MACHINE LEARNING-BASED DECISION SUPPORT FOR DETECTING MAIZE PLANT DISEASE

BELLO ABRAHAM ITEOLUWAKISI

18010301012

SCIENCE AND MATHEMATICS, COLLEGE OF BASIC AND APPLIED SCIENCES

IN FULFILMENT OF THE REQUIREMENTS FOR THE

AWARD OF DEGREE OF BACHELOR OF SCIENCE (B.Sc.)

IN

COMPUTER SCIENCE

MOUNTAIN TOP UNIVERSITY, IBAFO

OGUN STATE, NIGERIA

2022

DECLARATION

I hereby declare that i am the author of this project and that it is an account of my own study. It has never been submitted with a prior application for a higher degree from this university or another. References are provided for all citations and sources used in this study.

____

BELLO ABRHAM ITEOLUWAKSI

____

Date

CERTIFICATION

This is to certify that this project, machine learning based decision support for detecting maize plant disease was carried out by me Bello Abraham Iteoluwakisi (Matriculation Number: 18010301012) and duly supervised by Dr D.D Aleburu.

______________________________(Signature and Date)

Dr Deborah D. Aleburu

(Supervisor)

___________________________________(Signature and Date) Dr Matthew O. Adewole

Coordinator, Department of Computer Science and Mathematics

DEDICATION

This project writes up is dedicated to God who in His infinite love and mercy teaches, guides, protects, and leads me and has given me strength, protection, care, love and wisdom throughout the time, may all the glory be His. And my beloved parents, brother and sisters, friends and well-wishers and all who have stood by supported and encouraged me throughout this period.

ACKNOWLEDGEMENT

I will be eternally thankful to my parents, my sister Bunmi Ayangbile for her love and support during my time at Mountain Top University.

I'd also want to thank my supervisor, Dr. D.D. Aleburu, for her guidance, comments, and recommendations that helped me finish my research.

To all of my excellent professors, members of staff in the Department of Computer Science, and the college as a whole, you have all been the best during my four years here; may God continue to bless you in all your endeavours.

Thank you especially to all of my friends. I am appreciative for the time and knowledge provided to guarantee the completion of this project. I adore you. God bless all of you.

Above all, I want to convey my heartfelt thanks to Almighty God, the source of all life and my Saviour, for everything.

ABSTRACT

Maize is an essential crop for us humans, it can be both used in the manufacturing industry and also for foods, etc. The goal of this project is to develop a machine learning-based decision support system for detecting maize plant disease.

In order to achieve this, it was necessary to analyse and review machine learning techniques and algorithms best for detecting maize disease. Google colab was used to build the model to detect the disease. The model was built to reduce cost and time of detecting maize plant disease.

The model employs a convolutional neural network to identify specific diseases in maize and was able to get a satisfactory result.

The model was able to accurately detect the disease, this improves the rate at which maize crop loss can be reduced due to infections or diseases which have infected the plants.

Keywords: Machine learning (ML), Decision support system, Convolutional Neural Network (CNN)

TABLE OF CONTENT

LIST OF TABLES

LIST OF FIGURES

1.1 Background to the study

Maize is a part of the vital crops in the globe following rice and wheat (IITA, 2005). The crop remains one of the principal grains farmed in the sub-savanna region of Nigeria (Fatima & Adbul, 2005). For several million people in Africa, maize has become one of the main sources of carbohydrates and is a component of animal feed (Romney, Thorne, Lukuyu, & Thornton, 2003). Since it is always one of the first crops gathered for food during a time of need, maize is indeed quickly replacing other crops as the most favoured one. The crop has always received preferential treatment above all other crops, even cassava (Fakorede, 2001). Maize is known to produce considerable quantities of vitamins A and C when consumed in the immature stages. In Nigeria, maize production has gradually increased to industrial levels, with many agro-based enterprises dependent on it as an organic material. As a grain crop, its importance is mainly on the economic value and not on the number of farmers engaged in the cultivation (Iken & Amusa, 2004). 10.2 million tonnes of maize are being produced in the nation, grown on 4.9 million hectares of land, with an average production rate of 2.1 tmm/ha (FAO, 2018). This data places Nigeria behind other African nations like South Africa, which yield greater tons on a less amount of land.

A fundamental difficulty in agricultural economies is the restricted ability to detect infections, resulting in inadequate crop restoration efforts. Visual inspection was mostly employed to pinpoint diseases in plant at the place where they were cultivated, with the diseased plant being contrasted to a healthy one. Plant height, color, leaf form, and leaf density on the branches, as well as alterations in the root system, are all noticed in this example. If there are disease, the appearance of mycelium, sporangiophore, or sclerotia, if any, indicates the presence of a pathogenic fungus.

1.2 Statement of Problem

Plant disease lowers the quantity and efficiency of food, fiber, and biofuel crops as agriculture tries to feed the world's fast expanding population. Losses may be catastrophic or recurring, but they typically account for 42% of the six most important food crops' output. Post-harvest losses can be severe, especially if farms are located far from markets and have inadequate infrastructure and supply chain management. Many infections that are present after harvest also release poisons that can seriously harm consumers' health.

Farmers spend thousands of dollars managing maize disease, sometimes without enough technical support, which has detrimental effects such as inadequate disease control and contamination. In addition, the disease can interfere with the growth and cause damage to cultivated and naturally growing plants

This project focuses at improving maize food security by diminishing crop risk caused by diseases, using machine learning techniques.

1.3 Aim and Objectives

The aim of this project is to design and determine Machine Learning based Decision Support System for detecting maize disease, the specific objectives are to:

I. Review ML techniques and determine the best for detecting maize plant disease
II. Design and implement a model for detecting diseases in maize plant
III. Test and evaluate the model

1.4 Methodology

In order to have an excellent idea of the topic, and achieve the objectives the following steps will be taken.

I. Review ML techniques and determine the best for detecting maize plant disease

  - Researches and studies that have been made based on maize plan disease will be reviewed
  - Machine learning techniques that have been used in similar studies will be studied. Limitations and accuracy involving the machine learning techniques will be reviewed, therefore the machine learning technique considered to be the best will be used to detect diseases in maize plant in this project

II. Design and implement a model for detecting diseases in maize plant.
  - A dataset containing maize diseases images will be acquired.
  - The dataset will be used for training and to create the training data.
  - Test images will be classified according the disease.
  - The model will be compiled and trained.

III. Test and evaluate the model
  - The trained model will be tested in maize plant disease detection.
  - The result of the test will be shown.

1.5 Research Scope

The scope of the project focuses on the detection of maize plant disease using dataset image consisting of three diseases which are maize blight, maize common rust, maize gray leaf spot. The machine learning technique been used is CNN.

1.6 Significance of Study

Maize disease affects a significant number of the production and health of the plant considering people, something must be done by designing a system for detecting maize disease. When finding from this study are implemented, it will help farmers and also white-collar persons who are interested in having a maize farm, to solve the problem associated with inability to keep producing effectively maize plant due to disease.

1.7 Definition of Terms

i. Decision support system: A decision support system (DSS) is a computer software that aids in making decisions, rendering judgements, and selecting courses of action within a company or organization.

ii. TensorFlow is a free software library for artificial intelligence and machine learning. Although it may be used too many different tasks, deep neural network training and inference are given special attention.

iii. Convolutional Neural Network (CNN) is a Deep Learning method that can take in an input picture, give various elements and objects in the image importance (learnable weights and biases), and be able to distinguish between them.

iv. Machine Learning: Machine learning is the study of, and development of, "learning" methods, or methods that use data to enhance performance on a given set of tasks. Considered to be a component of artificial intelligence

v. Google colab: Collaboratory, sometimes referred to as "Colab," is a product of Google Research. Colab is particularly well suited to machine learning, data analysis, and teaching. It enables anybody to create and execute arbitrary Python code through the browser. Technically speaking, Colab is a hosted Jupyter notebook service that offers free access to computer resources, including GPUs, and requires no setup to use.

