

PROFESSIONAL TRAINING REPORT
at
Sathyabama Institute of Science and Technology
(Deemed to be University)

Submitted in partial fulfillment of the requirements for award of the
Bachelor of Engineering Degree in Compute Science and Engineering

By

Konderu Bhavani
(Reg. No.38110261)



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING
SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY
JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI – 600119, TAMILNADU

MARCH 2022



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with Grade “A” by NAAC

(Established under Section 3 of UGC Act, 1956)

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI– 600119

www.sathyabamauniversity.ac.in



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **K. BHAVANI (38110261)** who carried out the project entitled “**Powdery Mildew Prediction in Sandalwood Trees**” under my supervision from November 2021 to March 2021.

Internal Guide

Dr.J.Cruz Antony,MCA.,Ph.D.

Head of the Department

Dr.S. Vigneshwari M.E., Ph.D.

Dr.L.Lakshmanan M.E., Ph.D.

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I **KONDERU BHAVANI** hereby declare that the Project Report entitled **POWDERY MILDEW PREDICTION IN SANDALWOOD TREES** done by me under the guidance of **Dr.J.Cruz Antony,MCA.,Ph.D.** at Sathyabama Institute of Science and Technology is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE:

K. Bhavani

PLACE: Chennai

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T.Sasikala M.E., Ph.D., Dean**, School of Computing, **Dr.S.Vigneshwari M.E., Ph.D., and Dr.L.Lakshmanan M.E., Ph.D.**, Heads of the Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr.J.Cruz Antony,MCA.,Ph.D.**, for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

Powdery Mildew is a fungal disease that grows on leaves, branches in both field and greenhouse production of many crop plants, and its infection can reduce the yield and cause loss. Powdery Mildew is caused due to change in climate conditions with high humidity and moderate temperature. In this project we will create a machine learning model which will predict the chances of getting Powdery Mildew disease is in low, high or medium state based on climate conditions like temperature, humidity, soil moisture and cloud factor on Sandal Wood trees. In this project we are predicting the Powdery mildew disease data by using some machine learning algorithms such as Logistic regression, Decision Tree, Random Forest, Support Vector Machine and K-Nearest Neighbours. The model with highest accuracy is taken for web app building as it gives better predictions when compared with other models and it will suggest the precautions to be taken to avoid powdery mildew disease. Finally, the web app is deployed in Heroku platform to run the application on cloud.

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	
	LIST OF FIGURES	
1	INTRODUCTION	1
2	SYSTEM IMPLIMENTATION AND DESIGN	2
	2.1 Data Science and its Methodology	2
	2.2 Machine Learning	3
	2.3 Libraries Involved	4
	2.4 Data Collection	6
	2.5 Feature Engineering	7
	2.6 Exploratory Data Analysis	7
	2.7 Data Filtering	9
	2.8 Splitting the Data	10
	2.9 Model Building	10
	2.10 Web-App Representation	13
3	RESULTS, DISCUSSION AND PERFORMANCE ANALYSIS	15
4	CONCLUSION	16
	SCREENSHOTS	16
	SOURCE CODE	19
	REFERENCES	25

LIST OF FIGURES

FIGURE NO.	FIGURE NAME	PAGE NO.
2.1	Machine Learning	4
2.2	Data Set	6
2.3	correlation	7
2.4	Count plot of Cloud factor with respect to Symptoms	8
2.5	Count plot of Soil moisture with respect to Symptoms	8
2.6	Count plot of Temperature with respect to Symptoms	9
2.7	Count plot of Humidity with respect to Symptoms	9
2.8	Requirements.txt file	13
2.9	Procfile	14
2.10	Setup.sh file	14
4.1	Commands to run	16
4.2	Deployment in Heroku	17
4.3	Web page	18

CHAPTER 1

INTRODUCTION

Sandalwood (*Santalum album*) commonly known as Indian sandalwood is a small tropical tree native in the dry regions of Southern India. From its natural area of distribution, *S. album* was introduced to Northern, Central (Srinivasan *et al.*, 1992), North Western (Das, 2013) and North Eastern India (Viswanath, 2014; Babita *et al.*, 2016). The tremendous extension in the area of cultivation of sandalwood was only in the past decade, before that sandalwood was largely confined to the forests and government-owned plantations of Karnataka, Tamil Nadu and Kerala. In the early 2000s, Karnataka and Tamil Nadu governments brought a policy change respectively in the liberalization of growing sandalwood, which allowed many private owners and firms to cultivate sandalwood (Aparna Pallavi, 2018).

Plants are continually exposed to pathogens like bacteria, viruses, and fungi. These pathogens cause plant disease which leads to a significant loss in crop yield globally (Yang and Guo, 2017). Plant diseases affect the leaves at any moment between sowing and harvesting which steers to a huge loss on crop production. The most common plant diseases include bacterial blight, powdery mildew, leaf rust, brown spot, etc. (Annabel *et al.*, 2019). Machine learning techniques have an advantage over statistical methods by focusing on the data themselves and highlighting the performance of the given task.

AIM AND SCOPE OF THE PRESENT INVESTIGATION:

The main objective of this problem is to find whether the chances of getting powdery mildew disease is in low, high or medium state based on soil moisture, temperature, humidity, cloud factor.

