

FOREST COVER TYPE PREDICTION

A project report submitted in partial fulfillment of the requirements for the
degree of

Bachelor of Technology

in

Electronics and Communication Engineering

by

V.R.N.S.Harshavardhan(S170305)

SK.Amjut Ali(S170679)

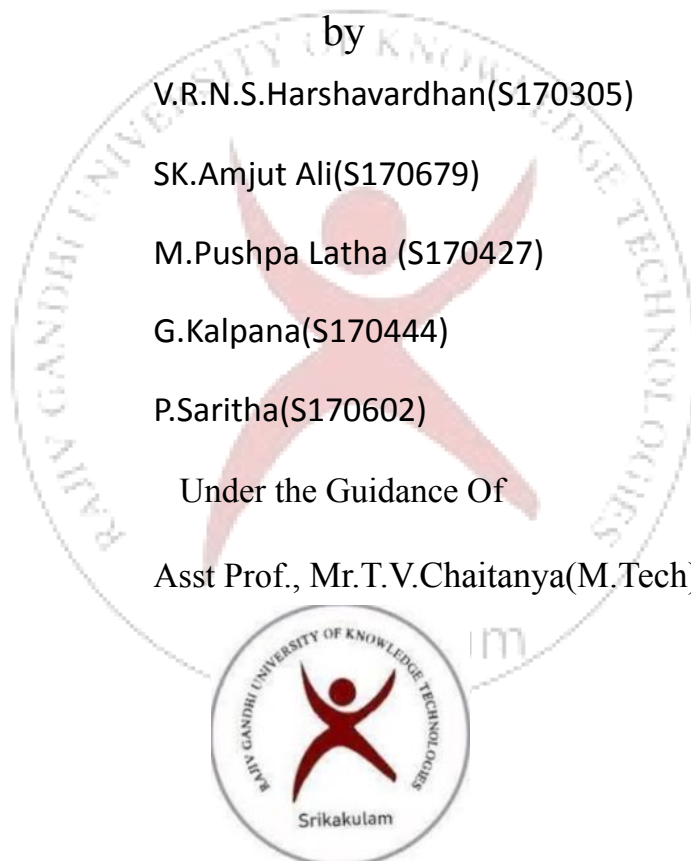
M.Pushpa Latha (S170427)

G.Kalpana(S170444)

P.Saritha(S170602)

Under the Guidance Of

Asst Prof., Mr.T.V.Chaitanya(M.Tech)



Department Of Electronics and Communication Engineering

Rajiv Gandhi University of Knowledge Technologies

SRIKAKULAM – 532410

September- 2022

Certificate

This is to certify that the work entitled “Forest Cover Type Prediction” is a bonafide record of authentic work carried out **by S170305 – V.R.N.S.Harshavardhan ,S170679 – SK.Amjut Ali ,S170427 – M.Pushpa Latha ,S170444– G.Kalpna ,S170602– P.Saritha** under my supervision and guidance for the partial fulfillment of the requirement of the award of the degree of Bachelor of Technology in the department of Electronics and Communication Engineering at RGUKT- SRIKAKULAM.

The results embodied in this work have not been submitted to any other university or institute for the award of any degree or diploma. This thesis, in our opinion, is worthy of consideration for the award of the degree of Bachelor of Technology in accordance with the regulations of the institute.

Mr.T.V.Chaitanya(M.Tech)

Assistant Professor,

Department of ECE,

RGUKT – Srikakulam

Head of the Department

Acknowledgement

With the profound respect and gratitude, We take the opportunity to convey our special thanks to our project guide **Mr.T.V.Chaitanya(M.Tech)**, Asst.Professor, Dept. of ECE for his excellence in supervision,encouragement,comments,suggestions,continuous guidance and support in completing this project work with great interest and he gave us the golden opportunity to do the wonderful project on the topic “**Forest Cover Type Prediction**”, which also helped us in doing a lot of research and We came to know about many new things,really thankful to them.

Electronics and Communication Engineering

IIIT – SRIKAKULAM

Date : - 29/09/2022

Place : - Nuzvid

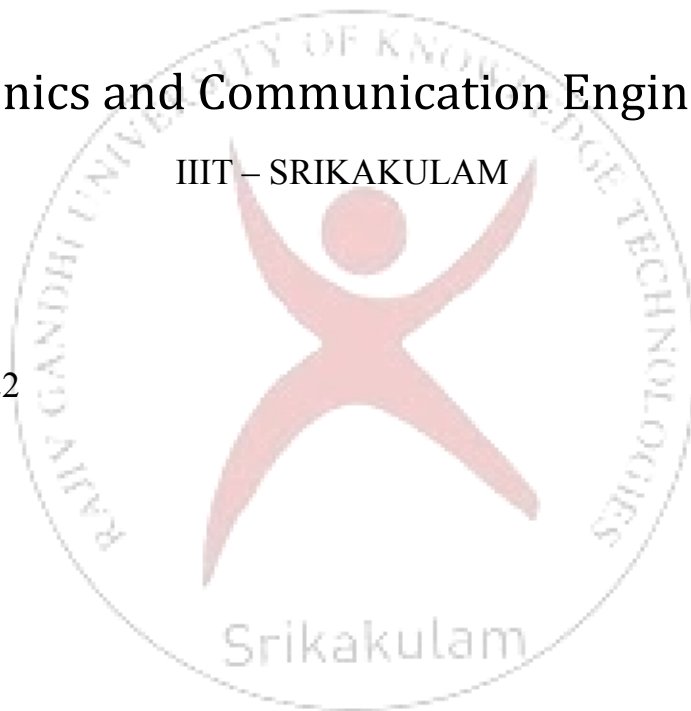
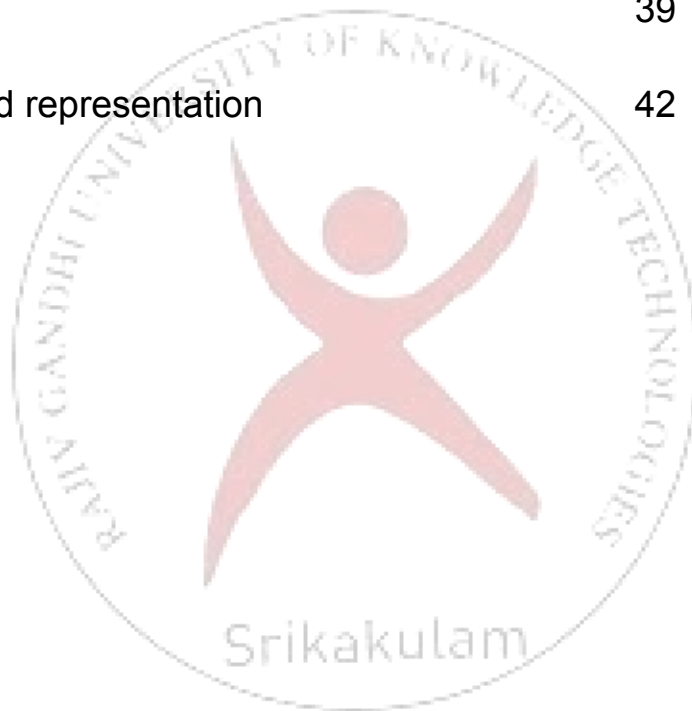


TABLE OF CONTENTS

Title	1
Certificate	2
Acknowledgement	3
Abstract	6
1.Introduction	7
2.Literature review	10
2.1.User characteristics	
2.2.Logistics regression model	
2.3.Decision tree	
3.Models	12
3.1.Logistic regression	
3.2.Steps in logistic regression	
3.3. Decision tree	
3.4Steps in Decision tree	
3.5.Random forest	
3.6.Steps in Random forest	
4.Deep learning	19
4.1Artificial neural networks	
4.2 Between deep learning and machine learning	

4.3 key differences	
5.Creating test data and train data	24
6.Methodologies	25
6.1. Data Set	
6.2. Features of data set	
7.Implementation	26
7.1.Preprocessing	
7.2. Data augmentation	
7.3.Understand, Clean and Format data	
7.4.EDA	
7.5.Feature engineering and selection	
7.6.Perform Hyperparameter tuning on the best model	
7.7. Interpret model results	
8.performance matrices	31
9.Classification accuracy	32

10. System requirements	33
11.What collab offers you	34
12.Python and its Libraries	35
13. Result and applications	36
14.Conclusion and future work	38
15.References	38
16.Annexure	39
17.Output and representation	42



Abstract

Forest is very important to humans in day to day life. Forest land is highly required for developing ecosystem management. Any changes that occur in the ecosystem should be carefully noticed to avoid further loss. This model is helpful in noticing the changes occurred due to heavy floods or any other calamities which affected the forest land.