CHAPTER TWO
LITERATURE REVIEW

2.1 Decision Support System

DSS which is also known as Decision Support System is a type of system information that aids in commercial or institutional decision-making DSSs support an overall risk management process, administrative, and planning levels. (Probably junior and senior administration) and accommodate people pass judgments concerning challenges It may be continually developing and hard to define in advance—i.e., decision issues that are both unplanned and moderately planned. DSS can be wholly programmed, operated by individuals, or a blend of the two. (Wikimedia Foundation, 2011). Management of operations and different planning sector in a business can use the DSS to aggregate facts and figures and synthesis it into actionable insight. In reality, mid- to upper-level management are the main user of these technologies.

A DSS, for example, could be used to figure out a sales profits for the upcoming seven months utilizing new product sales estimates. This isn't a simple calculation which could be conducted manually due to the huge number of factors that accompany expected income statistics. A DSS, in contrast side, may integrate all of the many variables and provide an outcome and substitute outcomes reflecting on the company's prior product sales figures and current market circumstances.

A DSS can also be designed to just about any sector, specialty, or field, including healthcare, government institutions, agricultural issues, and commercial activities. Many different industries, from medicine to agriculture, use decision support systems. To help diagnose a patient, a medical clinician may use a computerized decision support system for diagnostics and prescriptions. A DSS can assist a doctor to diagnose a patient by incorporating human inputs with pertaining electronic health data.

2.1.1 Characteristics of a Decision Support System

The purpose of adopting a DSS is just to convey information in an understandable format. A DSS system has advantages because it has the ability designed to create a wide range of reports according to the user's preferences. The DSS, for instance, can develop or present data graphically, such as in a bar chart illustrating anticipated revenue or as a formal report.

Data analysis is no longer restricted to massive, unwieldy mainframe systems as technology advances. Since a DSS is simply a program, it can be installed on most computer systems, including desktops and laptops. Certain DSS programs can also be opened through any mobile device.

The DSS's flexibility is incredibly beneficial for individuals who travel a lot. This helps give people the possibility to be updated at any and all times, providing them the ability to make the best decisions for their company and customers on the go or even on the spot

2.1.2 Development Frameworks

Similarly, to other systems, DSS systems require a structured approach. Such a framework includes people, technology, and the development approach. (Sprague & E.D, 1982)

The DSS Initial Framework is split into four phases:

- Intelligence
- Design
- Choice
- Implementation

The hardware and software technological layers of a DSS are:
:

1. The real program that the user will use. This section of the application enables the decision maker to make decisions in a specific issue area. The user can take action on that specific issue.

2. The generator includes a hardware/software system that allows users to quickly construct unique DSS applications. Application methods or platforms like as Crystal, Analytica, and iThink are used at this level.

3. Tools are low-level hardware/software. Special languages, function libraries, and linking modules are examples of DSS generators.

The DSS can be modified and redesigned at different intervals using an iterative development technique. Again when the process has been created, it must be tested and altered as necessary to get the expected goal.

2.1.3 Classification of Decision Support System

There are several ways to classify DSS applications. Not every DSS fits neatly into one of the categories, but may be a mix of two or more architectures. Holsapple and Whinston (Holsapple & A.B, 1996) classify DSS into the following six frameworks: text-oriented DSS, database-oriented DSS, spreadsheet-oriented DSS, solver-oriented DSS, rule-oriented DSS, and compound DSS. A compound DSS is the most popular classification for a DSS; it is a hybrid system that includes two or more of the five basic structures (Holsapple & A.B, 1996).

The support given by DSS can be separated into three distinct, interrelated categories: (Hackathorn & P.G, 1981, September) Personal Support, Group Support, and Organizational Support.

DSS components may be classified as:

1. Inputs: Factors, numbers, and characteristics to analyse
2. User knowledge and expertise*:* Inputs requiring manual analysis by the user
3. Outputs: Transformed data from which DSS "decisions" are generated
4. Decisions: Results generated by the DSS based on user criteria

DSSs which perform selected cognitive decision-making functions and are based on artificial intelligence or intelligent agent's technologies are called intelligent decision support systems (IDSS), (F & C.W, 2008) .

The nascent field of decision engineering treats the decision itself as an engineered object, and applies engineering principles such as design and quality assurance to an explicit representation of the elements that make up a decision.

2.1.4 Applications of Decision Support System

In theory, DSS can be implemented in any particular project. A clinical DSS for clinical diagnosis is one example. The advancement of a clinical DSS is split into four stages: The basic version is self-contained and thus does not adopts integration; the subsequent generation supports integration with other medical systems; the third is standard-based, and the fourth is service model-based (Wright & Sittig, 2008).

Businesses and management broadly apply DSS in its use case. Executive boards and other corporate performance analytics offer rapid strategic planning, recognition of negative developments, and effective budget allocation. DSS guarantees that certain data about any business is shown in the graphical manner. i.e., in a summarized way, which helps the management to take strategic decisions. For example, one of the applications is the management and development of complex anti-terrorism systems (Zhang & Babovic, 2011). Additional possibilities are a business loan officer assessing a loan applicant's credit or an engineering business that has placed on multiple projects and wanting to verify if their expenses are comparable.

Agriculture, marketing for ecological sustainability, is a rising field of DSS implementation, ideas, methods, and strategies. Agricultural DSSes began to be evolve and promoted in the 1990s (Papadopoulos, Shipp, Jarvis, Jewett, & Clarke, 1 July 1995). For instance, the DSS for Agricultural technology Transfer package, created with funding from USAID in the 1980s and 1990s, has enabled quick evaluation of a variety of agricultural manufacturing processes systems around the world to facilitate decision-making at the farm and policy levels. By using accuracy in farming, decisions may be made specifically for selected fields. The proper implementation of DSS in agro, however, is hampered by several issues.

2.2 Review of Artificial Intelligence

Artificial intelligence (AI), often referred to as machine learning in computer science, is information conveyed by computers as against natural intelligence exhibited by people. A reproduction of human processes and discernment by machines, especially a robot that is computer-controlled, is termed artificial intelligence (AI). These include reasoning (using rules to arrive at approximations or firm conclusions), self-correction, and learning (acquiring information and rules for applying it). Object recognition, speech recognition, and expert systems are some key implementations of AI.

Artificial intelligence (AI) application examples

Many distinct forms of technology that include artificial intelligence (AI):

1. Automation: A system or procedure that runs without human intervention. For instance, robotic process automation may be taught to carry out repetitive, high-volume operations that would often be handled by people. RPA vary away from IT automation in that it is flexible in response to changing conditions.

2. Robotics: An engineering discipline devoted to the creation of robots. Robots are frequently utilized to complete jobs that are challenging for humans to complete or consistently complete. They are employed by NASA to move big items in orbit or on auto assembly lines for the manufacture of automobiles. Machine learning is also being used by researchers to create socially intelligent robots.