CHAPTER 2

SYSTEM IMPLEMENTATION AND DESIGN

2.1 DATA SCIENCE AND ITS METHODOLOGY:

Data Science is kind, blended with various tools, algorithms, and machine learning principles. Most simply, it involves obtaining meaningful information or insights from structured or unstructured data through a process of analyzing, programming and business skills. It is a field containing many elements like mathematics, statistics, computer science, etc. Those who are good at these respective fields with enough knowledge of the domain in which you are willing to work can call themselves a Data Scientist. It's not an easy thing to do but not impossible too. You need to start from data, it's visualization, programming, formulation, development, and deployment of your model.

Data science is not a one-step process such that you will get to learn it in a short time and call ourselves a Data Scientist. It passes from many stages and every element is important. One should always follow the proper steps to reach the ladder. Every step has its value and it counts in your model.

Problem Statement: No work starts without motivation, Data science is no exception though. It's really important to declare or formulate our problem statement very clearly and precisely. Your whole model and it's working depend on our statement. Many scientists consider this as the main and much important step of Date Science. so make sure what's the problem statement and how well can it add value to business or any other organization. . This problem statement comes under classification problem of supervised learning as the target feature consists of categorical variables.

Data Collection: After defining the problem statement, the next obvious step is to go in search of data that you might require for our model. we must do good research, find all that you need. Data can be in any form i.e unstructured or structured. It might be in various forms like videos, spreadsheets, coded forms, etc. You must collect all these kinds of sources. In this project the data is collected from IWST nursesey.

Data Cleaning: As you have formulated your motive and also you did collect your data, the next step to do is cleaning. Data cleaning is the most favorite thing for data scientists to do. Data cleaning is all about the removal of missing, redundant, unnecessary and duplicate data from your collection. There are various tools to do so with the help of programming in Python.

Data Analysis and Exploration: It's one of the prime things in data science to do and time to get inner Holmes out. It's about analyzing the structure of data, finding hidden patterns in them, studying behaviors, visualizing the effects of one variable over others and then concluding. We can explore the data with the help of various graphs formed with the help of libraries using any programming language. In python we use matplotlib and seaborn for data analysis.

Data Modelling: The data is splitted into training set and test set and training data is used to build the models. The models that we use in the project are classification algorithms like LogisticRegression, svm, Decision Tree, Knn, Random Forest which are directly available from scikit-learn library from python.

Optimization and Deployment: To test the data and find how well it is performing is checked by its accuracy. In short, we check the efficiency of the data model and thus try to optimize it for better accurate prediction. Deployment deals with the launch of your model and let the people outside there to benefit from that. We deployed our model using streamlit and heroku.

2.2 MACHINE LEARNING:

Machine Learning is scientific field dealing with the way in which machine learning from experience. the purpose of machine learning is the construction of computer system that can adapt and learn from their experience. A more detailed and formal definition of machine learning is given by Mitchel: A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P , if its performance at tasks in T , as measured by P , improves with experience E .

With the rise of Machine Learning approaches we have the ability to find a solution to this issue, we have developed a system using data mining which has the ability to predict whether the chances of getting powdery mildew disease is in low, high

or medium state. This Problem is done by collecting Data of the previous records in which the different climate conditions for different stages of disease are given, on the basis of these records/experiences the machine was trained using the machine learning model which give the most accurate result. The aim of this research is to develop a system which can predict whether the chances of getting powdery mildew disease is in low, high or medium state. This research has focused on developing a system based on five classification methods namely, SVM, Logistic regression, Decision Tree, KNN and Random Forest.

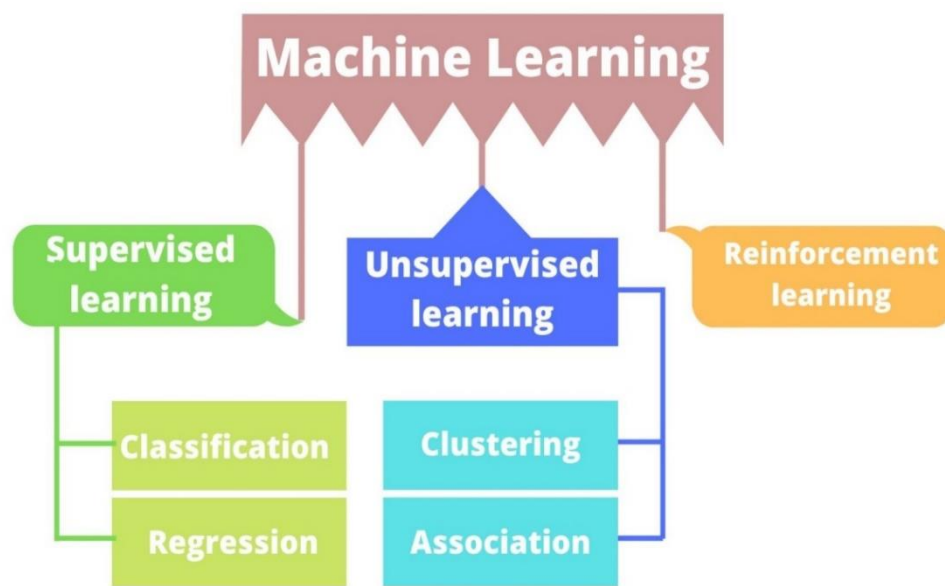


Fig 2.1: Machine Learning

prediction problems come under supervised learning, as in supervised learning we are going to train the model by giving both input and output and test the model by giving different inputs.