The goal is to predict seven different cover types in four different wilderness areas of the Roosevelt National Forest of Northern Colorado with the best accuracy. By using machine learning algorithms we have to predict the cover type. To get better accuracy we are using deep learning algorithms like ANN (artificial neural networks) and apply this on data and predict the forest cover type. Predicting the cover type is veryful to protect the forest trees further. By using those properties, we train a combination of different machine learning algorithms using various machine learning methods and evaluate their performance on real world datasets. the most suitable forest tree species for defined area and landscape has been investigated in the paper. A set of classifiers is constructed in order to build relations between type of soil and other features of forest area and preferable species of trees. Results and conclusions of this paper can be used in processing of other forest cover tasks too. The same methods can be implemented in order to get the preferable tree species for different areas if there's enough data to solve these tasks with machine learning techniques and deep learning techniques. The approach is to predict the forest cover type using the cartographic variables like aspect, slope, soil type, wilderness area etc. Various Data mining techniques such as decision trees, random forest, logistic regression trees are used for prediction of the forest cover type.

Keywords: Machine Learning, Logistic Regression, Decision tree, Random forest, Deep Learning, Artificial Neural Network(ANN), Cover type

Introduction

By using the Machine Learning concept, here we try to solve the problem and reach the goal or objective. Here in this project we have to take different cover types and wilderness areas of the **Roosevelt National Forest(northern Colorado)**. These areas represent forests with minimal human-caused disturbances. So forest covers are more a result of ecological processes than forest management practices. Machine learning is a scientific discipline that explores the construction and study of algorithms that can learn from data. Machine learning effectively learns from a training dataset and apply transformation on the test data to get the estimated results. In this new learning algorithms have emerged (e.g. Logistic regression, Decision trees and Random forest, Artificial neural networks) that has given excellent performance, accuracy and results. It consumes less time and human activity with automatic techniques that improves performance and accuracy and efficiency by discovering and exploiting regularities in the dataset using training and testing. Machine learning is mostly based on statistics. There are various algorithms which have connections with the statistics.

Machine Learning: Machine learning is a subset, an application of Artificial Intelligence (AI) that offers the ability to the system to learn and improve from experience without being programmed to that level. Machine Learning uses data to train and find accurate results. Machine learning focuses on the development of a computer program that accesses the data and uses it to learn from itself. There are various applications of machine learning including medical, bioinformatics, search engines, credit card fraud, natural language processing, speech and handwriting recognition, robotics and game playing.

Terminologies in Machine learning: ,Model ,Feature, Label(Target), Training and Test data, Prediction.

Deep Learning: Deep Learning is a subset of Machine Learning where the artificial neural network and the recurrent neural network come in relation. The algorithms are created exactly just like machine learning but it consists of many more levels of algorithms. All these networks of the algorithm are together called the artificial neural network. In much simpler terms, it replicates just like the human brain as all the neural networks are connected in the brain, which is exactly the concept of deep learning. It solves all the complex problems with the help of algorithms and its process.

In this article, we will explain how to approach a multi-class classification supervised machine learning (ML) problem (or project) with an end-to-end workflow.

Supervised: The model or algorithm is presented with example inputs and their desired outputs and then finding patterns and connections between the input and the output. The goal is to learn a general rule that maps inputs to outputs. The training process continues until the model achieves the desired level of accuracy on the training data. Features (variables to generate predictions) and target (the variable to be determined) are available in the data set.

Two most common use cases of Supervised learning are:

Classification: Inputs are divided into two or more classes, and the learner must produce a model that assigns unseen inputs to one or more (multi-label classification) of these classes and predict whether or not something belongs to a particular class. This is typically tackled in a supervised way. Classification models can be categorized in two groups: Binary classification and Multiclass Classification.

Regression: It is also a supervised learning problem, that predicts a numeric value and outputs are continuous rather than discrete. For example, predicting the stock prices using historical data.

Multi-class classification: In machine learning, multiclass or multinomial classification is the problem of classifying instances into one of three or more classes.

Four wilderness areas are:

1: Rawah

2: Neota

3: Comanche Peak

4: Cache la Poudre

Seven categories numbered from 1 to 7 in the `Cover_Type` column, to be classified:

1: Spruce/Fir

2: Lodgepole Pine

3: Ponderosa Pine

4: Cottonwood/Willow

5: Aspen

6: Douglas-fir

7: Krummholz

The forest cover dataset consists of predominant kinds of tree cover and various attributes which are cartographic and we used different algorithms for predicting the cover type using cartographic variables. The algorithms used are firstly classification. Secondly decision trees are used which are produced by algorithms that identify various ways of splitting a data into branch-like segments . The output is predicted by each and every iteration in the tree. Thirdly Random forest algorithms which are used for classification and prediction in which multiple decision tree are made by averaging at every iteration and the values are classified and predicted.

End-to-end Machine Learning Workflow Steps: to classify cover types and reply to the initiating question, where to find fantastic trees and how to detect them, the below steps will be followed:

1. Understand, Clean and Format Data
2. Exploratory Data Analysis
3. Feature Engineering & Selection
4. Compare Several Machine Learning Models
5. Perform Hyperparameter Tuning on the Best Model
6. Interpret Model Results
7. Evaluate the Best Model with Test Data (replying the initiating question)
8. Summary & Conclusions

2.Literature review

The attempt to explore the space of parameters and common variations for each learning algorithm as thoroughly as it is computationally feasible. This section summarizes the parameters used for each learning algorithm.

2.1.User Characteristics: It tells about the linear and multiple and multivariate regression which is an important analysis tool in statistics and machine learning. It aims to create the linear relationship between two sets of variables, namely the dependent/response and the independent/predictor variables, and then to predict the values of the dependent variables given new values of independent variables. In recent decades, regression analysis has been widely used for prediction and forecasting, and intersects much with the field of pattern recognition and machine learning. The multivariate logistic regression is another important because this model analysis is a formula used to predict the relationships between dependent and independent variables. It calculates the probability of something happening depending on multiple sets of variables. This is a common classification algorithm used in data science and machine learning.

2.2.Logistic regression model: Distance measures the euclidean distance between each cell and the nearest of a set of target features. It is one of the classification model used to Data Pre-processing step fitting, Logistic Regression to the Training set ,predicting the test result,test accuracy of the result(Creation of Confusion matrix),visualizing the test set result.Logistic regression works by measuring the relationship between the dependent variable (what we want to predict) and one or more independent variables (the features). It does this by estimating the probabilities with the help of its underlying logistic function.

2.3.Decision tree: In decision tree is another approach for classification. Decision trees are created by algorithms that distinguish various ways of dividing a data set into branch-like sections. These sections form an inverted decision tree that starts with a root node at the tip of the tree. The aim of analysis is reflected in this root node as a simple, linear display in the decision tree interface. The epithet of the field of data that is the object of analysis is usually exposed, along with the spread or distribution of the values that are contained in that area. Decision trees are the form which consist of multiple variable with multiple effects analyses. This multiple variables allow us to predict, explain, describe, or classify a target or outcome. Decision trees attempt to see a solid relationship between input values and the target values in a group of observations that form

a data set. When a set of input values is identified as possessing a solid relationship with a target value, then all of these values are grouped in a bin that becomes a branch of the decision tree. These groupings are defined by the observed shape of the kinship between the bin values and the object.

It gives importance of applying random forest on dataset. Random forest is statistical method for classification and decision–tree based supervised learning algorithm. Its algorithm is an ensemble classification which is unsurpassable in accuracy among current data mining algorithms. It has been used for Microarray cancer , android malware , and intrusion detection. The random forest consists of many individual trees. Each tree votes on an overall classification for the given set of data and the random forest algorithm chooses the individual classification with the most ballots. There are two different sources of randomness in Random Forests: random training set (bootstrap) and random selection of properties.



Models

Machine Learning Models

1. Logistic Regression:

Logistic regression is an example of supervised learning. It is used to calculate or predict the probability of a binary (yes/no) event occurring. It is a statistical analysis method to predict a binary outcome, such as yes or no, based on prior observations of a data set. Logistic regression model predicts a dependent data variable by analyzing the relationship between one or more existing independent variables. Logistic Regression is much similar to the Linear Regression except for how they are used. Linear Regression is used for solving Regression problems, whereas Logistic regression is used for solving the classification problem. In Logistic regression, instead of fitting a regression line, we fit an "S" shaped logistic function, which predicts two maximum values (0 or 1). Logistic Regression is a significant machine learning algorithm because it has the ability to provide probabilities and classify new data using continuous and discrete data sets. Logistic Regression can be used to classify the observations using different types of data and can easily determine the most effective variables used for the classification.

Logistic Function (Sigmoid function): The sigmoid function is a mathematical function used to map the predicted values to probabilities. It maps any real value into another value within the range of 0 and 1. The value of logistic regression must be between 0 and 1, so it forms a curve like "S" form. The S-form curve is called sigmoid function or logistic function. In logistic regression, we use the concept of threshold value. Which defines the probability of either 0 or 1. Such as values above the threshold value it ends to 1, and a value below the threshold values tends to 0.