3. Self-driving cars: Vehicles utilize a mix of object recognition, image recognition, and deep learning to develop automatic proficiency at vehicle piloting while remaining in a set lane and avoiding unforeseen obstacles, such as pedestrians. Machine learning: The science of getting a computer to act without programming. In simplest words, deep learning is the automation of predictive analytics. It is a class of machine learning. Machine learning algorithms come in three different varieties:

   I. Supervised learning: Data sets are labelled so that patterns can be detected and used to label new data sets

   II. Unsupervised learning: are sorted by similarities or differences without labels.

   III. Reinforcement learning: Data sets aren't labeled, yet an AI system receives feedback after carrying out one or more actions.

4. 4. Machine vision: Studies how computers to perceive. Using a camera, analog-to-digital conversion, and digital signal processing, this device records and examines visual data. Though typically equated to human vision, machine vision is not constrained by biology and may be designed to, for instance, see through walls. It is deployed in a considerable number of fields, such as medical picture analysis and signature identification. Often confused with machine vision, computer vision focuses on automated image processing.

5. Natural language processing: Basically is just the method through which a computer software interprets human language rather than computer language. One of the first and most well-known applications of NLP is spam detection, which evaluates an email's subject line and body to determine if it is spam. The methods used in NLP nowadays are based on machine learning. Text translation, sentiment analysis, and speech recognition are examples of NLP tasks.

2.2.1 Machine Learning

The research of methods and simulation approaches that computers utilize to carry out a particular task is known as machine learning (ML). Without anyone directly programming it to do so, machine learning algorithms construct a statistical model using data samples, or "testing dataset," in order to arrive at predictions or judgments. Algorithms for machine learning are utilized in a broad range of applications, including email filtering and computer vision, when it is challenging or impractical to create a traditional algorithm that really can adequately complete the task.

Machine learning techniques:

ML algorithms are usually sorted as supervised or unsupervised

In order to anticipate future events, supervised ML systems can integrate what has been recognized in the past to fresh data using named guides. The learning method develops an induced capacity to form expectancies about the yield esteems starting with the evaluation of a recognized training dataset. After sufficient planning, the framework may focus on any new contribution. In order to fix errors and improve the model, the learning algorithm can also compare its output with the predicted, correct yield.

However, when preparing data that is neither described nor labelled, unsupervised ML methods are used. Unsupervised learning considers how algorithms might infer from unlabelled data the ability to portray a hidden structure. The framework investigates the data and therefore can create deductions from datasets to depict embedded information from unlabelled data, even if it cannot interpret the intended result.

Semi-managed ML algorithms integrate either unlabelled and labelled data to prepare frequently a little amount of labelled data and a vast volume of unlabelled data, which places in the middle of unsupervised and supervised learning. The frameworks that takes advantage of this method can significantly increase learning accuracy. Semi-regulated learning is most frequently chosen when the acquired identified data skilled and suitable assets are needed to prepare it/gain from it. The acquisition of unlabelled data, on the other hand, typically doesn't need for additional resources.

A learning method known as reinforcement learning involves designing tasks, looking for errors or rewards, and associating the result with the condition. The primary benefit of reinforcement learning include experimentation search and postponed reward. This method helps computers and programming experts to figure out the ideal behavior in a specific context in order to improve its presentation. The support signal—basic reward input—is necessary for the specialist to understand which action is best.

ML enables the analysis of vast volumes of data. While it often delivers speedier, more accurate results to identify advantageous opportunities or dangerous risks, it may also take more time and resources to prepare it properly. The preparation of immense quantities of dataset can become much more increasingly feasible by combining AI with AI and subjective improvements.

2.3 Machine learning Algorithms

2.3.1 Principal Component Analysis (PCA)/SVD

This is a fundamental machine learning algorithm. It enables you to minimize the dimension of the data while losing as little information as possible. It is used in a diverse range of applications, including object identification, computer vision, data compression, and so forth. The computation of the principle components is simplified to computing the eigenvectors and eigenvalues of the original data's covariance matrix or the data matrix's singular decomposition. We may express numerous indications via one, so to speak, and work with a simpler model already. Of course, it will be non-viable to completely eliminate information loss, but the PCA approach will assist us in minimizing it.

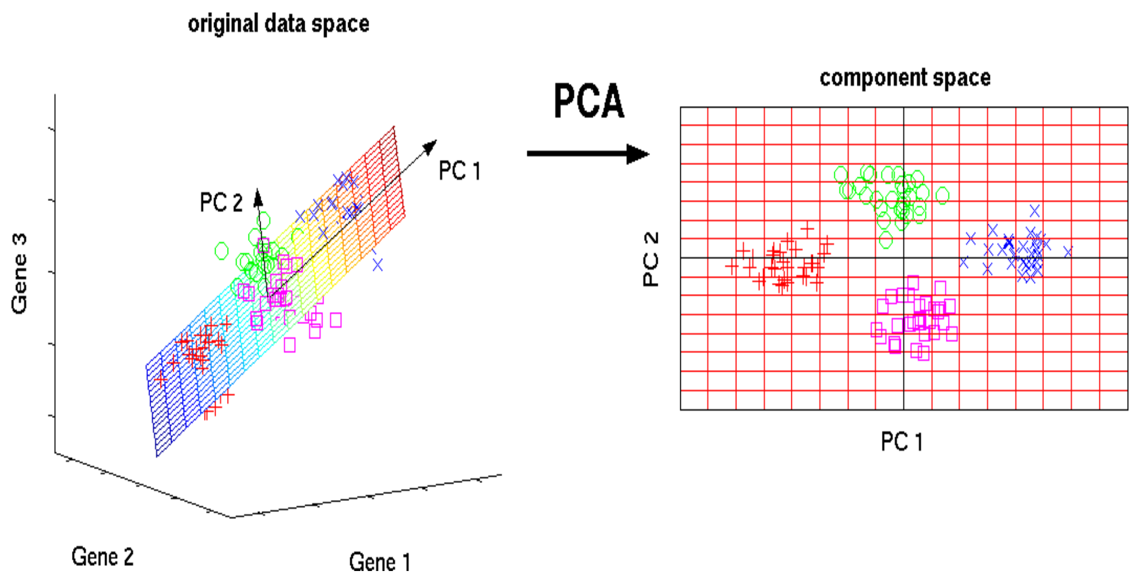


Figure 2-1 SVD

Source: (Kai Zhao, 2017)

2.3.2.1 Least Squares and Polynomial Fitting

The least squares approach is a mathematical strategy for solving various issues that is based on diminishing the sum of squares of some functions' deviations from the target variables. It may be used to "solve" over determined systems of equations (where the number of equations exceeds the number of unknowns), search for solutions in conventional (non-over determined) nonlinear systems of equations, and approximate the point values of a given function.

2.3.2.2 Constrained Linear Regression

Overshoots, false fields, and other issues can be confused by the least-squares approach. To lower uncertainty of the line we put in the set of data, we need to impose constraints. Matching the linear regression model assures that the weights do not behave "badly." L1 (LASSO) or L2 (Ridge Regression) models, or both, can be used (elastic regression).

2.3.3 K-Means Clustering

Clustering is the job of dividing a collection of items into groupings known as clusters. There should be "similar" things within each group, and the objects of separate groups should be as dissimilar as possible. The primary distinction between clustering and classification is that the list of groups is not explicitly specified and is determined during the algorithm's execution.

In the conventional implementation, the nearest centroid classifier algorithm is the simplest, but it is also the most incorrect clustering approach. It divides a vector space's set of components into k previously known clusters.

The technique attempts to reduce the standard deviation at each cluster's points. The main principle is that at each iteration, the center of mass for each cluster created in the previous step is recalculated, and the vectors are split into clusters again based on which of the recent improvements was closest in the chosen metric. When no cluster changes occur during any iteration, the algorithm ends.

