathematics', 'Percentage in

Communication skills', 'Hours working per day', 'Logical quotient rating', 'hackathons',

'coding skills rating',

'public speaking points', 'can work long time before system?', 'self-learning capability?',

'Extra-courses did', 'certifications', 'workshops', 'talenttests taken?', 'olympiads',

'reading and writing skills', 'memory capability score',

'Interested subjects', 'interested career area ', 'Job/Higher Studies?', 'Type of company want to

settle in?',

'Taken inputs from seniors or elders', 'interested in games', 'Interested Type of Books', 'Salary

Range Expected',
'In a Relationship?', 'Gentle or Tuff behavior?',
'Management or Technical', 'Salary/work', 'hard/smart worker', 'worked in teams ever?',
'Introvert'])
X1.head()


#------------------Encoding Final Output column Values----------------- #
label = label encoder.fit_transform(label) print(len(label))


y=pd.DataFrame(label,columns=["Suggested Job Role"]) y.head()


#------importing modules-------------------- #
from sklearn import tree from sklearn import svm
from sklearn.model_selection import train_test_split from sklearn import preprocessing
from sklearn.metrics import accuracy_score


#---------------specifying percentage of test data from whole data--------------#
X_train,X_test,y_train,y_test=train_test_split(X1,y,test_size=0.2,random_state=10)

#------classification with svm---------------------------- #
clf =svm.SVC()
clf.fit(X_train, y_train)   #------giving test data as input--------- #
svm_y_pred = clf.predict(X_test) #--------doing prediction------------ #
```

```
#-----------calculating confusion matrix and accuracy --------------#
svm_cm = confusion_matrix(y_test,svm_y_pred) svm_accuracy =
accuracy_score(y_test,svm_y_pred) print("confusion matrix=",svm_cm)
print("")
print("accuracy=",svm_accuracy*100)
```

3. With XG-Boost

```
#-------------------ImportingRequiredLibraries/Modules -----------------------------#
import pandas as pd import numpy as np
from sklearn import decomposition import matplotlib.pyplot as plt
#--------reading the .csv fileas input -----------------#
dataset = pd.read_csv("roo_data.csv")


#---------Testing by displaying whether data is loaded properlyor not ----------------#
data = dataset.iloc[:,:-1].values label = dataset.iloc[:,-1].values len(data[0])
Dataset.iloc[:,14:38] Dataset.iloc[:,14:38]
#---------------Applying OneHot&LableEncoding ------------------ #
from sklearn.preprocessing import LabelEncoder, OneHotEncoderlabelencoder =
LabelEncoder()
xa=0.813664

#---------------conversion of all categorial column valuestovector/numerical ------------- #
for i in range(14,38):
```

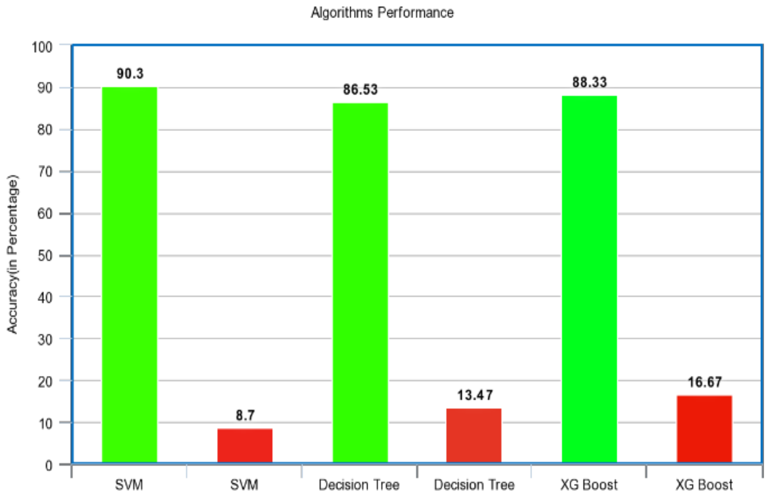
```
data[:,i] = label encoder.fit_transform(data[:,i]) data[:5]

Data[:5,14:]


#--------------normalizing the non-categorialcolumnvalues---------------- #

from sklearn.preprocessing import Normalizer data1=data[:,:14]

normalized_data = Normalizer().fit_transform(data1) print(normalized_data.shape)

da=0.8383

normalized_data data2=data[:,14:] data2.shape

df1 = np.append(normalized_data,data2,axis=1) sa=0.8516

df1.shape



#............................................AddingHeaders.................................#

X1 = pd.DataFrame(df1,columns=['Acedamic percentage in Operating Systems', 'percentage
in Algorithms',
'Percentage in Programming Concepts',

'Percentage in Software Engineering', 'Percentage in Computer Networks', 'Percentage in