Types of logistic regression:

Binomial: In binomial Logistic regression, there can be only two possible types of the dependent variables, such as 0 or 1, Pass or Fail, etc.

Multinomial: In multinomial Logistic regression, there can be 3 or more possible unordered types of the dependent variable, such as "cat", "dogs", or "sheep".

Ordinal: In ordinal Logistic regression, there can be 3 or more possible ordered types of dependent variables, such as "low", "Medium", or "High".

Steps in Logistic Regression:

- 1.Data pre-processing step
- 2.Fitting Logistic Regression to the Training Set
- 3.Predicting the test result
- 4.Test Accuracy of the result(creation of confusion matrix)
- 5.Visulaizing the test set result

1.Data pre-processing step: In this step we will prepare the data,so that we can use it in our code efficiently.

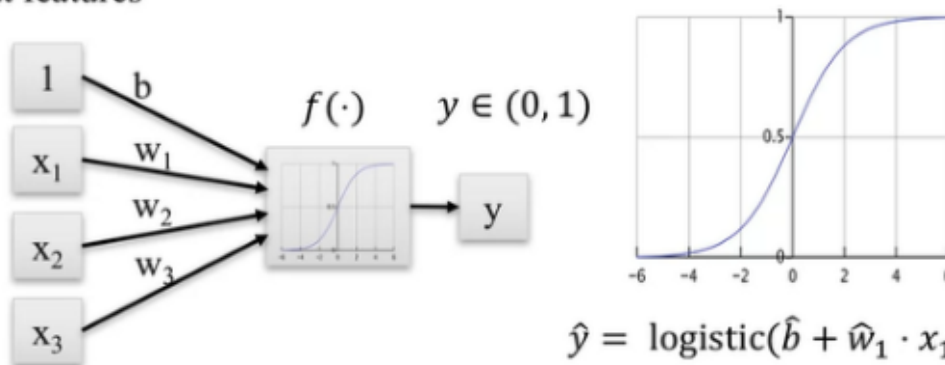
2. Fitting Logistic Regression to the Training set: We have well prepared our dataset, and now we will train the dataset using the training set. For providing training or fitting the model to the training set, we will import the LogisticRegression class of the sklearn library.After importing the class, we will create a classifier object and use it to fit the model to the logistic regression.

3. Predicting the Test Result: Our model is well trained on the training set, so we will now predict the result by using test set data.

4.Test Accuracy of the result: Now we will create the confusion matrix here to check the accuracy of the classification. To create it, we need to import the confusion_matrix function of the sklearn library. After importing the function, we will call it using a new variable cm. The function takes two parameters, mainly y_true(the actual values) and y_pred (the targeted value returned by the classifier).

5. Visualizing the training set result: Finally, we will visualize the training set result. to visualize the result, we will use the ListedColormap class of the matplotlib library.

Input features



$$\hat{y} = \text{logistic}(\hat{b} + \hat{w}_1 \cdot x_1 + \dots \hat{w}_n \cdot x_n)$$

$$= \frac{1}{1 + \exp[-(\hat{b} + \hat{w}_1 \cdot x_1 + \dots \hat{w}_n \cdot x_n)]}$$

2. Decision Tree:

Decision Tree is a Supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome. In a Decision tree, there are two nodes, which are the Decision Node and Leaf Node. Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches. The decisions or the test are performed on the basis of features of the given dataset. It is a graphical representation for getting all the possible solutions problem/decision based on given conditions. It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure. In order to build a tree, we use the CART algorithm, which stands for Classification and Regression Tree algorithm. A decision tree simply asks a question, and based on the answer (Yes/No), it further splits the tree into subtrees. A decision tree contains categorical data as well as numerical data. Decision Trees usually mimic human thinking ability while making a decision, so it is easy to understand. The logic behind the decision tree can be easily understood because it shows a tree-like structure. Steps to follow the decision tree is, Begin the tree with the root node, says S, which contains the complete dataset. Find the best attribute in the dataset using Attribute Selection Measure (ASM). Divide the S into subsets that contain possible values for the best attributes. Generate the decision tree node, which contains the best attribute. Recursively make new decision trees using the subsets of the dataset created in step -3. Continue this process until a stage is reached where you cannot further classify the nodes and called the final node as a leaf

node. While implementing a Decision tree, the main issue arises that how to select the best attribute for the root node and for sub-nodes. So, to solve such problems there is a technique which is called as Attribute selection measure or ASM. By this measurement, we can easily select the best attribute for the nodes of the tree. There are two popular techniques for ASM, which are:

Information Gain

Gini Index

Information Gain: Information gain is the measurement of changes in entropy after the segmentation of a dataset based on an attribute. It calculates how much information a feature provides us about a class. According to the value of information gain, we split the node and build the decision tree. A decision tree algorithm always tries to maximize the value of information gain, and a node/attribute having the highest information gain is split first. It can be calculated using the below formula:

Information Gain = Entropy(s) - [(weighted avg) * Entropy(each feature)]

Gini Index: Gini index is a measure of impurity or purity used while creating a decision tree in the CART (Classification and Regression Tree) algorithm. An attribute with the low Gini index should be preferred as compared to the high Gini index. It only creates binary splits, and the CART algorithm uses the Gini index to create binary splits. Decision tree is simple to understand as it follows the same process which a human follows while making any decision in real-life. It can be very useful for solving decision-related problems. It helps to think about all the possible outcomes for a problem. There is less requirement of data cleaning compared to other algorithms. The decision tree contains lots of layers, which makes it complex. It may have an overfitting issue, which can be resolved using the Random Forest algorithm. For more class labels, the computational complexity of the decision tree may increase.

Python Implementation of Decision Tree: Now we will implement the Decision tree using Python.

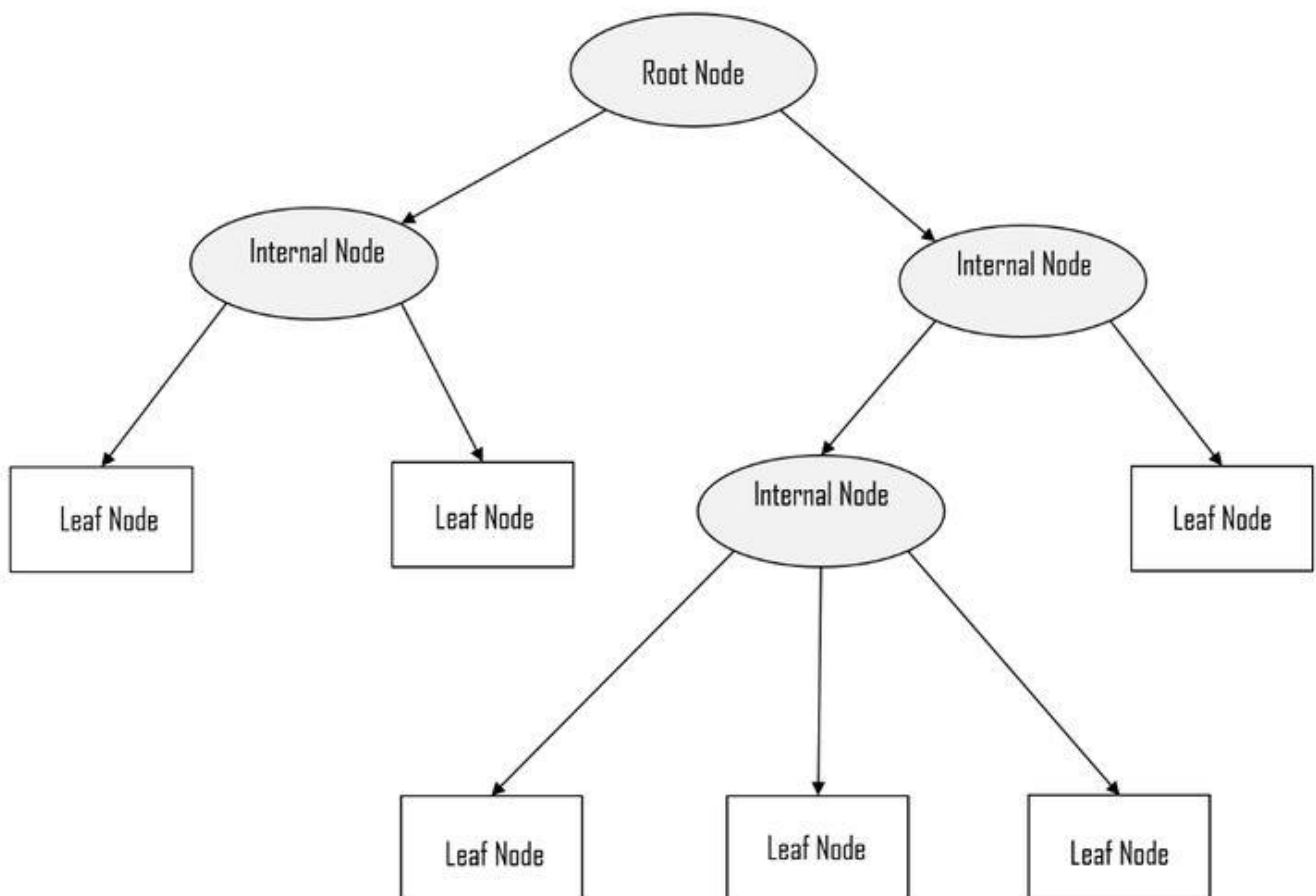
For this, we will use the dataset "user_data.csv," which we have used in previous classification models.