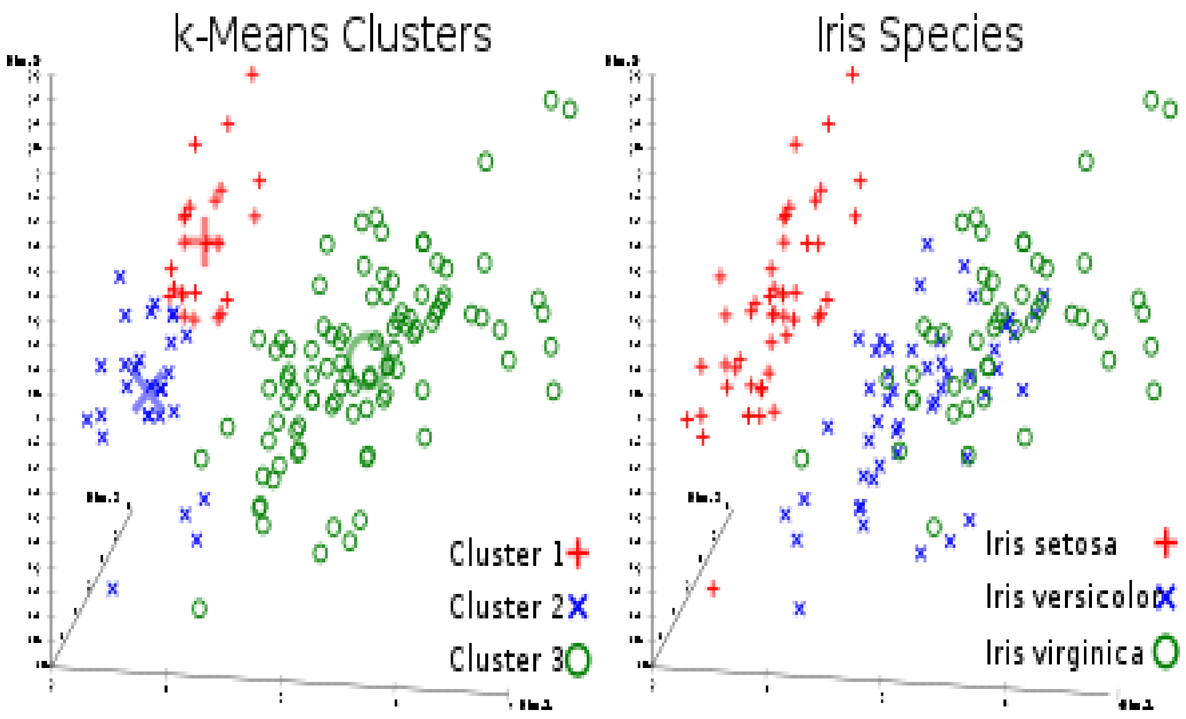


Figure 2-2: K-Means Clustering

Source: (Prasad Kulkarni, 2020)

2.3.4 Logistic Regression

After adding weights, logistic regression is confined to linear regression with non-linearity (sigmoid function or tanh is commonly employed), and hence the output limit is close to + / - classes (which equals 1 and 0 in the case of sigmoid). Gradient descent is used to optimize cross-entropy loss functions.

It is important to note that logistic regression is utilized for classification rather than regression. In general, it is similar to a single-layer neural network. Learned optimization techniques such as gradient descent and L-BFGS. It is frequently used by NLP developers and is referred to as the "maximum entropy classification approach."

2.3.5 Support Vector Machines (SVM)

SVM is a linear model, similar to linear/logistic regression. The distinction is that it features a margin-based loss function. You may optimize the loss function using optimization methods such as L-BFGS or SGD.

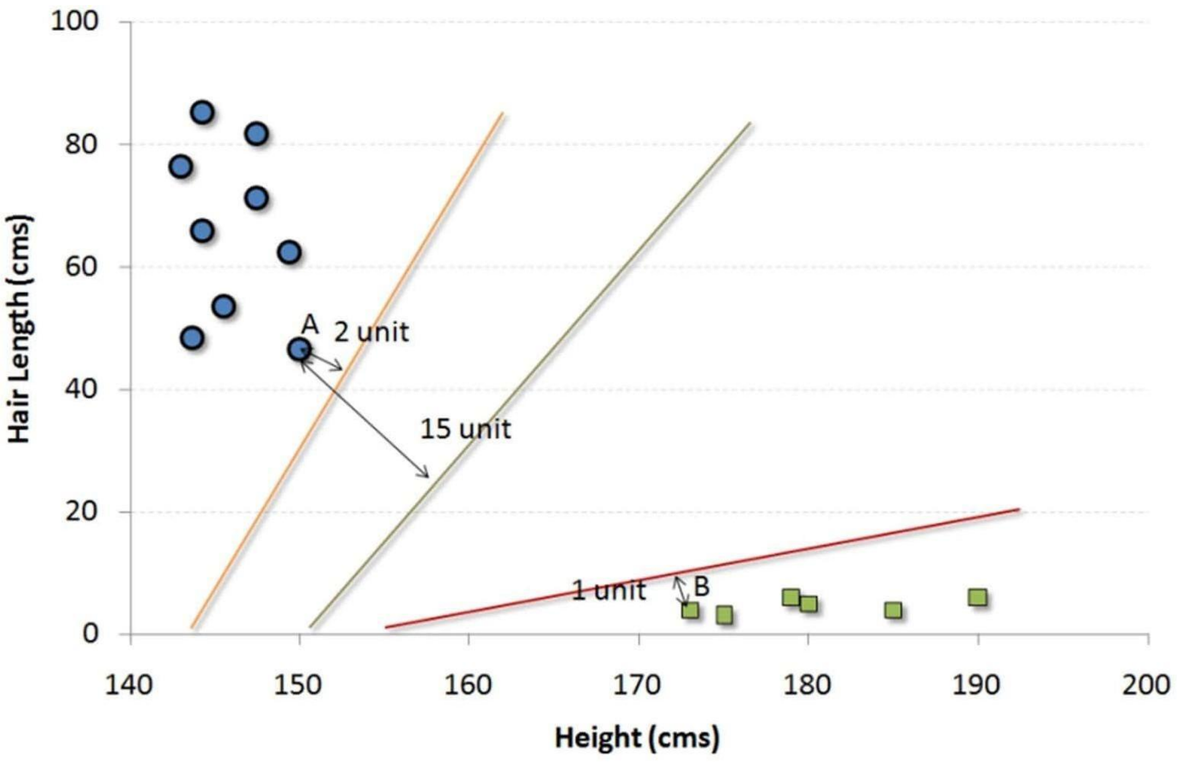


Figure 2-3: Support Vector Machine

Source: blog.ineuron.ai (Jay Singh)

2.3.6 Feed-Forward Neural Networks

These are basically multi-level logistic regression classifiers. Non-linearity separates several scale layers (sigmoid, tanh, relu + SoftMax, and the interesting new selu). They're also known as multilayer perceptrons. As auto encoders, FFNN may be used for categorizing and "learning without a tutor."