2.3 LIBRARIES INVOLVED:

Numpy:

NumPy is a python library used for working with arrays. It also has functions for working in the domain of linear algebra, flourier transform, and matrices. It is an

open-source project and you can use it freely. NumPy stands for Numerical Python.

Pandas:

Panda's is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series. It is an open source project and you can use it freely. Data Frames in pandas plays major role in building and correcting the data

Matplotlib:

Matplotlib is an amazing visualization library in Python for 2D plots of arrays. Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy stack. it allows us visual access to huge amounts of data in easily digestible visuals. Matplotlib consists of several plots like line, bar, scatter, histogram etc

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.

SciKit Learn:

Scikit-learn is probably the most useful library for machine learning in Python. The sklearn library contains a lot of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction. Sklearn is used to build machine learning models. It should not be used for reading the data, manipulating and summarizing it. There are better libraries for that. Scikit-learn comes loaded with a lot of features. Scikit-learn comes loaded with a lot of features .In supervised learning, Starting from Generalized linear models (e.g Linear Regression), Support Vector Machines

(SVM), Decision Trees to Bayesian methods – all of them are part of scikit-learn toolbox

2.4 DATA COLLECTION:

The dataset consists of powdery mildew symptoms (class/target variable) on sandalwood from IWST Nursery classified as High, Medium, and Low along with its respective abiotic features viz., Cloud factor, Soil moisture, Temperature, and Humidity from the year 2016 to 2020. The abiotic features were also discretized as per table below.

Abiotic features	Discrete values	Actual value (range)
Cloud factor	Broken	3 to 7 Oktas
	Closed	8 to 9 Oktas
	Open	0 to 2 Oktas
Soil moisture	Adequate	70 to 90 %
	Inadequate	below 70% (or 50 to 70%)
Temperature	Maximum	30°C to 35°C
	Minimum	15°C to 25°C
	Moderate	25°C to 30°C
Humidity	Maximum	90% to 100%
	Minimum	70% to 90%
	Moderate	50% to 70%

Fig 2.2: Data Set

we use panda's library to read the data set. The function 'pd.read_csv' helps to load the data present in the jupyter notebook and the dataset is assigned to an object which helps to print the data. The data set is now supplied to machine learning model, on the basis of this data set the model is trained. Every new climate conditions details filled at the time of application form acts as a test data

set. After the operation of testing, model predict whether the chances of getting powdery mildew disease is in low, high or medium state based upon the inference it conclude on the basis of the training data sets.

2.5 FEATURE ENGINEERING:

Correlation:

correlation is a term that is a measure of the strength of a linear relationship between two quantitative variables. This is when one variable increases while the other increases and vice versa.

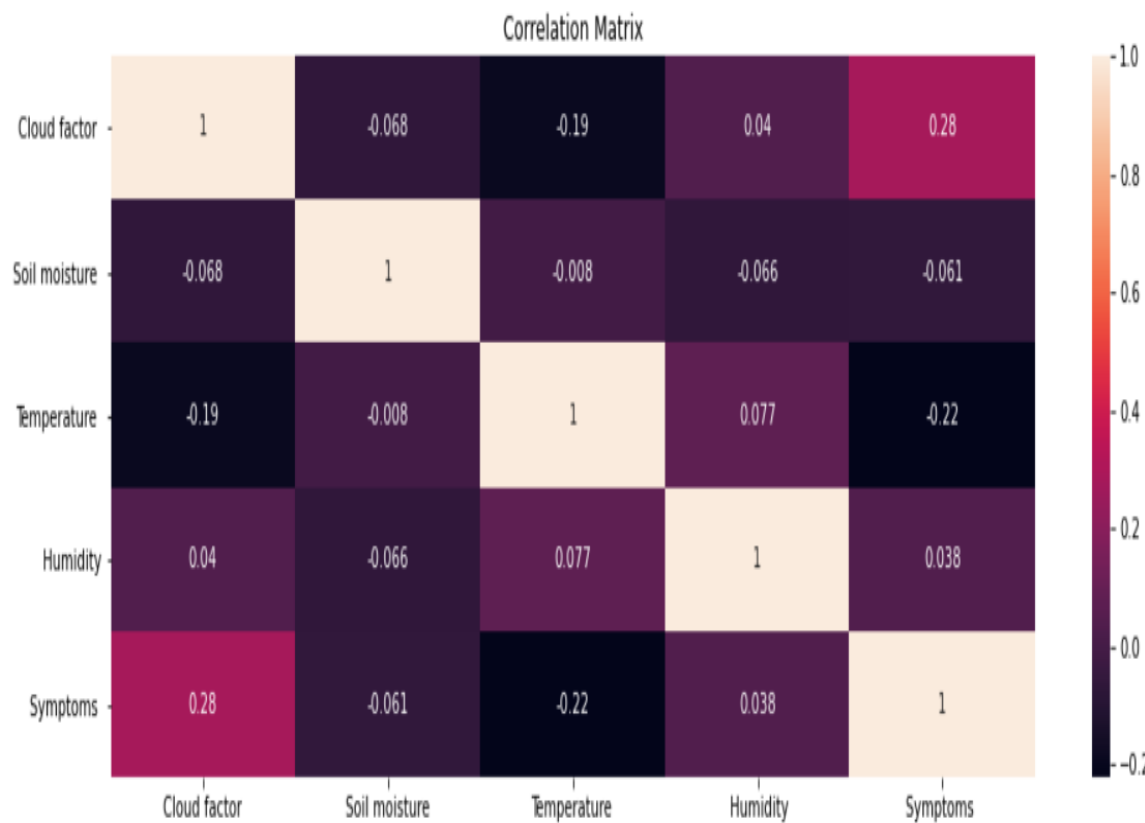


Fig 2.3: correlation

2.6 EXPLORATORY DATA ANALYSIS:

Exploratory data analysis is an approach to analyzing data sets to summarize their main characteristics, often with visual methods. A statistical model can be used or not, but primarily EDA is for seeing what the data can tell us beyond the formal modeling or hypothesis testing task