Some steps are following

1. Data pre-processing step
2. Fitting a Decision tree algorithm to the training set
3. Predicting the test result
4. Test Accuracy of the result
5. Visualizing the test set result

Data pre- processing is prepare the data:

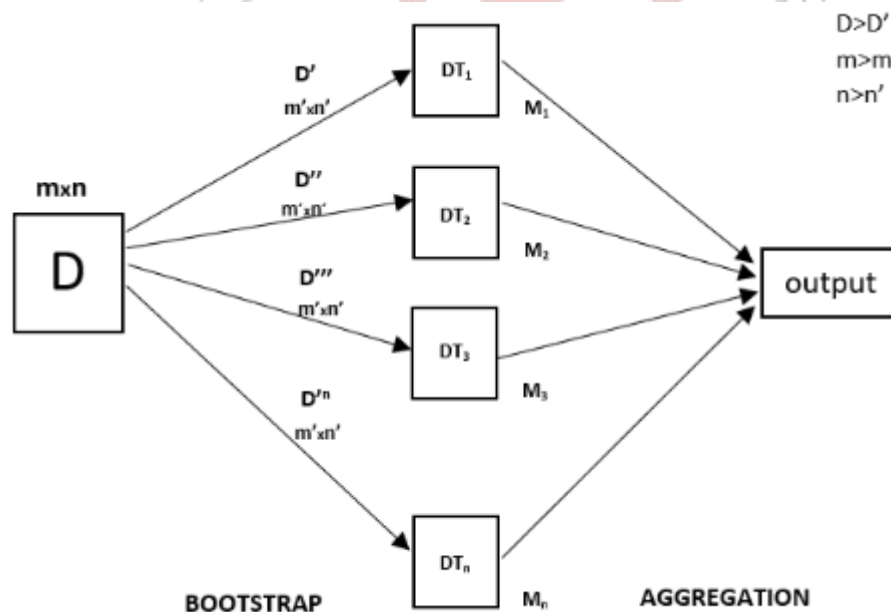
Fitting a Decision-Tree algorithm to the Training set: Now we will fit the model to the training set. For this, we will import the DecisionTreeClassifier class from sklearn.tree.
predict the test results:predict the test set result,we will create a new prediction vector y_pred.
Test accuracy of the result:need to use confusion matrix
Visualizing the training set result:Plot the graph for the decision tree classifier.



3.Random Forest:

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.

"Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset." Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output. The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting. Random forest combines multiple trees to predict the class of the dataset, it is possible that some decision trees may predict the correct output, while others may not. But together, all the trees predict the correct output. Therefore, below are two assumptions for a better Random forest classifier: There should be some actual values in the feature variable of the dataset so that the classifier can predict accurate results rather than a guessed result. The predictions from each tree must have very low correlations.



why we should use the Random Forest algorithm is, It takes less training time as compared to other algorithms. It predicts output with high accuracy, even for the large dataset it runs efficiently. It can also maintain accuracy when a large proportion of data is missing. Random Forest works in two-phase first is to create the random forest by combining N decision tree, and second is to make predictions for each tree created in the first phase. The Working process can be explained in the below steps:

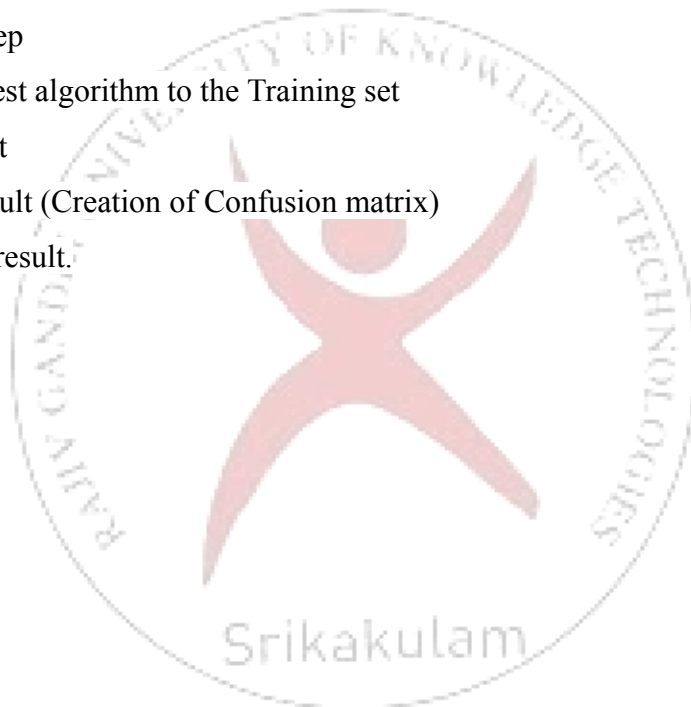
1. Select random K data points from the training set.
2. Build the decision trees associated with the selected data points (Subsets).

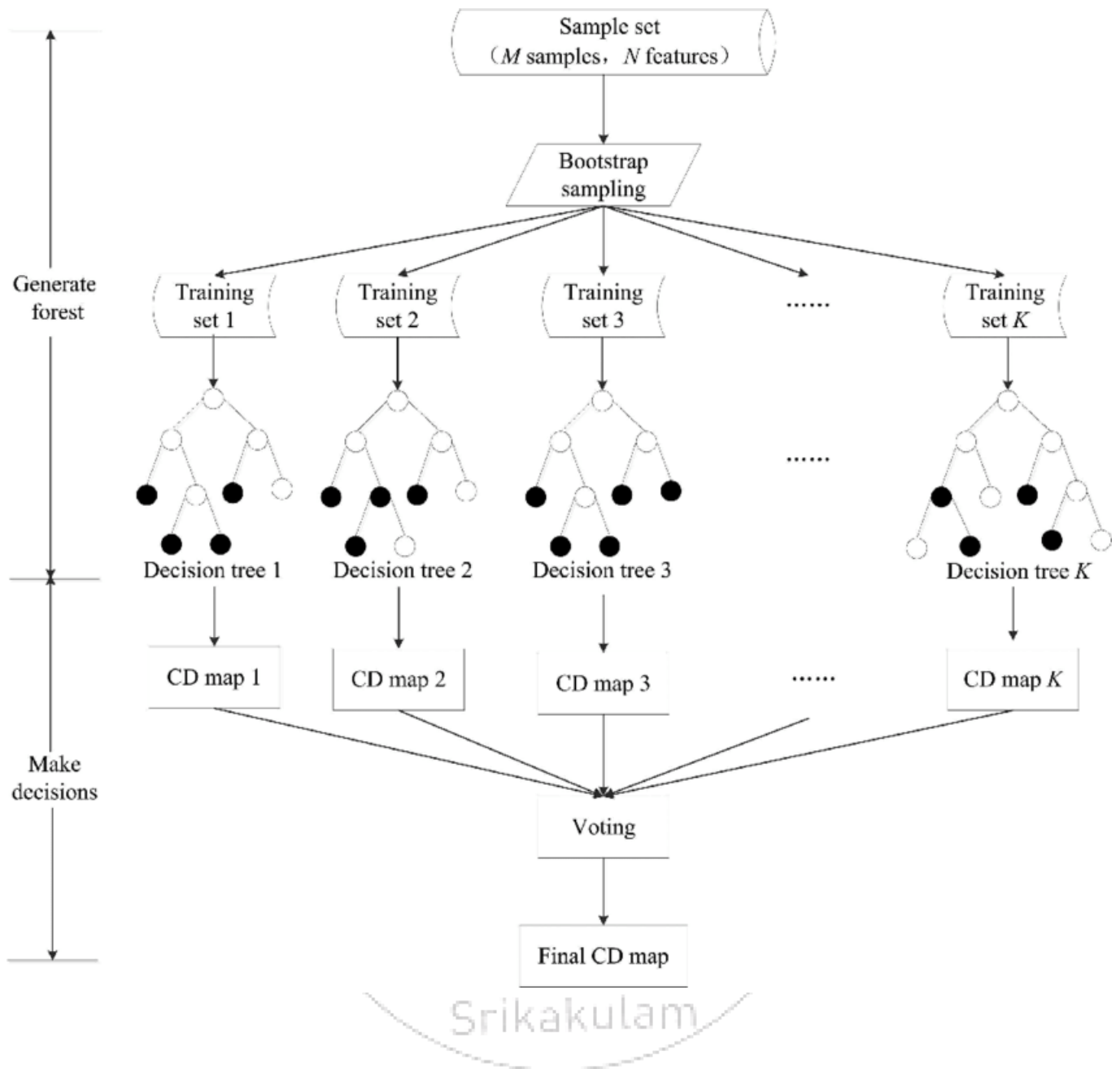
3. Choose the number N for decision trees that you want to build.
4. Repeat Step 1 & 2.
5. For new data points, find the predictions of each decision tree, and assign the new data points to the category that wins the majority votes.