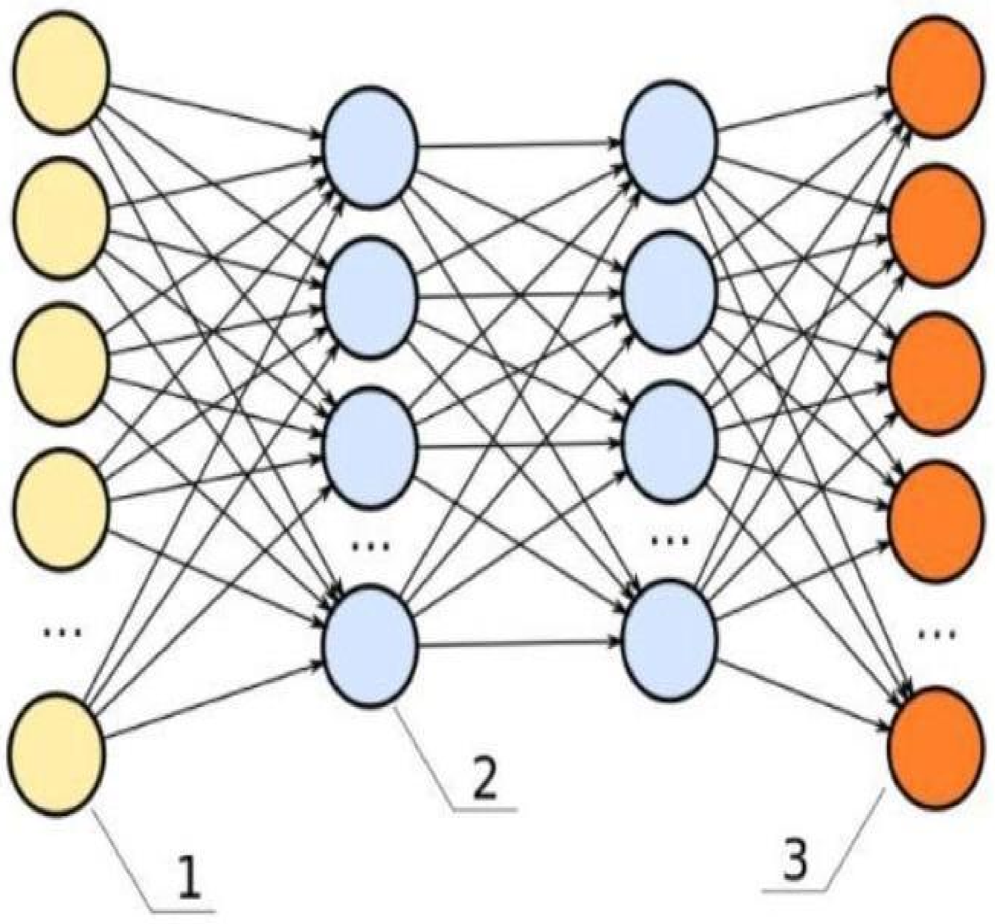


Figure 2-4: Forward Neural Network

Source: (Jiri Materna, 2019)

2.3.7 Recurrent Neural Networks (RNNs)

RNNs simulate sequences by repeatedly applying the very same weight set on the input and aggregator state at time t. Pure RNNs are no longer often utilized, but its equivalents, such GRU and LSTM, are the most current solutions for the majority of sequence modelling issues. LSTM is utilized in pure RNN in place of a straightforward thick layer. RNN may be used for any text categorization, machine translation, or language modelling problem.

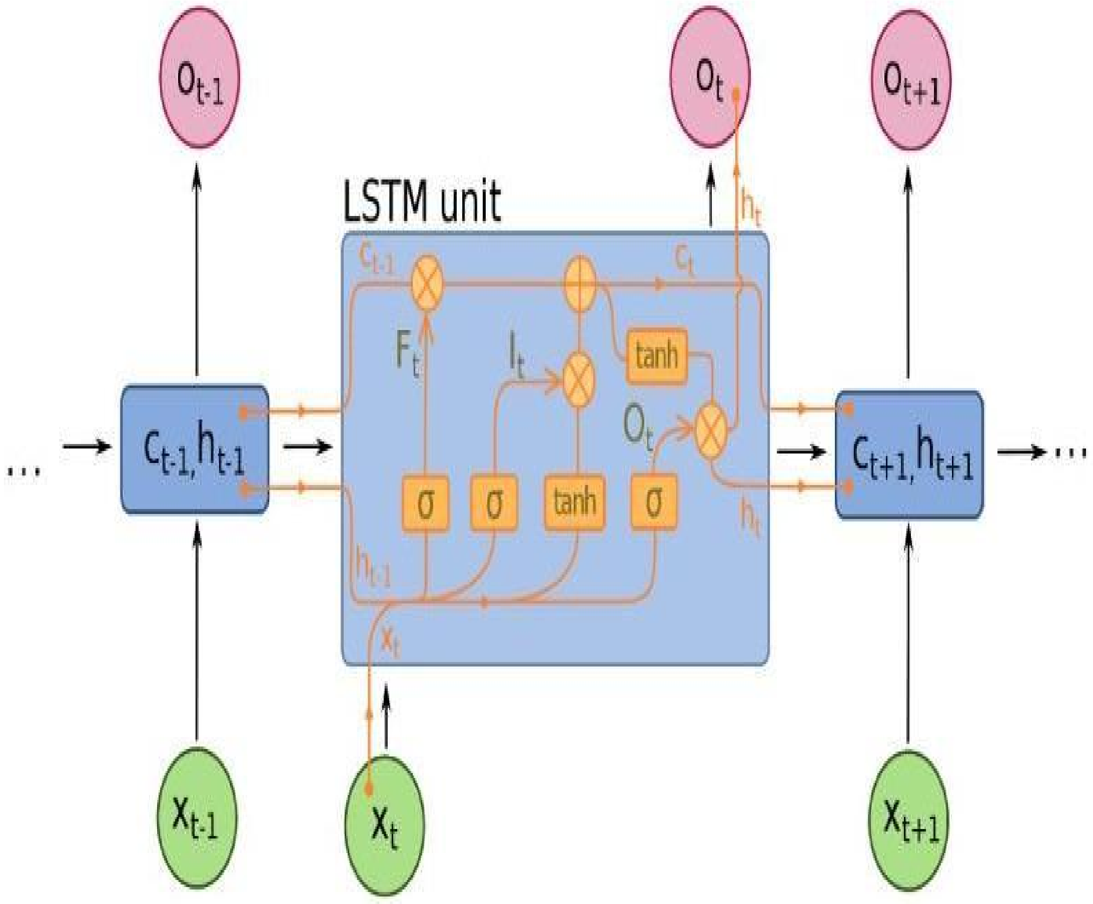


Figure 2-5: LSTM

Source: (Saul Dobilas, 2020)

2.3.8 Decision Trees

Among the most frequently used algorithms for machine learning, Used in predictive models for statistics and data analysis. The structure depicts the "leaves" and "branches." The characteristics of the goal function are determined by the "branches" of the decision tree, the values of the objective function are stored in the "leaves," and the remaining nodes include qualities for which the cases differ. To classify a new case, you need to go down the tree to the leaf and give the appropriate value. The goal is to create a model that predicts the value of the target variable based on several input variables.