Symptoms	High	Low	Medium
Cloud factor			
Broken	222	1	44
Closed	125	3	103
Open	101	12	277

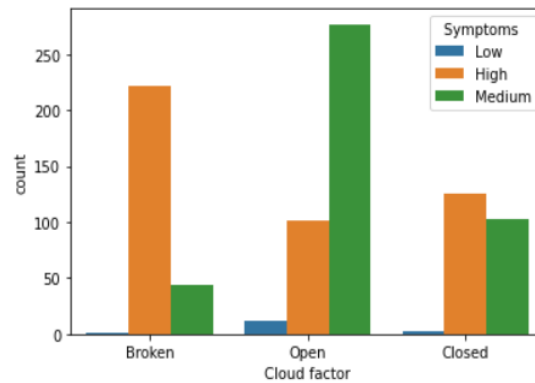


Fig 2.4: Count plot of Cloud factor with respect to Symptoms

From the above figure, we can see the chances of getting powdery mildew disease is high when the cloud is broken and closed

Symptoms	High	Low	Medium
Soil moisture			
Adequate	448	16	420
Inadequate	0	0	4

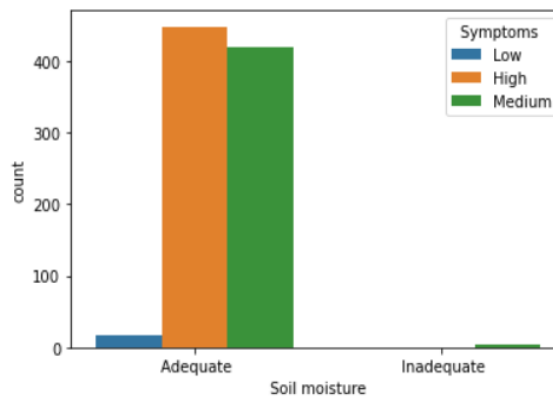


Fig 2.5: Count plot of Soil moisture with respect to Symptoms

From the above figure, we can see the chances of getting powdery mildew disease is high when the soil moisture is adequate

Symptoms	High	Low	Medium
Temperature			
Maximum	42	0	43
Minimum	99	2	243
Moderate	307	14	138

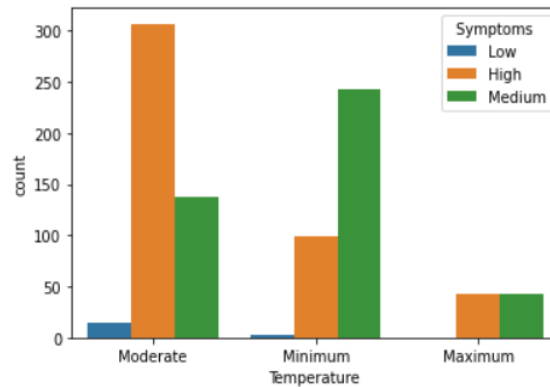


Fig 2.6: Count plot of Temperature with respect to Symptoms

From the above figure, we can see the chances of getting powdery mildew disease is high when the temperature is Moderate and minimum

Symptoms	High	Low	Medium
Humidity			
Maximum	183	16	9
Minimum	20	0	262
Moderate	245	0	153

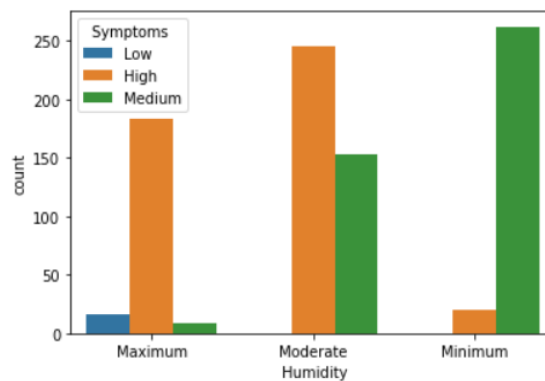


Fig 2.7: Count plot of Humidity with respect to Symptoms

From the above figure, we can see the chances of getting powdery mildew disease is high when the Humidity is Maximum and moderate.

2.7 DATA FILTERING:

The data which was collected might contain nan values or missing values that may lead to inconsistency. To gain better results data need to be preprocessed so as to improve the efficiency of the algorithm

We need to convert categorical features to numerical form to avoid disturbances in building the model. This is done by using map function. But the data that we collected is already filtered.

2.8 SPLITTING THE DATA:

The data is splitted into training data and testing data .We have to import **model_selection** from sklearn library and called the function train test split which helps us to split the data into training set and testing set .we have to split the data in such a that training data is 80% and test data is 20% of the total data.

Training set is used for training the data and for Cross Validation we use Test sets as common best practices. This allows you to tune various parameters of the algorithm without making judgements that specifically conform to training data

2.9 MODEL BUILDING:

Logistic Regression:

Logistic regression, despite its name, is a linear model for classification rather than regression. Logistic regression is also known in the literature as logit regression, maximum-entropy classification and the log-linear classifier. In this model, the probabilities describing the possible outcomes of a single trial are modeled using a logistic function.