Random Forest is capable of performing both Classification and Regression tasks. It is capable of handling large datasets with high dimensionality. Although random forest can be used for both classification and regression tasks, it is not more suitable for Regression tasks. Applications of Random Forest: Banking, Medicine, Land use, Marketing Python Implementation of Random Forest Algorithm. Now we will implement the Random Forest Algorithm tree using Python. For this, we will use the same dataset "user_data.csv".

Implementation Steps are:

1. Data Pre- processing step
2. Fitting the Random forest algorithm to the Training set
3. Predicting the test result
4. Test accuracy of the result (Creation of Confusion matrix)
5. Visualizing the test set result.





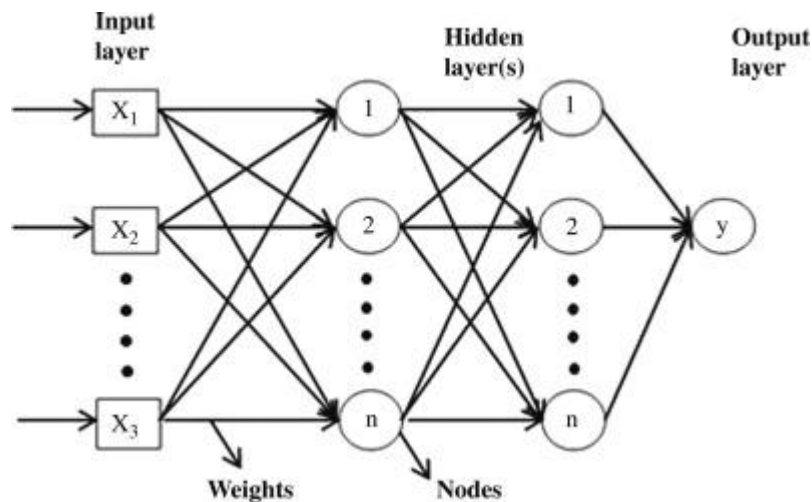
Deep Learning Model:

4. Artificial neural networks:

Artificial Neural Network(ANN) uses the processing of the brain as a basis to develop algorithms that can be used to model complex patterns and prediction problems. The term "Artificial Neural Network" is derived from Biological neural networks that develop the structure of the human brain. Similar to the human brain that has neurons interconnected to one another, artificial neural networks also have neurons that are interconnected to one another in various layers of the networks. These neurons are known as nodes.

The architecture of an artificial neural network:

To understand the concept of the architecture of an artificial neural network, we have to understand what a neural network consists of. In order to define a neural network that consists of a large number of artificial neurons, which are termed units arranged in a sequence of layers. Let us look at various types of layers available in an artificial neural network.



Input Layer: As the name suggests, it accepts inputs in several different formats provided by the programmer.

Hidden Layer: The hidden layer presents in-between input and output layers. It performs all the calculations to find hidden features and patterns.

Output Layer: The input goes through a series of transformations using the hidden layer, which finally results in output that is conveyed using this layer.

The artificial neural network takes input and computes the weighted sum of the inputs and includes a bias. This computation is represented in the transfer function. It determines the weighted total is passed as an input to an activation function to produce the output.

Advantages of Artificial Neural Network:

Parallel processing capability: Artificial neural networks have a numerical value that can perform more than one task simultaneously.

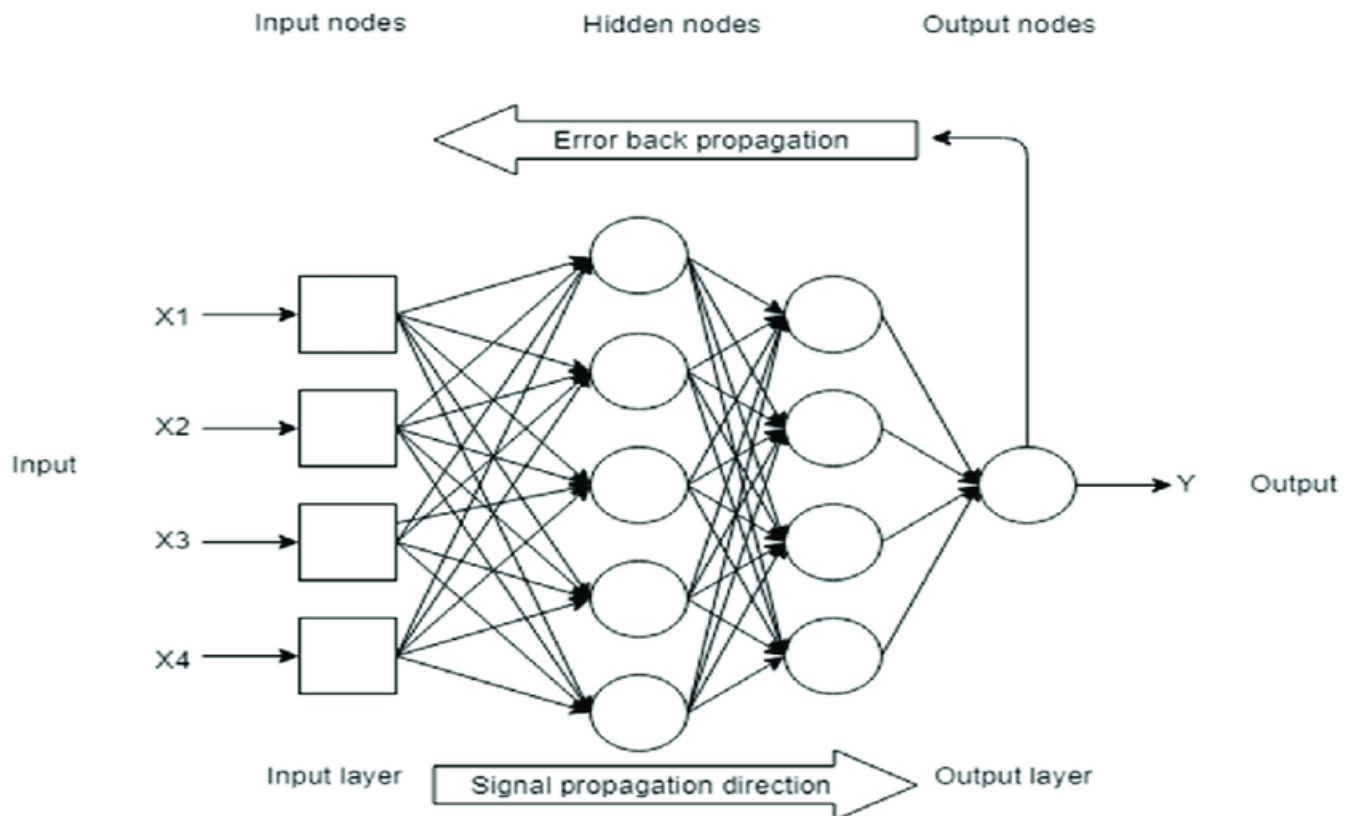
Storing data on the entire network: Data that is used in traditional programming is stored on the whole network, not on a database. The disappearance of a couple of pieces of data in one place doesn't prevent the network from working.

Capability to work with incomplete knowledge: After ANN training, the information may produce output even with inadequate data. The loss of performance here relies upon the significance of missing data.

Having a memory distribution: For ANN to be able to adapt, it is important to determine the examples and to encourage the network according to the desired output by demonstrating these

examples to the network. The succession of the network is directly proportional to the chosen instances, and if the event can't appear to the network in all its aspects, it can produce false output. Having fault tolerance: Extortion of one or more cells of ANN does not prohibit it from generating output, and this feature makes the network fault-tolerance.

Applications of Artificial Neural Networks are image recognition, face recognition, medical diagnosis. How do artificial neural networks work is Artificial Neural Network can be best represented as a weighted directed graph, where the artificial neurons form the nodes. The association between the neuron outputs and neuron inputs can be viewed as the directed edges with weights. The Artificial Neural Network receives the input signal from the external source in the form of a pattern and image in the form of a vector. These inputs are then mathematically assigned by the notations $x(n)$ for every n number of inputs. Afterward, each of the input is multiplied by its corresponding weights (these weights are the details utilized by the artificial neural networks to solve a specific problem). In general terms, these weights normally represent the strength of the interconnection between neurons inside the artificial neural network. All the weighted inputs are summarized inside the computing unit. If the weighted sum is equal to zero, then bias is added to make the output non-zero or something else to scale up to the system's response. Bias has the same input, and weight equals to 1. Here the total of weighted inputs can be in the range of 0 to positive infinity. Here, to keep the response in the limits of the desired value, a certain maximum value is benchmarked, and the total weighted inputs are passed through the activation function. The activation function refers to the set of transfer functions used to achieve the desired output. There is a different kind of activation function, but primarily either linear or non-linear sets of functions. Some of the commonly used sets of activation functions are the Binary, linear, and Tan hyperbolic sigmoidal activation functions.