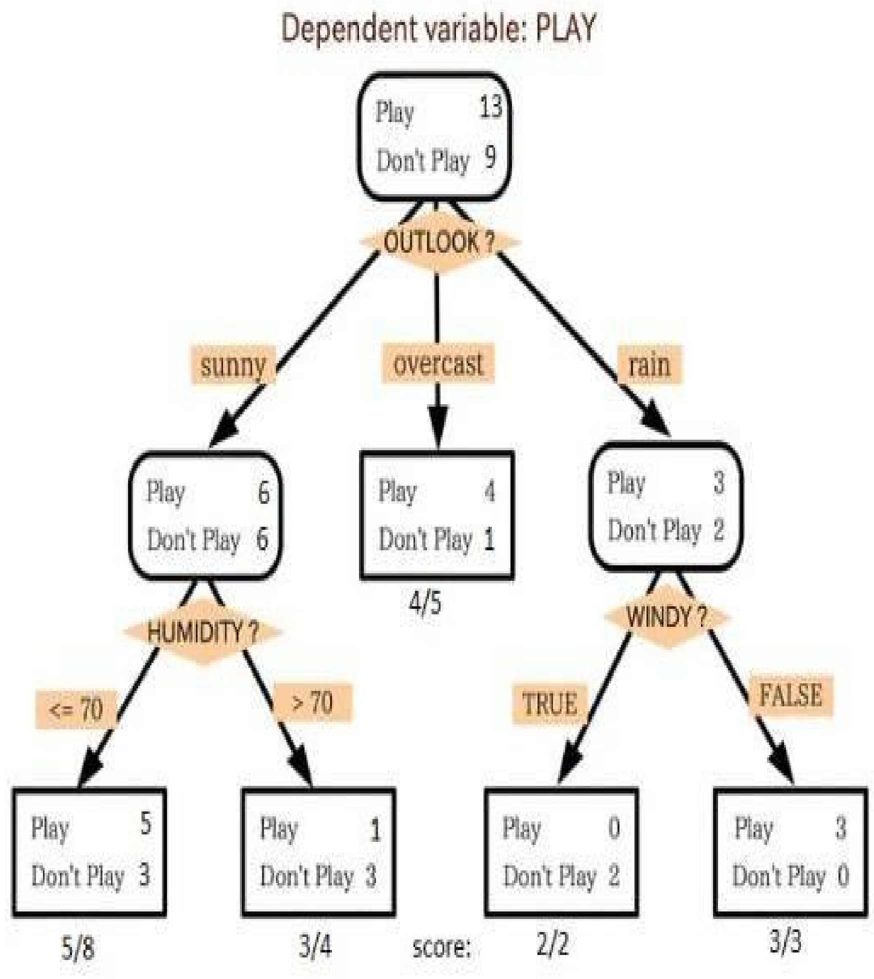


Figure 2-6: Decision Tree

Source: (Jay Singh, 2019)

2.3.9 Convolutional Neural Networks

Convolutional neural networks were largely responsible for all recent advances in machine learning. For image classification, object identification, or even picture segmentation, they are employed. Convolutional layers in networks, which were created by Jan Lekun at the start of the 1990s, serve as hierarchical object extractors. It can also be harnessed for text manipulation (and even for working with graphics).

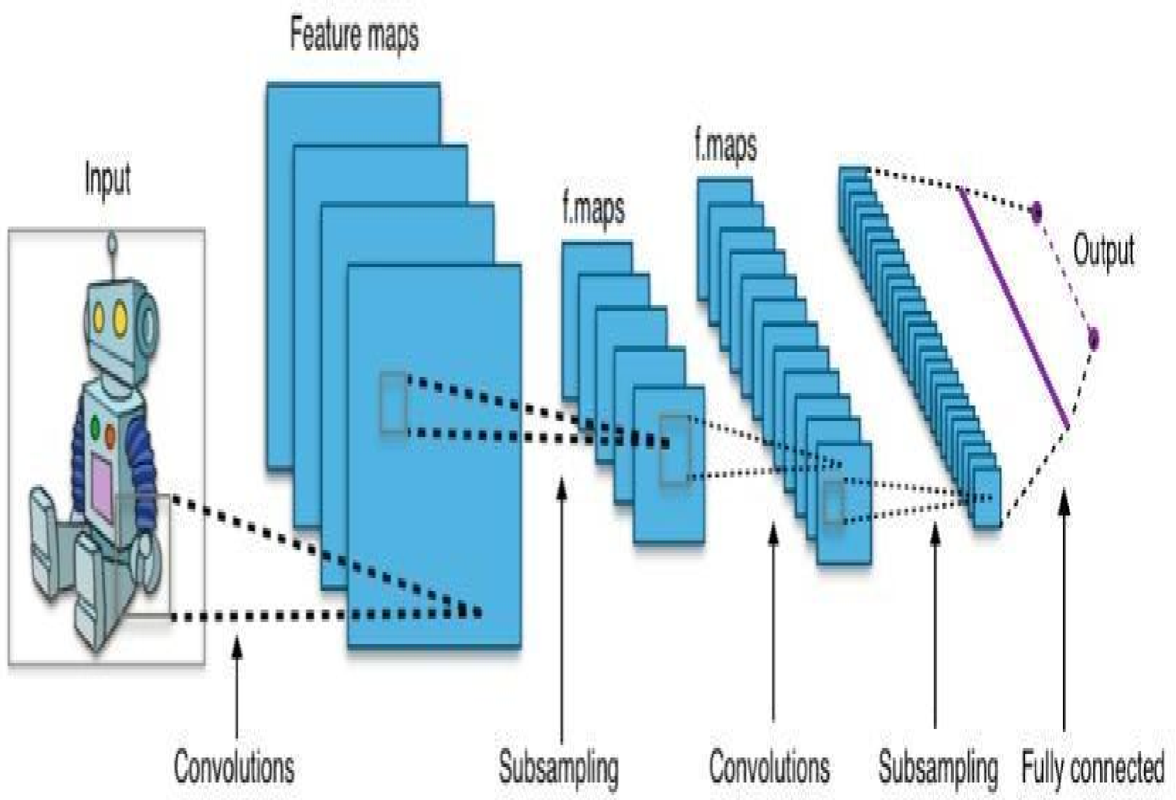


Figure 2-7: Convolutional Neural Network

Source: (Niklas Lang, 2021)

Comparatively speaking to other image classification algorithms, CNN's utilize a minimal amount of pre-processing. They are used in natural language processing, content - based recommendation systems, classification techniques, medical image analysis, and videos and images identification. A specific kind of linear processing is convolution. Convolutional networks are simple neural networks that, in at least one of their layers, employ convolution rather than standard matrix multiplication. Although they may be utilized with one-dimensional and three-dimensional data, convolutional neural networks, or CNNs for short, are a specific kind of neural network model created for working with two-dimensional picture data.

Depending on the picture resolution, computers interpret an input image as a repository of pixels. It will appear as h * w * d (h = Height, w = Width, d = Dimension) depending just on image quality. E.g., An image of a 4 x 4 x 1 array of a grayscale picture and an image of a 6 x 6 x 3 array of an RGB matrix (3 refers to RGB values).

Technically, each input picture is passed into a series of convolution layers with filters pooling, fully connected layers (FC), and the SoftMax function to identify an item with probabilistic values between 1 and 0. This is done in order to train and evaluate CNN models.

2.3.9.1 Convolution Layer

Convolution refers to the first layer which is utilized to extract features within an input image. By learning visual features from small squares of incoming data, convolution maintains the link amongst pixels. It's an arithmetical method that essentially needs two inputs: a filterorkernel and an image matrix. Convolution of an image with different filters has the potential to complete tasks such as edge detection, blur, and sharpening.

2.3.9.2 Strides

The stride reflects the amount of pixels travelled across the input matrix. When stride is set to 1, the filters are moved one pixel at a time. When stride is set to 2, the filters are moved two pixels at a time, and so on.

2.3.9.3 Padding

Occasionally, the filter does not precisely suit the input image. We have two choices:

i. Pad the photo with zeros to make it fit

ii. Pad the photo with zeros to make it fit.is known as valid padding, because it maintains just the legitimate parts of the picture.