We have to import the LogisticRegression class of Linear_model module from sklearn library and create the object for LogisticRegression class and perform a **fit** function by passing the training data. Then we are predicting the model outputs by calling a function called **predict**. and then we are finding the accuracy of the model by calling the accuracy_score function of the metrics module from sklearn library which was found to be 69 percent.

Support Vector Machine:

SVM is a set of related supervised learning methods for classification and regression. SVM simultaneously minimizes the empirical classification error and maximizes the geometric margin. So SVM is called Maximum Margin Classifiers. SVM is a general algorithm based on guaranteed risk bounds of statistical learning

theory, the so-called structural risk minimization principle. SVMs can efficiently perform nonlinear classification using what is called the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces. The kernel trick allows constructing the classifier without explicitly knowing the feature space

We have to import the `svc` class of the `svm` module from the `sklearn` library and create the object for the `svc` class and perform a **fit** function by passing the training data. Then we are predicting the model outputs by calling a function called **predict** and then we are finding the accuracy of the model by calling the `accuracy_score` function of the `metrics` module from `sklearn` library which was found to be 79 percent.

Decision tree:

A decision tree is a flowchart-like tree structure where an internal node represents a feature (or attribute), the branch represents a decision rule, and each leaf node represents the outcome. The topmost node in a decision tree is known as the root node. It learns to partition on the basis of the attribute value. It partitions the tree in a recursive manner called recursive partitioning. This flowchart-like structure helps you in decision making. It's visualization like a flowchart diagram which easily mimics the human level thinking. That is why decision trees are easy to understand and interpret. It shares internal decision-making logic. The decision tree is a distribution-free or non-parametric method, which does not depend upon probability distribution assumptions

We have to import `DecisionTreeClassifier` from the `tree` module from the `sklearn` library, and create the object for `DecisionTreeClassifier` class and perform a **fit** function by passing the training data. Then we are predicting the model outputs by calling a function called **predict** and then we are finding the accuracy of the model by calling the `accuracy_score` function of the `metrics` module from `sklearn` library which was found to be 79 percent.

K-Nearest Neighbors:

The simplest machine learning algorithm visualized so far till now is the K-NN algorithm due to its high accuracy. The smallest reversible entities that make up the basic small units to collaborate and unite in order to represent a solid structure is this data that is totally machine independent in all fields. All the facts and statistics that are combined together for the purpose of reference and analysis are raw data objects. It is the basic means of calculation and system reasoning for normalization purposes. The beginning method comes as the collection of the data needed and further articulating the given problem statement which defines the required data for system analysis. We must manipulate the phenomenon of gathering or else combining the data and information from widely available different sources. Now our main task comes in the form of data preprocessing where three main tasks are performed those are formatting, cleaning and sampling

We have to import the `K_Neighbour_Classifier` of `neighbour` module from the `sklearn` library, and should create the object for the `K_Neighbour_Classifier` class and perform a **fit** function by passing the training data. Then we are predicting the model outputs by calling a function called **predict** and then we are finding the accuracy of the model by calling the `accuracy_score` function of the `metrics` module from `sklearn` library which was found to be 80 percent.

Random Forest:

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random decision forests correct for decision trees' habit of overfitting to their training set. Random forests can be used to rank the importance of variables in a regression or classification problem in a natural way

We have to import `RandomForestClassifier` from the `ensemble` class from the `sklearn` library and create the object for `RandomForestClassifier` class and perform a **fit** function by passing the training data. Then we are predicting the model outputs by calling a function called **predict** and then we are finding the

accuracy of the model by calling the `accuracy_score` function of the `metrics` module from `sklearn` library which was found to be 79 percent.

2.10 WEB-APP REPRESENTATION:

To see the results of the model, we are creating a machine learning app using `streamlit`. `Streamlit` is an open-source Python library that makes it easy to build beautiful custom web-apps for machine learning and data science. The code for web app is written in `app.py` file

To run the application in cloud, we deployed the codes of this project in Heroku platform (Heroku is a platform as a service that enables developers to build, run, and operate applications entirely in the cloud)

We are fetching the model by creating pickle file of model building code into `app` code and we are passing the inputs to the model and it will show the output. The inputs are taken in web page by using the pre defined functions present in the `streamlit` library of python. The functions that we use for input are `st.number_input`, `st.selectbox`. then passing these inputs to `st.predict` function (which will take the inputs and returns the output of the model) If the output predicted by the model is 0 then we are displaying 'Chances of getting Powdery Mildew disease is Low(20% to 40%)', if it is 1 then we are displaying 'Chances of getting Powdery Mildew disease is Medium(40% to 70%)' and if it is 2 we are displaying 'Chances of getting Powdery Mildew disease is High(70% to 90%)'.