Between Machine Learning and Deep Learning

Machine learning offers a variety of techniques and models you can choose based on your application, the size of data you're processing, and the type of problem you want to solve. A successful deep learning application requires a very large amount of data (thousands of images) to train the model. Deep learning is a type of machine learning, which is a subset of artificial intelligence. Machine learning is about computers being able to think and act with less human intervention; deep learning is about computers learning to think using structures modeled on the human brain. Machine learning requires less computing power; deep learning typically needs less ongoing human intervention. Deep learning can analyze images, videos, and unstructured data in ways machine learning can't easily do. Every industry will have career paths that involve machine and deep learning.

Key differences between machine learning and deep learning

While there are many differences between these two subsets of artificial intelligence, here are five of the most important:

1. Human Intervention

Machine learning requires more ongoing human intervention to get results. Deep learning is more complex to set up but requires minimal intervention thereafter.

2. Hardware

Machine learning programs tend to be less complex than deep learning algorithms and can often run on conventional computers, but deep learning systems require far more powerful hardware and resources. This demand for power has driven has meant increased use of graphical processing units. GPUs are useful for their high bandwidth memory and ability to hide latency (delays) in memory transfer due to thread parallelism (the ability of many operations to run efficiently at the same time.)

3. Time

Machine learning systems can be set up and operate quickly but may be limited in the power of their results. Deep learning systems take more time to set up but can generate results instantaneously (although the quality is likely to improve over time as more data becomes available).

4.Approach

Machine learning tends to require structured data and uses traditional algorithms like linear regression. Deep learning employs neural networks and is built to accommodate large volumes of unstructured data.

5. Applications

Machine learning is already in use in your email inbox, bank, and doctor's office. Deep learning technology enables more complex and autonomous programs, like self-driving cars or robots that perform advanced surgery.

Here in this project we are using the both machine learning and deep learning techniques. But compare to machine learning algorithms deep learning technique gives better accuracy. So that we prefer ANN(Artificial neural networks) over the Random forest algorithm. Initially we apply the Machine Learning Algorithms to predict the cover type. By using The logistic regression we get the accuracy but using decision trees we get more accuracy compared to logistic and again by using random forest we get accuracy more than decision trees. Again we using the deep learning technique ANN we get more accuracy compare to above algorithms. So here we prefer Deep learning.

Creating Test Data and Train data

split the the data set into two sets: a training set and a testing set.

- *train* the model using the training set.
- *test* the model using the testing set.

Train the model means *create* the model.

Test the model means test the accuracy of the model.

- 1.Start the Data
- 2.Split the data into train or test
- 3.Display the training set
- 4.Display the test data set
- 5.fit the dataset

By using the syntax we can split the data ,fitting the data set into the model.

Methodology

Dataset:

One of the crucial drawbacks of the present study in the area of Forest cover type Prediction is a major drop in classification performance of the models on real images taken in fields compared to the images from a controlled environment. The reason for this is that there is a lack of large public datasets of forest cover types and most of the present achievements are based on the Forest cover type Prediction.

In our project, object detection algorithms were explored in the forest cover type detection task.

This dataset includes information on tree type, shadow coverage, distance to nearby landmarks (roads etcetera), soil type, and local topography, horizontal distance to hydrology, vertical distance to hydrology.

Features of the dataset:

Dataset Description Table:

Data field	Description
elevation	Elevation in meters
aspect	Aspect in degrees azimuth
slope	Slope in degrees
horz_hydro	Horz Dist to nearest surface water features
vert_hydro	Vert Dist to nearest surface water features
horz_road	Horz Dist to nearest roadway
hillshade_9am	Hillshade index at 9am, summer solstice (0 to 255 index)
hillshade_noon	Hillshade index at noon, summer solstice (0 to 255 index)
hillshade_3pm	Hillshade index at 3pm, summer solstice (0 to 255 index)
horz_fire	Horz Dist to nearest wildfire ignition points
wild	Wilderness area designation (4 binary columns)
soil_type	Soil Type designation (40 binary columns)

Choice of training-testing set distribution:

The train-test split procedure is used to estimate the performance of machine learning

algorithms when they are used to make predictions on data not used to train the model. It is a fast and easy procedure to perform, the results of which allow you to compare the performance of machine learning algorithms for your predictive modeling problem.

Train: 80%, Test: 20%



Implementation

■ **Preprocessing:**

Preprocessing is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model. Real-world data generally contains noises, and missing values, and may be in an unusable format that cannot be directly used for machine learning models. Data preprocessing is a required task for cleaning the data and making it suitable for a machine learning model which also increases the accuracy and efficiency of a machine learning model.

● **Data augmentation:**

Data augmentation is a strategy that enables practitioners to significantly increase the diversity

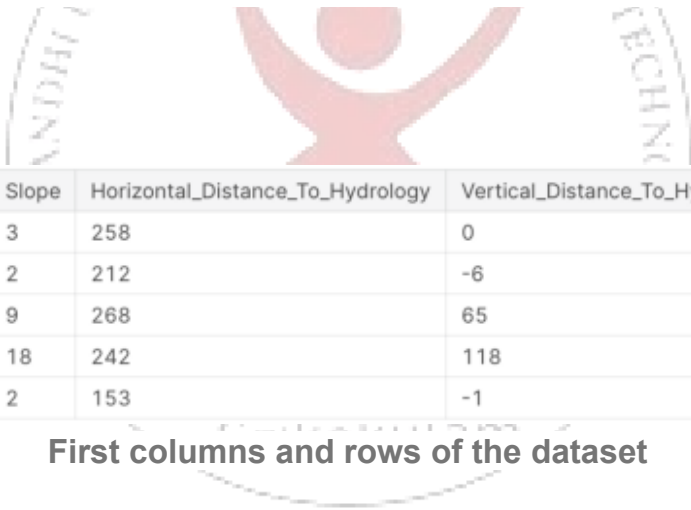
of data available for training models, without actually collecting new data. Data augmentation techniques such as cropping, padding, and horizontal flipping are commonly used to train large neural networks. It is also the process of modifying, or “augmenting” a dataset with additional data. This additional data can be anything from images to text, and its use in machine learning algorithms helps improve their performance.

1. Understand, Clean and Format Data

Let's load the training data and create `trees` data frame:

```
trees = pd.read_csv("/kaggle/input/learn-together/train.csv")
```

I always find it useful looking at the first and last rows:



	Id	Elevation	Aspect	Slope	Horizontal_Distance_To_Hydrology	Vertical_Distance_To_Hydrology	Horizontal_Distance_T
0	1	2596	51	3	258	0	510
1	2	2590	56	2	212	-6	390
2	3	2804	139	9	268	65	3180
3	4	2785	155	18	242	118	3090
4	5	2595	45	2	153	-1	391

First columns and rows of the dataset

Soil_Type33	Soil_Type34	Soil_Type35	Soil_Type36	Soil_Type37	Soil_Type38	Soil_Type39	Soil_Type40	Cover_Type
0	0	0	0	0	0	0	0	3
0	0	0	0	0	0	0	0	3
0	0	0	0	0	0	0	0	3
0	0	0	0	0	0	0	0	3
0	0	0	0	0	0	0	0	3

Last columns and rows of the dataset

The very first observation is, there are some negative values in the `Vertical_Distance_To_Hydrology` column. I will examine this in more detail in the *Check for Anomalies & Outliers* section.

To understand `trees` dataframe, let's look at the data types and descriptive statistics. With pandas `info` method, we can list the non-null values and data types

With `describe` method, we can observe the descriptive statistics:

	Id	Elevation	Aspect	Slope	Horizontal_Distance_To_Hydrology	Vertical_Distance_To_Hydrology
count	15120.000000	15120.000000	15120.000000	15120.000000	15120.000000	15120.000000
mean	7560.500000	2749.322553	156.676653	16.501587	227.195701	51.076521
std	4364.91237	417.678187	110.085801	8.453927	210.075296	61.239406
min	1.000000	1863.000000	0.000000	0.000000	0.000000	-146.000000
25%	3780.750000	2376.000000	65.000000	10.000000	67.000000	5.000000
50%	7560.500000	2752.000000	126.000000	15.000000	180.000000	32.000000
75%	11340.250000	3104.000000	261.000000	22.000000	330.000000	79.000000
max	15120.000000	3849.000000	360.000000	52.000000	1343.000000	554.000000

Descriptive statistics of the first columns

Check for Anomalies & Outliers: The first anomaly was observed with the negative values in the `Vertical_Distance_To_Hydrology` column. The definition is:

Vertical distance to nearest surface water features

With some research and using logic, negative values show that the nearest surface water is below that data point or it is below the sea level.