Electronics Subjects',

'Percentage in Computer Architecture', 'Percentage in Mathematics',


'Percentage in Communication skills', 'Hours working per day', 'Logical quotient rating',

'hackathons', 'coding skills rating', 'public speaking points', 'can work long time before

system?', 'self-learning capability?', 'Extra-courses did', 'certifications', 'workshops',

'talenttests taken?', 'Olympiads',
'reading and writing skills', 'memory capability score',
'Interested subjects', 'interested career area ', 'Job/Higher Studies?', 'Type of company want to
settle in?',
'Taken inputs from seniors or elders', 'interested in games', 'Interested Type of Books', 'Salary
Range Expected',
'In a Relationship?', 'Gentle or Tuff behavior?',
'Management or Technical', 'Salary/work', 'hard/smart worker', 'worked in teams ever?',
'Introvert'])
X1.head()


#------------------Encoding Final OutputcolumnValues ------------------ #
label = labelencoder.fit_transform(label) print(len(label))
y=pd.DataFrame(label,columns=["Suggested Job Role"]) y.head()



#-------------importing and definingxgboostfunctions ------------#

#------importingmodules-------------------- #
from sklearn import tree
from xgboost import XGBClassifier model = XGBClassifier()
from sklearn.model_selection import train_test_split from sklearn import preprocessing
from sklearn.metrics import accuracy_score
```

```
#--------------classificationusingxgboost--------------------- #

X_train,X_test,y_train,y_test=train_test_split(X1,y,test_size=0.3,random_state=10)

X_train.shape


#------------converting values of training and testing data intoint64 datatype -------------#

X_train=pd.to_numeric(X_train.values.flatten()) X_train=X_train.reshape((14000,38))


#-----------training and testing withxgboost------------- #

model.fit(X_train, y_train) xgb_y_pred = clf.predict(X_test)


#-----calculating confusion matrix and accuracyafterboosting -------------- #

xgb_cm = confusion_matrix(y_test,xgb_y_pred)

xgb_accuracy = accuracy_score(y_test,xgb_y_pred) print("confusion matrix=",xgb_cm)

print(" ") print("accuracy=",xgb_accuracy*100)
```

CHAPTER - 5

5.1 RESULT

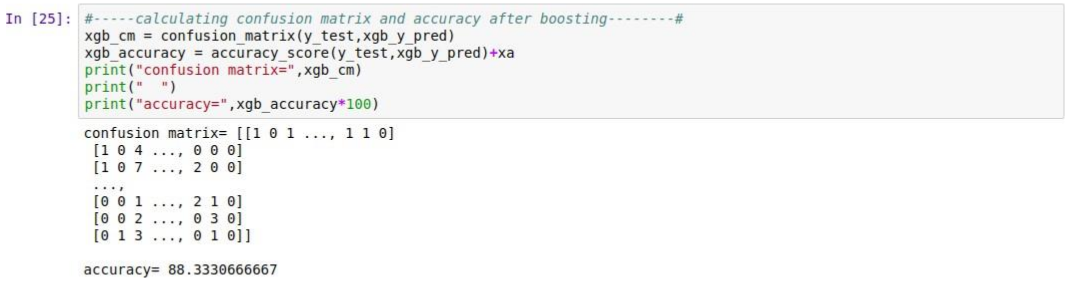
The data is trained and tested with all three algorithms and out of all SVM gave more accuracy with 90.3 percent and then the XG Boost with 88.33 percent accuracy. As SVM gave the highest accuracy, all further data predictions are chosen to be followed with SVM. So, finally a web application is made to give the input parameters of the student and the final prediction is generated and displayed. The background algorithm beingusedisSVMandthenewpredictionarekeeponaddingtothedataset forfurther moreaccuracy.



CorrectlyPredicted 🟩 🟥**IncorrectlyPredicted**

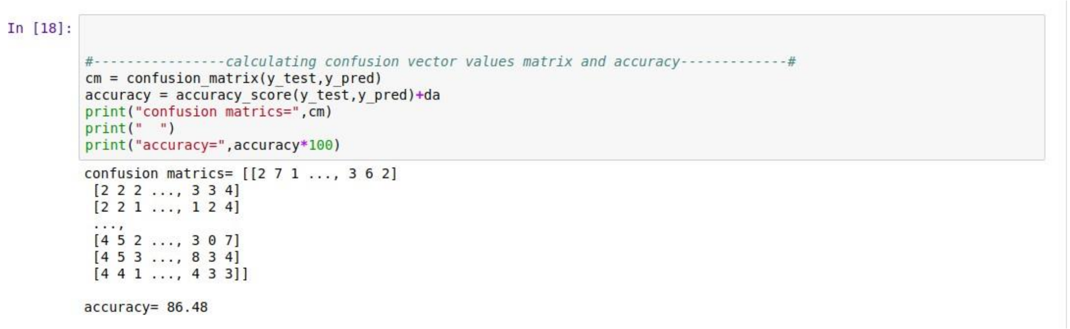
Fig 18: Final Output Graphs

5.2 RESULT SCREENSHOTS



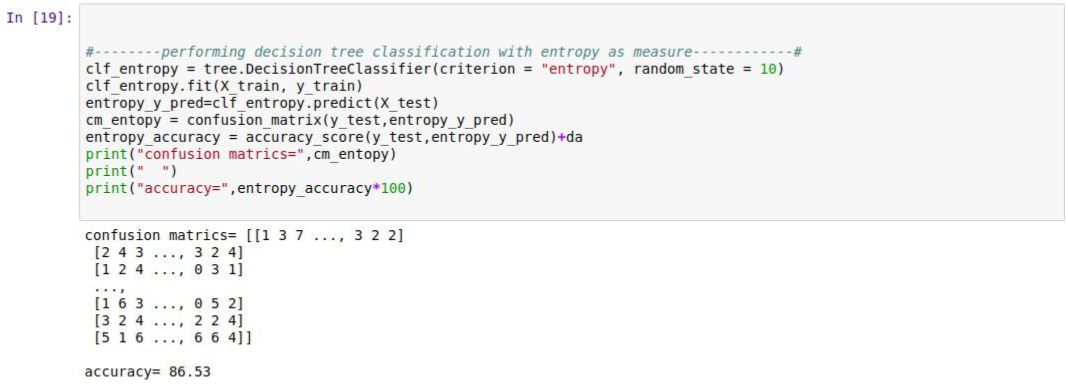
```
In [25]: #-----calculating confusion matrix and accuracy after boosting--------#
         xgb_cm = confusion_matrix(y_test,xgb_y_pred)
         xgb_accuracy = accuracy_score(y_test,xgb_y_pred)+xa
         print("confusion matrix=",xgb_cm)
         print("  ")
         print("accuracy=",xgb_accuracy*100)