To run this web page, we need to run command '`streamlit run app_name.py`' which will create a local host url. But to run the app in cloud, we are deploying the codes into heroku platform (platform as a service). To deploy the codes we need to create three more files, they are `requirements.txt`, `procfile`, `setup.sh` file. After deployment in heroku it will give us an url which will run our web page on cloud.

```

1 matplotlib==3.3.1
2 pandas==1.1.0
3 scikit-learn==0.23.1
4 seaborn==0.10.1
5 streamlit==0.65.2
6 numpy==1.19.1
7

```

Fig 2.8: Requirements.txt file

The above figure is Requirements.txt file which consists of all the libraries that we have used in project. This file is used for installing the libraries needed for this project in heroku platform.

```

1 web: sh setup.sh && streamlit run app.py
2

```

Fig 2.9: Procfile

The above figure is Procfile which was used to understand from which file the execution should start in heroku platform.

```

1 mkdir -p ~/.streamlit/
2 echo "\
3 [general]\n\
4 email = \"your-email@domain.com\"\n\
5 \" > ~/.streamlit/credentials.toml
6 echo "\
7 [server]\n\
8 headless = true\n\
9 enableCORS=false\n\
10 port = $PORT\n\
11 \" > ~/.streamlit/config.toml
12

```

Fig 2.10: Setup.sh file

The above figure is Setup.sh file which consists of bash language and it is used to create a necessary environment's for streamlit app to run on the cloud.

CHAPTER 3

RESULTS, DISCUSSION AND PERFORMANCE ANALYSIS

The accuracy of model is calculated for every algorithms that we have used in this problem, the algorithm which has highest accuracy is considered as perfect model

The accuracy of different models are as follows-

- Logistic regression accuracy=0.69
- Svm accuracy=0.79
- Decision tree accuracy=0.79
- Knn accuracy=0.80
- Random forest accuracy=0.79

The final result has been collected successfully by using machine learning models. There is different accuracy related to different machine learning algorithms used such as support vector machines, random forest simplification, logistic regression, KNN and decision trees according to the Powdery Mildew prediction dataset. From the above results the accuracy of model is high for KNN model, so we have accumulated the result as per KNN in which we have got the accuracy of 80%.

CHAPTER 4

CONCLUSION

The final result has been collected successfully by using machine learning models. There is different accuracy related to different machine learning algorithms used such as support vector machines, random forest, logistic regression, KNN and decision trees according to the Powdery Mildew prediction dataset. From the above results the accuracy of model is high for K-Nearest Neighbors model, so we have accumulated the result for testing data as per K-Nearest Neighbours in which we have got the accuracy of 80%.

FUTURE WORK:

Other models like neural network and discriminate analysis can be used individually or combined for enhancing reliability and accuracy prediction.

SCREENSHOTS:

```
Anaconda Prompt (anaconda3) - streamlit run app.py

(base) C:\Users\Konderu Bhavani>cd C:\Users\Konderu Bhavani\OneDrive\Desktop\powdery mildew

(base) C:\Users\Konderu Bhavani\OneDrive\Desktop\powdery mildew>streamlit run app.py

You can now view your Streamlit app in your browser.

Local URL: http://localhost:8501
Network URL: http://192.168.1.4:8501
```

Fig 4.1: Commands to Run

The above figure tells about the command to run the app on anaconda prompt, the command used is 'streamlit run app.py'.

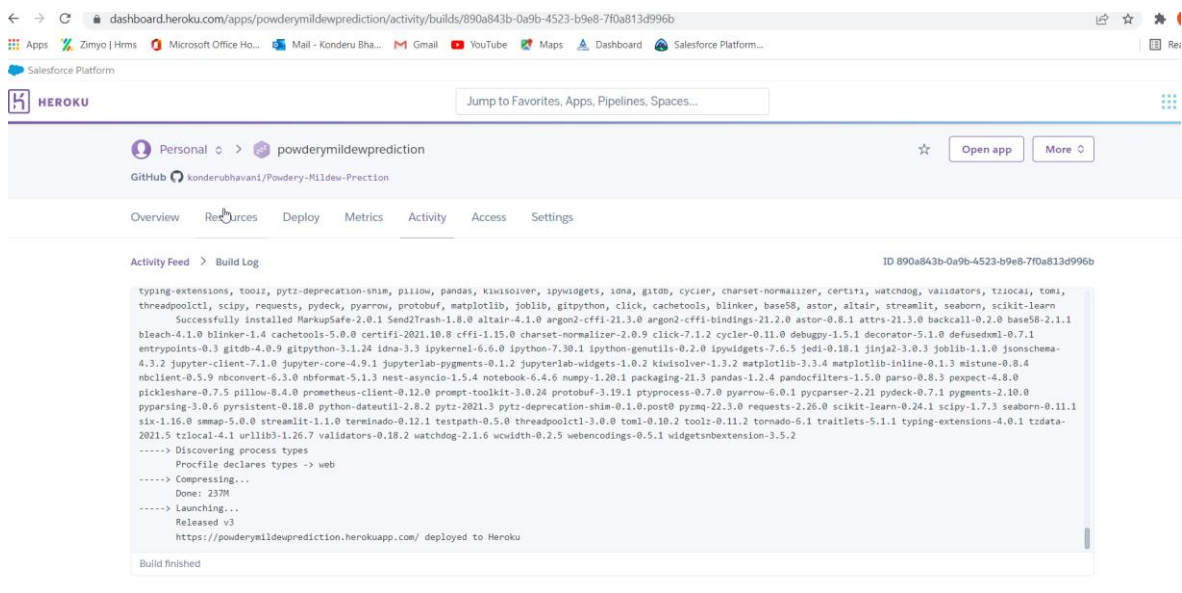


Fig 4.2: Deployment in Heroku

The above figure is heroku platform, which is used to run the web page on cloud. After deployment of codes in heroku it has gave the url which can be seen in below figure.