2. Exploratory Data Analysis (EDA)

EDA is the first step in this workflow where the decision-making process is initiated for the feature selection. Some valuable insights can be obtained by looking at the distribution of the target, relationship of the features to the target and link between the features.

My preference is to start by looking at the target, then examine the features and its relations to the target.

Data set have balanced labels, resulting in an almost equal number of cover types for each class. This will be an advantage when we are applying ML algorithms because the model will have a good chance to learn patterns of all classes without needing further balancing strategies.

Wilderness Area — Soil Type — Cover Type:

Reverse-one-hot-encoding will be applied to soil type columns using the below functions. As a result, one column `Soil_Type` with discrete numbers between 1 and 40 will be added.

Relationships Between Continuous Features :

Soil type and wilderness area columns were discrete, and both are the one-hot-encoded version of one categorical feature.

Rest of the features are addressed as continuous:

Elevation, Aspect, Slope, Horizontal_Distance_To_Hydrology, Vertical_Distance_To_Hydrology, Horizontal_Distance_To_Roadways, Hillshade_9am, Hillshade_Noon, Hillshade_3pm, Horizontal_Distance_To_Fire_Points.

To visualize all of them with one function Seaborn's `PairGrid` will be plotted to provide the flexibility of adjusting upper and lower diagonal with different plots.

Pearson Coefficients of the Features and Target:

As a last step of the EDA, when Pearson coefficients of the features and targets are observed, only 1% of the one-hot-encoded soil type columns are effective in determining the `Cover_Type` (not shown in this article but in the notebook here). So they will be excluded and Pearson coefficients will be revisited with the continuous features, one-hot-encoded wilderness areas, `Soil_Type` and `Cover_Type`. With a label-encoded `Soil_Type` column there is a stronger correlation with the `Cover_Type`.

3. Feature Engineering & Selection

Extracting new features based on the existing ones and eliminating features with some methods and algorithms are called feature engineering & selection.

There are horizontal and vertical distance to hydrology features, which blinks for adding the euclidian distance of the two. Also, adding linear combinations of the numeric features is a common practice in feature engineering.

- Training set is used as an input, so that machine learning models can catch the patterns in the features and utilize them to distinguish the target.
- Validation set is used to evaluate the ML model's performance and quantify its ability to generalize patterns to a new data set.



4. Compare Several Machine Learning Models

Sometimes it is hard to know which machine learning model will work effectively for a given problem. So, I always try several machine learning models.

5. Perform Hyperparameter Tuning on the Best Model

Searching for the best combination of model parameters are called hyperparameter tuning which can dramatically improve the model's performance. I will use the random search algorithm with cross-validation for hyperparameter-tuning.

6. Interpret Model Results

Confusion Matrix: One of the most common methods for the visualization of a classification model results is the confusion matrix. Fantastic tree confusion matrix will be a 7x7 matrix. I will use the normalized confusion matrix, so the percentage of actual cover types correctly guessed out of all guesses in that particular category will appear in the diagonal of the matrix.

Performance Metrics:

We used accuracy and loss as the model metrics for both training ,validation and testing the model.

Confusion Matrix:

It is the easiest way to measure the performance of a classification problem where the output can be of two or more type of classes. A confusion matrix is nothing but a table with two dimensions viz. “Actual” and “Predicted” and furthermore, both the dimensions have “True Positives (TP)”, “True Negatives (TN)”, “False Positives (FP)”, “False Negatives (FN)” .



5.4.2. Classification Accuracy

It is most common performance metric for classification algorithms. It may be defined as the number of correct predictions made as a ratio of all predictions made. We can easily calculate it by confusion matrix with the help of following formula –

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN}$$

We can use `accuracy_score` function of `sklearn.metrics` to compute accuracy of our classification model.

Functional Requirements:

In software engineering, a functional requirement defines a function of a software system or its component. A function is described as a set of inputs, behavior, and outputs. Functional requirements may be calculations, technical details, data manipulation and processing, and other specific functionality that define what a system is supposed to accomplish. Behavioral requirements describing all the cases where the system uses the functional requirements are captured in use cases.

Functional Requirements for Present Project

1. Input: Forest cover types of the trained classes
2. Output: Cover type
3. Process: ANN
4. Training: ANN

■ System Requirements

To be used efficiently, all computer software needs certain hardware components or other software resources to be present on a computer. These prerequisites are known as (computer) system requirements and are often used as a guideline as opposed to an absolute rule. Industry analysts suggest that this trend plays a bigger part in driving upgrades to exist computer systems than technological advancements.

● Software Requirements

Software Requirements deals with defining software resource requirements and prerequisites that need to be installed on a computer to provide optimal functioning of an application. These requirements or pre-requisites are generally not included in the software installation package and need to be installed separately before the software is installed.

Software Requirements for Present Project

- Operating System: Windows 10
- Platform & Languages: Google Colab & Python.
- Libraries / packages: Pandas, numpy, seaborn, sickit learn, matplotlib, keras

About Software:

Colaboratory, or “Colab” for short, is a product from Google Research. Colab allows anybody to write and execute arbitrary python code through the browser, and is especially well suited to machine learning, data analysis and education. Google Colab is a powerful platform for learning and quickly developing machine learning models in Python. It is based on Jupyter notebook and supports collaborative development. The team members can share and concurrently edit the notebooks, even remotely. The notebooks can also be published on GitHub and shared with the general public. Colab supports many popular machine learning libraries which can be easily loaded in your notebook. Colab supports many popular ML libraries such as Pandas, Numpy, seaborn, PyTorch, TensorFlow, Keras and OpenCV, etc.....

What Colab Offers You?

As a programmer, you can perform the following using Google Colab.

- Write and execute code in Python
- Document your code that supports mathematical equations
- Create/Upload/Share notebooks
- Import/Save notebooks from/to Google Drive
- Import/Publish notebooks from GitHub
- Import external datasets e.g. from Kaggle
- Integrate PyTorch, TensorFlow, Keras, OpenCV
- Free Cloud service with free GPU

About Language:

Python is a general-purpose object-oriented programming language with high-level programming capabilities. It has become famous because of its apparent and easily understandable syntax, portability and easy to learn. Python is a programming language that includes features of C and Java. It provides the style of writing an elegant code like C, and for object-oriented programming, it offers classes and objects like Java. It has fewer syntactical constructions than other languages.

Python is Interpreted – Python is processed at run-time by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.

Python is Interactive – you can actually sit at a Python prompt and interact with the interpreter directly to write your program.

Python is Object-Oriented – Python supports Object-Oriented style or technique of programming that encapsulates code within objects.

Python is a Beginner's Language – Python is a great language for the beginner level programmers and

supports the development of a wide range of applications from simple text processing to WWW browsers to games.

Why Python?

- Python works on different platforms Windows, Mac, Linux, Raspberry Pi, etc.
- Python has a simple syntax similar to the English language.
- Python has syntax that allows developers to write programs with fewer lines than some other programming languages.
- Python runs on an interpreter system, meaning that code can be executed as soon as it is written. This means that prototyping can be very quick.
- Python can be treated in a procedural way, an object-oriented way or a functional way.

Packages/Libraries:

A library is a collection of precompiled codes that can be used later on in a program for some specific well-defined operations. Other than pre-compiled codes, a library may contain documentation, configuration data, message templates, classes, and values, etc. A Python library is a collection of related modules. It contains bundles of code that can be used repeatedly in different programs. It makes Python Programming simpler and convenient for the programmer. As we don't need to write the same code again and again for different programs. Python libraries play a very vital role in fields of Machine Learning, Data Science, Data Visualization, etc. The Python Standard Library contains the exact syntax, semantics, and tokens of Python. It contains built-in modules that provide access to basic system functionality like I/O and some other core modules. Most of the Python Libraries are written in the C programming language. The Python standard library consists of more than 200 core modules. All these work together to

make Python a high-level programming language. Python Standard Library plays a very important role. Without it, the programmers can't have access to the functionalities of Python.

1.Pandas: Pandas are an important library for data scientists. It is an open-source machine learning library that provides flexible high-level data structures and a variety of analysis tools. It eases data analysis, data manipulation, and cleaning of data. Pandas support operations like Sorting, Re-indexing, Iteration, Concatenation, Conversion of data, Visualizations, Aggregations, etc.

2.numpy: The name "Numpy" stands for "Numerical Python". It is the commonly used library. It is a popular machine learning library that supports large matrices and multi-dimensional data. It consists of in-built mathematical functions for easy computations. Even libraries like TensorFlow use Numpy internally to perform several operations on tensors. Array Interface is one of the key features of this library.