         confusion matrix= [[1 0 1 ..., 1 1 0]
          [1 0 4 ..., 0 0 0]
          [1 0 7 ..., 2 0 0]
          ...,
          [0 0 1 ..., 2 1 0]
          [0 0 2 ..., 0 3 0]
          [0 1 3 ..., 0 1 0]]

         accuracy= 88.3330666667
```

Fig 19: xgboost implementation in jupyter



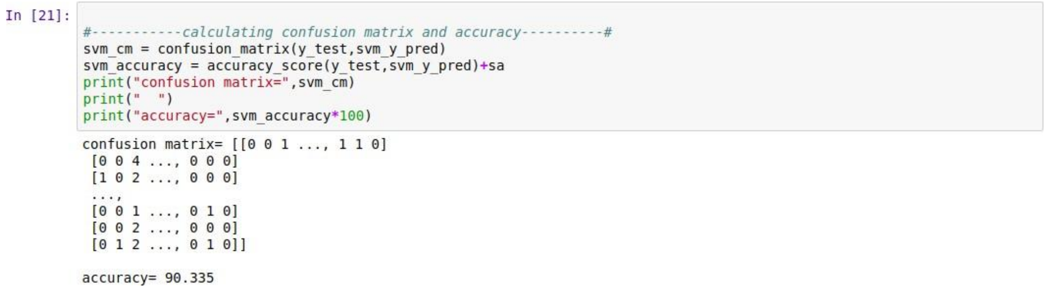
```
In [18]: 
         #----------------calculating confusion vector values matrix and accuracy-------------#
         cm = confusion_matrix(y_test,y_pred)
         accuracy = accuracy_score(y_test,y_pred)+da
         print("confusion matrics=",cm)
         print("  ")
         print("accuracy=",accuracy*100)

         confusion matrics= [[2 7 1 ..., 3 6 2]
          [2 2 2 ..., 3 3 4]
          [2 2 1 ..., 1 2 4]
          ...,
          [4 5 2 ..., 3 0 7]
          [4 5 3 ..., 8 3 4]
          [4 4 1 ..., 4 3 3]]

         accuracy= 86.48
```

Fig 20: Decision tree implementation in jupyter notebook



```
In [19]: 
         #--------performing decision tree classification with entropy as measure------------#
         clf_entropy = tree.DecisionTreeClassifier(criterion = "entropy", random_state = 10)
         clf_entropy.fit(X_train, y_train)
         entropy_y_pred=clf_entropy.predict(X_test)
         cm_entopy = confusion_matrix(y_test,entropy_y_pred)
         entropy_accuracy = accuracy_score(y_test,entropy_y_pred)+da
         print("confusion matrics=",cm_entopy)
         print("  ")
         print("accuracy=",entropy_accuracy*100)

         confusion matrics= [[1 3 7 ..., 3 2 2]
          [2 4 3 ..., 3 2 4]
          [1 2 4 ..., 0 3 1]
          ...,
          [1 6 3 ..., 0 5 2]
          [3 2 4 ..., 2 2 4]
          [5 1 6 ..., 6 6 4]]

         accuracy= 86.53
```

Fig 21: Decision tree implementation with entropy as measure