app - Streamlit

https://powderymildewprediction.herokuapp.com

Powdery Mildew Prediction ML App

Select Cloud Factor

Open (0 to 2 Oktas)

Select Soil moisture

Adequate (70% to 90%)

Select Temperature

Moderate (25C to 30C)

Select Humidity

Moderate(70%rh to 90%rh)

predict

Chances of getting Powdery Mildew disease is Medium(40% to 70%)

Precautions Or Treatment to avoid Powdery Mildew:

app - Streamlit

https://powderymildewprediction.herokuapp.com

predict

Chances of getting Powdery Mildew disease is Medium(40% to 70%)

Precautions Or Treatment to avoid Powdery Mildew:

BAKING SODA SOLUTION: Mix 1 tablespoon baking soda and ½ teaspoon liquid soap such as Castile soap (not detergent) in 1 gallon of water. Spray liberally, getting top and bottom leaf surfaces and any affected areas. This method may work better as a preventative measure, although it does have some effect on existing powdery mildew as well.

POTASSIUM BICARBONATE: Mix 1 tablespoon potassium bicarbonate and ½ teaspoon liquid soap (not detergent) in 1 gallon of water. Spray liberally to all affected areas. This mixture may work better than baking soda.

NEEM OIL: By itself, neem oil has mixed reviews on its effectiveness to treat powdery mildew, but it can be added to the above mixtures for an extra boost.

POWDERY MILDEW FUNGICIDE: Use sulfur-containing organic fungicides as both preventive and treatment for existing infections.

TRIM OR PRUNE: Remove the affected leaves, stems, buds, fruit or vegetables from the plant and discard. Some perennials can be cut down to the ground and new growth will emerge. Do not compost any damaged or diseased foliage as the spores can spread and persist in the composted material. Disinfect pruners and all tools after using on infected plants.

Fig 4.3: Web Page

The above figure is web page which predicts whether the chances of getting powdery mildew disease is in low, high or medium state based on the given inputs.

SOURCE CODE:

Model building code:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn import svm
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn import metrics
df=pd.read_csv("data_1.csv")
df
df.info()
df.describe()
list(df.columns)
df['Temperature '].value_counts()
df[' Cloud factor '].value_counts()
df[' Soil moisture '].value_counts()
df[' Humidity'].value_counts()
df['Symptoms '].value_counts()
```

```

sns.countplot(df[' Cloud factor '],hue=df['Symptoms '])
print(pd.crosstab(df[' Cloud factor '],df['Symptoms ']))
sns.countplot(df[' Soil moisture '],hue=df['Symptoms '])
print(pd.crosstab(df[' Soil moisture '],df['Symptoms ']))
sns.countplot(df['Temperature '],hue=df['Symptoms '])
print(pd.crosstab(df['Temperature '],df['Symptoms ']))
sns.countplot(df[' Humidity'],hue=df['Symptoms '])
print(pd.crosstab(df[' Humidity'],df['Symptoms ']))

df['Symptoms '].replace('Low',0,inplace=True)
df['Symptoms '].replace('Medium',1,inplace=True)
df['Symptoms '].replace('High',2,inplace=True)
df['Temperature '].replace('Moderate',0,inplace=True)
df['Temperature '].replace('Minimum',1,inplace=True)
df['Temperature '].replace('Maximum',2,inplace=True)
df[' Cloud factor '].replace('Open',0,inplace=True)
df[' Cloud factor '].replace('Broken',1,inplace=True)
df[' Cloud factor '].replace('Closed',2,inplace=True)
df[' Soil moisture '].replace(' Adequate',0,inplace=True)
df[' Soil moisture '].replace(' Inadequate',1,inplace=True)
df[' Humidity'].replace('Moderate',0,inplace=True)
df[' Humidity'].replace('Minimum',1,inplace=True)
df[' Humidity'].replace('Maximum',2,inplace=True)
df.drop(' Date ',axis='columns',inplace=True)
plt.figure(figsize=(16,5))
sns.heatmap(df.corr(),annot=True)
plt.title('Correlation Matrix')

y = df['Symptoms ']

```

```

X = df.drop(['Symptoms'], axis = 1)

X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2,random_state=0)

lr_model = LogisticRegression()

lr_model.fit(X_train,y_train)


lr_prediction = lr_model.predict(X_test)

print('Logistic Regression accuracy = ',
metrics.accuracy_score(lr_prediction,y_test))

svm_model = svm.SVC()

svm_model.fit(X_train,y_train)

svc_prediction = svm_model.predict(X_test)

print('SVM accuracy = ', metrics.accuracy_score(svc_prediction,y_test))


dt_model = DecisionTreeClassifier()

dt_model.fit(X_train,y_train)

dt_prediction = dt_model.predict(X_test)

print('Decision Tree accuracy = ', metrics.accuracy_score(dt_prediction,y_test))

knn_model = KNeighborsClassifier()

knn_model.fit(X_train,y_train)


knn_prediction =knn_model.predict(X_test)

print('KNN accuracy = ', metrics.accuracy_score(knn_prediction,y_test))

from sklearn.metrics import confusion_matrix

cm = confusion_matrix(y_test,knn_prediction)

print(cm)

data={'y_actual':y_test,'y_predicted':knn_prediction}

da=pd.DataFrame(data,columns=['y_actual','y_predicted'])

confusion_matrix=pd.crosstab(da['y_actual'],da['y_predicted'],rownames=['Actual'],
colnames=['Predicted'])

```

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

```
rf_model = RandomForestClassifier(n_estimators = 10, criterion = 'entropy',  
random_state = 0)
```

```
rf_model.fit(X_train, y_train)
```

```
rf_prediction=rf_model.predict(X_test)
```

```
print('random forest accuracy = ',metrics.accuracy_score(rf_prediction,y_test))
```

```
import pickle
```

```
pickle.dump(knn_model,open('model.pkl','wb'))
```

App Building Code:

```
import streamlit as st
```

```
import pickle
```

```
import numpy as np
```

```
model=pickle.load(open('model.pkl','rb'))
```

```
def inputs():
```

```
    CloudFactor = st.selectbox("Select Cloud  
Factor",tuple(cloudfactor_label.keys()))
```

```
    Soilmoisture = st.selectbox("Select Soil  
moisture",tuple(soilmoisture_label.keys()))
```

```
    Temperature = st.selectbox("Select  
Temperature",tuple(temperature_label.keys()))
```

```
    Humidity = st.selectbox("Select Humidity",tuple(humidity_label.keys()))
```

```
#encoding
```

```
v_CloudFactor=get_value(CloudFactor,cloudfactor_label)
```

```
v_Soilmoisture=get_value(Soilmoisture,soilmoisture_label)
```

```

v_Temperature=get_value(Temperature,temperature_label)
v_Humidity=get_value(Humidity,humidity_label)

data=np.array([[v_CloudFactor,v_Soilmoisture,v_Temperature,v_Humidity]]).astype(np.int64)

return data

cloudfactor_label={'Open (0 to 2 Oktas)':0,'Broken (3 to 7 Oktas)':1,'Closed (8 to 9 Oktas)':2}
soilmoisture_label={'Adequate (70% to 90%)':0,'Inadequate (Below 70%)':1}
temperature_label={'Moderate (25C to 30C)':0,'Minimum (15C to 25C)':1,'Maximum (30C to 35C)':2}
humidity_label={'Moderate(70%rh to 90%rh)':0,'Minimum (50%rh to 70%rh)':1,'Maximum (90%rh to 100%rh)':2}

#get the value
def get_value(val,my_dict):
    for key,value in my_dict.items():
        if val==key:
            return value

def main():
    html_temp = """
    <div style="background-color:#006400;padding:10px">
    <h2 style="color:white;text-align:center;">Powdery Mildew Prediction ML App
    </h2>
    </div>
    """

    st.markdown(html_temp, unsafe_allow_html=True)

```

```

input=inputs()

if st.button("predict"):
    prediction=model.predict(input)
    pred='{0}'.format(prediction[0])
    if pred == '0':
        st.header("Chances of getting Powdery Mildew disease is Low(20% to 40%)")
    elif pred == '1':
        st.header("Chances of getting Powdery Mildew disease is Medium(40% to 70%)")
    else:
        st.header("Chances of getting Powdery Mildew disease is High(70% to 90%)")

    st.subheader("Precautions Or Treatment to avoid Powdery Mildew:")

    st.markdown("BAKING SODA SOLUTION: Mix 1 tablespoon baking soda and ½ teaspoon liquid soap such as Castile soap (not detergent) in 1 gallon of water. Spray liberally, getting top and bottom leaf surfaces and any affected areas. This method may work better as a preventative measure, although it does have some effect on existing powdery mildew as well.")

    st.markdown("POTASSIUM BICARBONATE: Mix 1 tablespoon potassium bicarbonate and ½ teaspoon liquid soap (not detergent) in 1 gallon of water. Spray liberally to all affected areas. This mixture may work better than baking soda.")

    st.markdown("NEEM OIL: By itself, neem oil has mixed reviews on its effectiveness to treat powdery mildew, but it can be added to the above mixtures for an extra boost.")

    st.markdown("POWDERY MILDEW FUNGICIDE: Use sulfur-containing organic fungicides as both preventive and treatment for existing infections.")

    st.markdown("TRIM OR PRUNE: Remove the affected leaves, stems, buds, fruit or vegetables from the plant and discard. Some perennials can be cut down to the ground and new growth will emerge. Do not compost any damaged or diseased foliage as the spores can spread and persist in the composted material. Disinfect pruners and all tools after using on infected plants.")

if __name__ == '__main__':

```

main()

REFERENCES

1. [1] Annabel, L. S. P., Annapoorani, T., & Deepalakshmi, P. (2019, April). Machine Learning for Plant Leaf Disease Detection and Classification—A Review. In 2019 International Conference on Communication and Signal Processing (ICCSP) (pp. 0538-0542). IEEE.
2. Bhatia, A, Chug, A., and Singh, A.P. "Hybrid SVM-LR Classifier for Powdery Mildew Disease Prediction in Tomato Plant." In 2020 7th International Conference on Signal Processing and Integrated Networks (SPIN), pp. 218-223. IEEE, 2020.
3. Noa Schor, Avital Bechar, Timea Ignat, Aviv Dombrovsky, Yigal Elad, and Sigal Berman," Robotic Disease Detection in Greenhouses: Combined Detection of Powdery Mildew and Tomato Spotted Wilt Virus", IEEE, Dec 2015
4. Usama Mokhtar, Mona A. S. Ali, "Tomato leaves diseases detection approach based on support vector machines", 2015
5. Pranjali B. Padol, S. D. Sawant, "Fusion Classification Technique Used to Detect Downy and Powdery Mildew Grape Leaf Diseases", 2016