2.3.9.4 Non-Linearity (ReLU)

Rectified Linear Unit is the abbreviation for a non-linear operation. The outcome is (x) = max (0, x). What is the significance of ReLU? ReLU's goal is to incorporate nonlinear characteristics into our ConvNet. Because real-world data would necessitate that our ConvNet learn non-negative linear values. In place of ReLU, numerous different non-linear functions such as tanh or sigmoid can be employed. The bulk of data scientists utilize ReLU because it beats the other two.

2.3.9.5 Pooling Layer

Whenever the photos are too immense, the pooling layers portion would lower the amount of variables. Spatial pooling, also known as down sampling, decreases the dimensionality of each map while retaining the essential information. There are several forms of spatial pooling:

1. Max Pooling
2. Average Pooling
3. Sum Pooling

The biggest element from the corrected feature map is used in max pooling. Choosing the biggest element might also mean selecting the average pooling. Sum pooling is the aggregate of all components inside the feature map.

2.3.9.6 Fully Connected Layer

The compressed matrix into a vector at the FC layer and routed it through a fully connected layer, akin to a neural network.

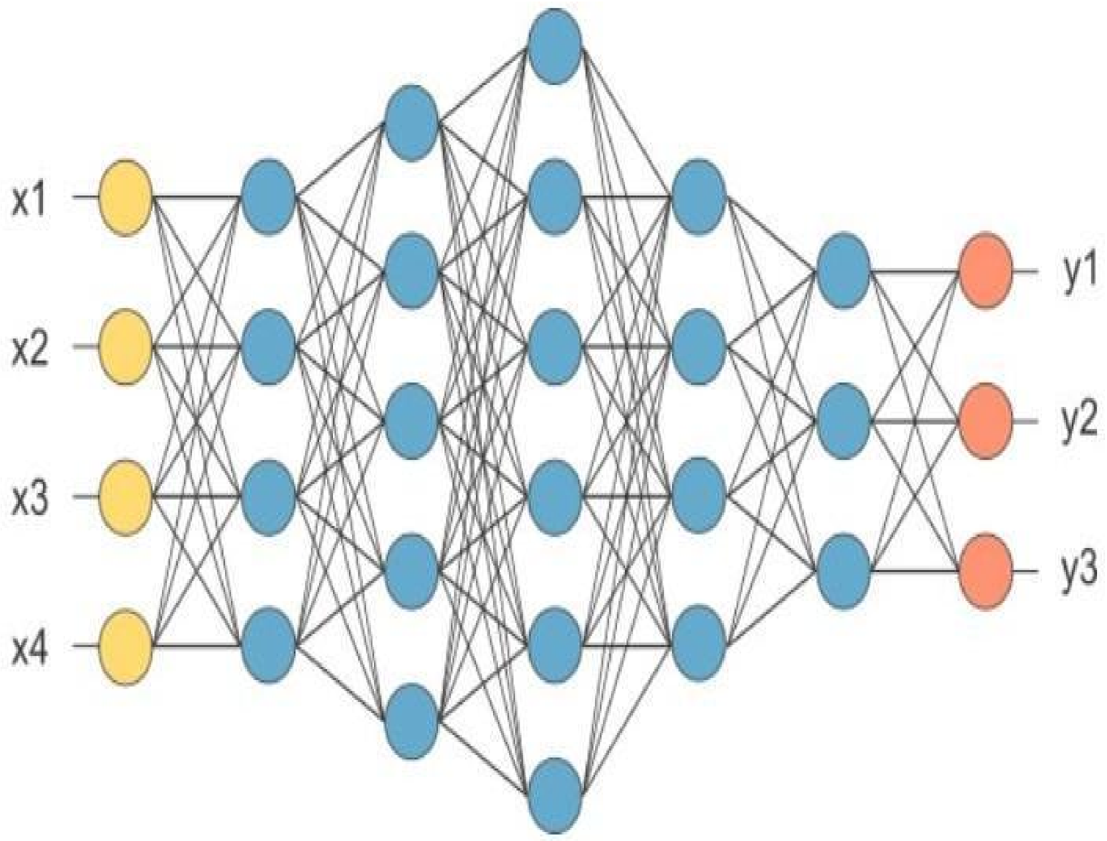


Figure 2-8: Fully Connected Layer

Source: (Kate Reyes, 2020)

The feature map matrix will be turned into a vector (x1, x2, x3...) in the picture above. We merged these characteristics into a model using fully linked layers. Finally, we have an activation function such as Soft-max or sigmoid to categorize the outputs as a cat, dog, automobile, truck, and so on.

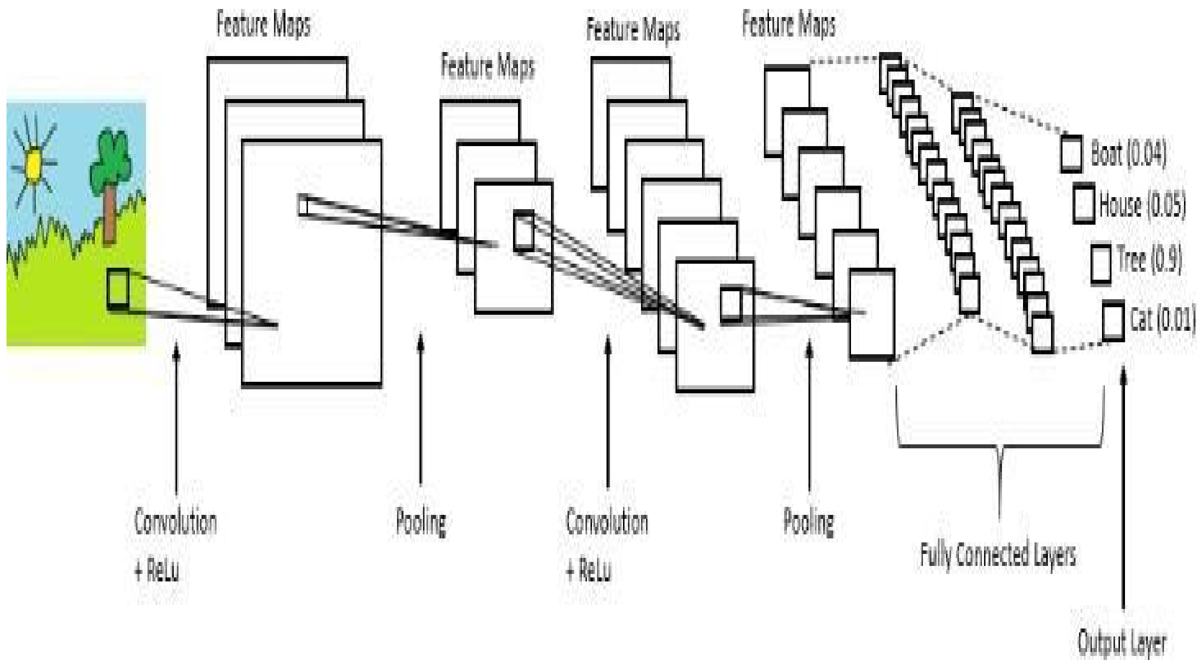


Figure 2-9: Convolutional Neural Network

Source: (Vegard Flovik, 2021)

2.4 Related Works

Image processing is implemented in agriculture for disease diagnosis and fruit grading (Monica, Ashwani, & Rushikesh, 2013). To detect disease, they had to use an ANN which stands for artificial neural network. They built two databases: one for training on previously reported disease photographs and another for implementing query images. The weights of training datasets are changed via back propagation. They examine three feature vectors: hue, textures, and morphologies (Monica, Ashwani, & Rushikesh, 2013). The morphological trait exceeds the other two features, they found.