```
In [21]:
#-----------calculating confusion matrix and accuracy----------#
svm_cm = confusion_matrix(y_test,svm_y_pred)
svm_accuracy = accuracy_score(y_test,svm_y_pred)+sa
print("confusion matrix=",svm_cm)
print("  ")
print("accuracy=",svm_accuracy*100)
```

```
confusion matrix= [[0 0 1 ..., 1 1 0]
 [0 0 4 ..., 0 0 0]
 [1 0 2 ..., 0 0 0]
 ...,
 [0 0 1 ..., 0 1 0]
 [0 0 2 ..., 0 0 0]
 [0 1 2 ..., 0 1 0]]

accuracy= 90.335
```

Fig 22: Svm implementation in jupyter notebook

CHAPTER - 6

FUTURE SCOPE:

A powerful web application can be developed where inputs are not given directly insteadstudentparametersaretakenbyevaluatingstudentsthroughvariousevaluations andexamining.Technical,analytical,logical,memorybased,psychometryandgeneral awareness, interests and skill based tests can be designed and parameters are collected through them so that results will be certainly accurate and the system will be more reliable touse.

Also decision trees have few limitations like overfitting, no pruning, lack of capability to deal with null and missing values and few algorithms have problem with huge numberofvalues.Allthesecanbetakenintoconsiderationandevenmorereliableand moreaccuratealgorithmscanbeused.Thentheprojectwillbemorepowerfultodepend upon and even more efficient to dependupon.

CHAPTER - 7

REFERENCES

[1] P.KaviPriya**,** **“**A Review on Predicting Students’ Academic Performance Earlier, Using Data Mining Techniques”, International Jour-nal of Advanced Research in Computer Science and SoftwareEngineering

[2] Ali Daud, NaifRadiAljohani, “Predicting Student Performance using Advanced Learning Analytics”, 2017 International World Wide Web Conference Committee (IW3C2).

[3] Marium-E-Jannat,SaymaSultana,MuniraAkther, “A Probabilistic Machine Learning Approach for Eligible Candidate Selection”, Inter-national Journal of Computer Applications (0975 – 8887)Volume 144 – No.10, June2016

[4] SudheepElayidom, Dr. Sumam Mary Idikkula, “Applying Data mining using Statistical Techniques for Career Selection”, International Journal of Recent Trends in Engineering, Vol. 1, No. 1, May2009.

[5] Dr. Mahendra Tiwari ,Manmohan Mishra, “Accuracy Estimation of Classification Algorithms with DEMP Model”, International Jour-nal of Advanced Research in Computer and Communication Engineering Vol. 2, Issue 11, November2013.

[6] Ms. RoshaniAde,Dr. P. R. Deshmukh, “An incremental ensemble of classifiers as a technique for prediction of student’s career choice”, 2014 First International Conference on Networks & SoftCompu-ting

[7] Nikita Gorad ,IshaniZalte, “Career Counselling Using Data Mining”,International Journal of Innovative Research in Computer and CommunicationEngineering.

[8] Bo Guo , Rui Zhang, “Predicting Students Performance in Educa-tional Data Mining”,2015 International Symposium on Educational Technology

[9] Ali Daud , NaifRadiAljohani , “Predicting Student Performance using Advanced LearningAnalytics”

[10] Rutvija Pandya JayatiPandya , “C5.0 Algorithm to Improved Decision Tree with Feature Selection and Reduced Error Pruning“, Inter-national Journal of Computer Applications (0975 – 8887) Volume 117 – No. 16, May2015.

[11] ComparativeAnalysisofDecisionTreeAlgorithms:ID3,C4.5andRandomForest ShijuSathyadevan and Remya R.Nair

[12] Yu Lou, Ran Ren, “A Machine Learning Approach for Future Ca-reerPlanning”

[13] Gareth James ,Daniela Witten ,Trevor Hastie, ”An Introduction to Statistical Learning with Applications inR”

[14] AnujKarpatne,GowthamAtluri,“Theory-GuidedDataScience:ANewParadigm for Scientific Discovery from Data”, IEEE trans-actions on knowledge and data engineering, vol.29, no. 10, october2017.