3.Seaborn: Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics. For a brief introduction to the ideas behind the library, you can read the introductory notes or the paper.

4.Matplot lib: This library is responsible for plotting numerical data. And that's why it is used in data analysis. It is also an open-source library and plots high-defined figures like pie charts, histograms, scatterplots, graphs, etc.

5.Scikit-learn: It is a famous Python library to work with complex data. Scikit-learn is an open-source library that supports machine learning. It supports variously supervised and unsupervised algorithms like linear regression, classification, clustering, etc. This library works in association with Numpy and SciPy.

6.keras: Keras is a high-level, deep learning API developed by Google for implementing neural networks. It is written in Python and is used to make the implementation of neural networks easy. It also supports multiple backend neural network computation. Keras is an open-source software library that provides a Python interface for artificial neural networks. Keras acts as an interface for the TensorFlow library. Up until version 2.3, Keras supported multiple backends, including TensorFlow, Microsoft Cognitive Toolkit, Theano, and PlaidML.

7.Keras Dense: Dense layer is the regular deeply connected neural network layer. It is most common and frequently used layer. Dense layer does the below operation on the input and return the output.

$$\text{output} = \text{activation}(\text{dot}(\text{input}, \text{kernel}) + \text{bias}).$$

Result: We have to built a solution using ANN for which we got accuracy 88% that should be able to predict the given different cover types in different wilderness areas of the Roosevelt National Forest.

Applications:

1. To identify the endanger condition so that we can protect those particular cover types.
2. Easy location of willow cover type which will be useful for making Cricket bats.
- 3.To detect the respective cover type which will also helps in manufacturing process in medicine.

limitations:

- 1 .Some approaches to machine learning rely on the accessibility of large amounts of labeled training data, the creation or curation of which can be resource intensive and time-consuming.
- 2.Humans are good at transferring ideas from one problem domain to another. This remains challenging for computers(Machine Learning) even with the latest machine learning techniques.
- 3.The models which worked on could not generate atmost accuracy.
- 4.It is not robust enough to handle very large datasets.

Conclusion and Future work

CONCLUSION:

The goal of this project is to predict the cover of forest based on cartographical data.. Previous reported scores are compared to results of this project and significant improvement noticed. Moreover, during this work 4 different classification techniques were applied and their advantages and disadvantages are discussed. Overall, best results are obtained using Artificial neural networks, which gave 88% and Random forest or extreme random decision trees, which gave 86% accuracy for train sets, for decision trees, which gave 79%accuracy , and for logistic regression , which gave accuracy 47%. .

FUTURE WORK:

In this interim Project work, We have reviewed by taking different Machine Learning models and observed accuracy based on training and test data. One approach is to guarantee a minimum level of performance. Hence this project members appeals for creating algorithms that can computationally scale to large data sets by using Deep learning models like CNN.

We have to create an APP. This code acts as backend . In that App , it will ask about different attributes such as elevation,slope,soil types etc as input and it will produce the results which indicate covertypes.

References:

1. <https://www.kaggle.com/code/devanshiipatel/forest-cover-type-classification>
2. <https://archive.ics.uci.edu/ml/datasets/Covertypes?ref=datanews.io>
3. Text book: Machine Learning using Python by U.Dinesh kumar and Manaranjan Pradhan

Annexure:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.preprocessing import StandardScaler
from collections import Counter
from imblearn.over_sampling import SMOTE
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix, accuracy_score, classification_report

import keras
from keras.models import Sequential
from keras.layers import Dense
data = pd.read_csv("/content/drive/MyDrive/covtype.csv")
df.head()
from google.colab import drive
drive.mount('/content/drive/')
df=pd.read_csv('/content/drive/MyDrive/covtype.csv')
from google.colab import drive
drive.mount('/content/drive')
df.head()
df.tail()
df.shape
df.info()
duplicate = df[df.duplicated()]
duplicate
plt.style.use('seaborn')
corr = df.corr()
corr.style.background_gradient(cmap='PuBu').set_precision(2)
df.corr().unstack().sort_values().drop_duplicates()
fig, ax = plt.subplots(19, 3, figsize=(50, 120))
for variable, subplot in zip(df, ax.flatten()):
    sns.histplot(df[variable], ax=subplot)
```



```

for label in subplot.get_xticklabels():
    label.set_rotation(90)
sns.countplot?
sns.countplot(data=df, x='Cover_Type')
cols = ['Elevation', 'Aspect', 'Slope', 'Horizontal_Distance_To_Hydrology',
'Vertical_Distance_To_Hydrology', 'Horizontal_Distance_To_Roadways', 'Hillshade_9am',
'Hillshade_Noon', 'Hillshade_3pm', 'Horizontal_Distance_To_Fire_Points']

scaler = StandardScaler()
for col in cols:
    df[[col]] = scaler.fit_transform(df[[col]])////////////////////////////////////
df
X = df.iloc[:, :-1]
y = df.iloc[:, -1]
_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)

print("X_train:", X_train.shape)
print("X_test:", X_test.shape)
print("y_train:", y_train.shape)
print("y_test:", y_test.shape)
classifier = Sequential()
classifier.add(Dense(units=54, kernel_initializer='he_uniform', activation='relu', input_dim = 54))
classifier.add(Dense(units=50, kernel_initializer='he_uniform', activation='relu'))
classifier.add(Dense(units=35, kernel_initializer='he_uniform', activation='relu'))
classifier.add(Dense(units=8, activation='softmax'))

# Compiling the ANN
classifier.compile(optimizer='adam', loss='sparse_categorical_crossentropy', metrics=['accuracy'])
classifier.summary()
model_history = classifier.fit(X_train, y_train, validation_split=0.33, batch_size=1000, epochs=100)
# List all data in history
print(model_history.history.keys())

# Summarize history for accuracy
plt.plot(model_history.history['accuracy'])
plt.plot(model_history.history['val_accuracy'])
plt.title('Model Accuracy')
plt.ylabel('Accuracy')

```

```

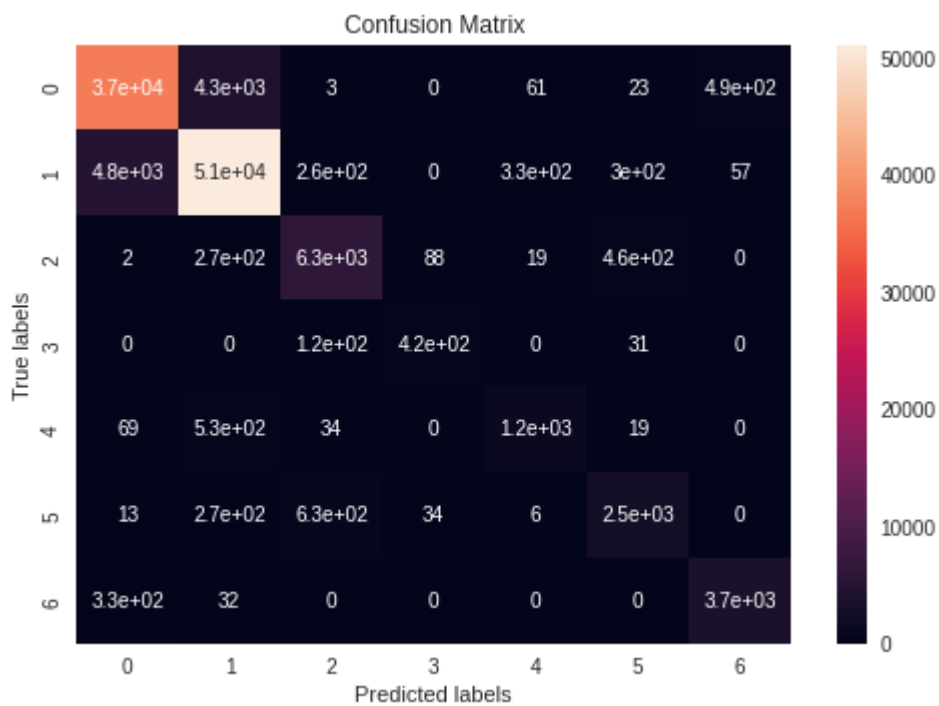
plt.xlabel('Epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()
plt.plot(model_history.history['loss'])
plt.plot(model_history.history['val_loss'])
plt.title('Model Loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()
score = classifier.evaluate(X_test, y_test)
score
y_pred = classifier.predict(X_test)
y_pred = np.argmax(y_pred, axis=1)
y_pred
accuracy_score(y_pred, y_test)
print(classification_report(y_pred, y_test))
cm = confusion_matrix(y_test, y_pred)

ax= plt.subplot()
sns.heatmap(cm, annot=True, ax=ax); #annot=True to annotate cells

# labels, title and ticks
ax.set_xlabel('Predicted labels')
ax.set_ylabel('True labels')
ax.set_title('Confusion Matrix')

```

OUTPUT:



Representation of coverytype:

