PLANT DISEASE DETECTION AND CLASSIFICATION

PROJECT REPORT

Submittedby

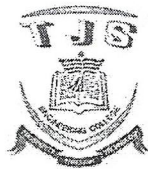|	
AKASH TIWARI.N	**(112819104003)**
MOHAN RAJ.S	**(112819104026)**
NAGARAJ.V	**(112819104032)**
SURESH CHELLADURAI	**(112819104056)**

In partial fulfilment for the award

of the degree Of

BACHELOR OF ENGINEERING

In

COMPUTER SCIENCE AND ENGINEERING

T.J.S.ENGINEERINGCOLLEGE,PERUVOYAL

ANNAUNIVERSITY:CHENNAI600025

APRIL 2022

ANNAUNIVERSITY:CHENNAI60025

BONAFIDECERTIFICATE

Certificatethatthisprojectreport**"PLANT DISEASE DETECTION AND CLASSIFICATION"**bonafideworkofthefollowing students.

AKASH TIWARI.N	(1128191
MOHAN RAJ.S	(1128191
NAGARAJ.V	(1128191
SURESH CHELLADURAI	(1128191(

Who carried out the project work under my supervision
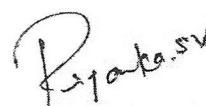
SIGNATURE SINATURE

Dr.S.ANBU,M.E.,Ph.D.,   Mrs.S.V.PRIYANKA.,M.E(C

HEAD OF THE DEPARTMENT   SUPERVISOR

Department of Computer Science   Department of Computer
Engineering   science and Engineering,

T.J.S. Engineering College,   T.J.S. Engineering college

Peruvoyal   peruvoyal.

Submittedforvivavoceheldon___18 / 06 / 22___atT.J.S.EngineeringCollege,

INTERNALEXAMINER   EXTERNALEXAMI
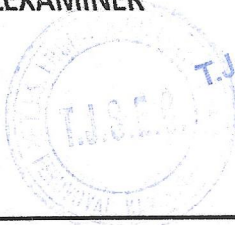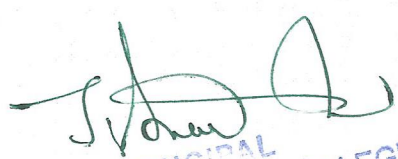
2

ABSTRACT

Crop diseases are a noteworthy risk to sustenance security, however their quick distinguishing proof stays troublesome in numerous parts of the world because of the non attendance of the important foundation. Emergence of accurate techniques in the field of leaf-based image classification has shown impressive results.

This paper makes use of Random Forest in identifying between healthy and diseased leaf from the data sets created. Our proposed paper includes various phases of implementation namely dataset creation, feature extraction, training the classifier and classification.

The created datasets of diseased and healthy leaves are collectively trained under Random Forest to classify the diseased and healthy images. For extracting features of an image we use Histogram of an Oriented Gradient (HOG).

Overall, using machine learning to train the large data sets available publicly gives us a clear way to detect the disease present in plants in a colossal scale.
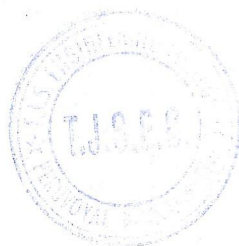
4

OBJECTIVES:
- To know the basics of algorithmic problem solving
- To read and write simple Python programs.
- To develop Python programs with conditionals and loops.
- To define Python functions and call them.
- To use Python data structures -- lists, tuples, dictionaries.
- To do input/output with files in Python.

UNIT I   ALGORITHMIC PROBLEM SOLVING   9

Algorithms, building blocks of algorithms (statements, state, control flow, functions), notation (pseudo code, flow chart, programming language), algorithmic problem solving, simple strategies for developing algorithms (iteration, recursion). Illustrative problems: find minimum in a list, insert a card in a list of sorted cards, guess an integer number in a range, Towers of Hanoi.

UNIT II   DATA, EXPRESSIONS, STATEMENTS   9

Python interpreter and interactive mode; values and types: int, float, boolean, string, and list; variables, expressions, statements, tuple assignment, precedence of operators, comments; modules and functions, function definition and use, flow of execution, parameters and arguments; Illustrative programs: exchange the values of two variables, circulate the values of n variables, distance between two points.

UNIT III   CONTROL FLOW, FUNCTIONS   9

Conditionals: Boolean values and operators, conditional (if), alternative (if-else), chained conditional (if-elif-else); Iteration: state, while, for, break, continue, pass; Fruitful functions: return values, parameters, local and global scope, function composition, recursion; Strings: string slices, immutability, string functions and methods, string module; Lists as arrays. Illustrative programs: square root, gcd, exponentiation, sum an array of numbers, linear search, binary search.

UNIT IV   LISTS, TUPLES, DICTIONARIES   9

Lists: list operations, list slices, list methods, list loop, mutability, aliasing, cloning lists, list parameters; Tuples: tuple assignment, tuple as return value; Dictionaries: operations and methods; advanced list processing - list comprehension; Illustrative programs: selection sort, insertion sort, mergesort, histogram.

UNIT V   FILES, MODULES, PACKAGES   9

Files and exception: text files, reading and writing files, format operator; command line arguments, errors and exceptions, handling exceptions, modules, packages; Illustrative programs: word count, copy file.

                                                        TOTAL: 45 PERIODS

OUTCOMES:

Upon completion of the course, students will be able to
- Develop algorithmic solutions to simple computational problems
- Read, write, execute by hand simple Python programs.
- Structure simple Python programs for solving problems.
- Decompose a Python program into functions.
- Represent compound data using Python lists, tuples, dictionaries.
- Read and write data from/to files in Python Programs.
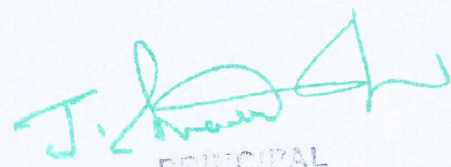
TEXT BOOKS:
1. Allen B. Downey, ``Think Python: How to Think Like a Computer Scientist", 2nd edition, Updated for Python 3, Shroff/O'Reilly Publishers, 2016
2. Guido van Rossum and Fred L. Drake Jr, ¯An Introduction to Python – Revised and updated for Python 3.2, Network Theory Ltd., 2011.

REFERENCES:

1. John V Guttag, ¯Introduction to Computation and Programming Using Python", Revised and expanded Edition, MIT Press , 2013
2. Robert Sedgewick, Kevin Wayne, Robert Dondero, ¯Introduction to Programming in Python: An Inter-disciplinary Approach, Pearson India Education Services Pvt. Ltd., 2016.
3. Timothy A. Budd, ¯Exploring Python‖, Mc-Graw Hill Education (India) Private Ltd.,, 2015.
4. Kenneth A. Lambert, ¯Fundamentals of Python: First Programs‖, CENGAGE Learning, 2012.
5. Charles Dierbach, ¯Introduction to Computer Science using Python: A Computational Problem-Solving Focus, Wiley India Edition, 2013.
6. Paul Gries, Jennifer Campbell and Jason Montojo, ¯Practical Programming: An Introduction to Computer Science using Python 3‖, Second edition, Pragmatic Programmers, LLC, 2013.