In this work (Zulkifi, Abdul, Ali, Shakaff, & S, 2012), they photographed a chilli plant leaf and processed it to determine the health of the plant. Their technique guarantees that the pesticides are only administered to the infected chilli plant. They used MATLAB for extraction of features and image recognition. In this work, pre-processing is accomplished via the use of Fourier filtering, edge detection, and morphological procedures. To categorize objects, machine vision enhances the image processing methodology. The image is acquired through the use of a digital camera, as well as the GUI was created using the LABVIEW software tool. A leaf image's segmentation is crucial for obtaining a feature from it.

A comparison of the Otsu threshold and the k-means clustering technique for detecting diseased leaves in (Mrunalini & Prashant, 3, March 2012). They found that with k-means clustering, the retrieved values of the features are lower. The clarity is superior than other methods. The RGB picture is utilized for illness identification. The green pixels are recognized using k-means clustering algorithms, and the variable threshold value is achieved using otsu's approach. The color cross methodology is used for feature extraction. The RGB picture is transformed into the HSI translation. The SGDM matrix is constructed for texture statistics calculation, and the feature is computed using the GLCM function (H, S, M, M, & Z, March 2011).

Xiuqing Wang, Chunxia Zhang, and Xudong Li created an DSP and FPGA-based technology for monitoring and controlling plant diseases (Chunxia, Xiuqing , & Xudong, 2010). For monitoring and diagnostics, video data or image from the field plant is obtained using the FPGA. The video or picture data is processed and encoded using the DSP TMS320DM642. For data transport, the nRF24L01 single chip 2.4 GHz radio transmitter is employed. It features two data compression and transmission methods to fulfil the needs of various users, and it leverages multi-channel wireless connection to reduce overall system costs.

Table 2-1: Related Works

Author/Year	**Title**	**Research Objective**	**Methodology**	**Strength**	**Gap/Limitation**
Xiaolin Sun and Jianshu Wei, 2019	Identification of maize disease based on transfer learning	To demonstrate a method for identifying corn leaf diseases. to analyse and segment the picture, extract the image's colour, texture, form, and other attributes, then categorize it using an artificial neural network	Plant leaf diseases were diagnosed through Convolutional Neural Networks (CNN), and the CaffeNet model was modified using a fine-tuning approach, achieving exceptional results of identifying the disease.	It has accuracy of 81% in the third experiment	Small image dataset used for training the model.

Malusi Sibiya and Mbuyu Sumbwanyambe, 2019	Computational Procedure for the Recognition and Classification of Maize Leaf Diseases Out of Healthy Leaves Using Convolutional Neural Network	To use CNN in order to model a network for image recognition and classification of diseases.	Use of deep CNN built in the Neuroph studio to conduct a study to detect the three types of maize diseases that occurred in the large open maize fields.	Data gathered by the user from any angle of the leaves won't alter the outcome since it does away with the usage of calculation techniques and cameras for data collecting.	CNN's does not train and test Gray scale images.
Blake Richey, Sharmin Majumder, Mukul Shirvaikar, Nasser Kehtarnavaz, Nasser, 2020	Real-time detection of maize crop disease via a deep learning-based smartphone app	Introducing a CNN for both feature extraction and image Classification to identify specific diseases in crops.	Implementation of CNN and transfer learning on smartphone platforms for field deployment.	Classification and prediction diseased leaf are accurate	Misclassification of disease leaf with healthy leaf

Jatin Arora , Utkarsh Agrawal ,Prerna Sharma, , 2020	Classification of Maize leaf diseases from healthy leaves using Deep Forest	Using models that has been compared with traditional machine learning models such as SVM, random forest, Logistic regression, KNN and decision tree as well as deep models such as CNN and leNet5 to show its superiority	Implementation of multiple models applied on the given dataset, using RandomSearch, GridSearch, and other python functions to find the optimal configuration of each model.	Achieving the same accuracy with f1 score which maximum accuracy of 96.25% and maximum F1 score of 0.9624 among all other models .	

Sumita Mishra, Rishabh Sachan, Diksha Rajpal, 2020	Deep Convolutional Neural Network based Detection System for Real- time Corn Plant Disease Recognition	to demonstrate a deep learning system for identifying corn diseases in plants that can run on a standalone mobile device (Raspberry Pi, smartphone), without the need for an internet connection.	Use of a Deep learning based automated detection technique to detect leaf blight and rust.	The Deep CNN's performance study successfully revealed an accuracy rate of 98.40%.	Live images captured from smart phone has an average accuracy of 88.66%

CHAPTER THREE
RESEARCH METHODOLOGY

3.1 Introduction

This project work presents the building process of a DSS for detecting maize plant disease. This chapter reports the design phase of the model, it involves decomposing the whole system into smaller parts and defining the relationship among the constituent parts.

3.2 SDLC Model

The software process model of choice is the waterfall technique. A few fundamental activities, including the definition of the functional requirements, architectural design, detailed design, execution, component verification, integration and requirement validation, may be completed using the waterfall technique.

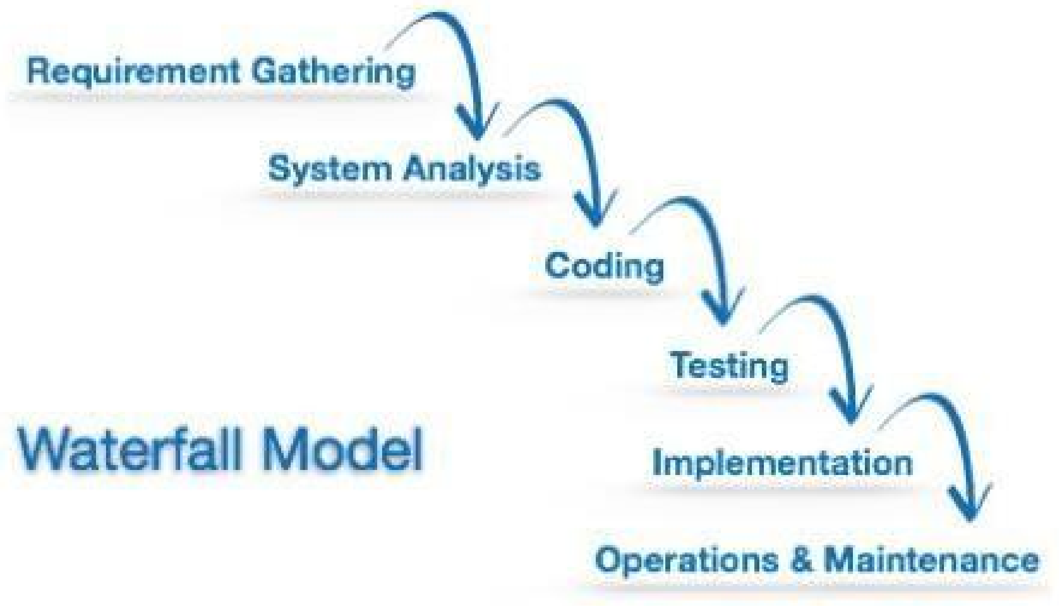


Figure 3-1: Waterfall Model

Source: (James Green 2018)

The model's sequential nature prevents us from undoing or redoing our activities. This technique works best when developers have previously planned and produced similar software and are familiar with all of its domains.

3.3 Description of Dataset

Images of 3912 from a collection of maize plant diseases were utilized to develop this system. Obtainable via Kaggle, which provides an open archive of pictures on plant disease, include maize blight, maize common rust, maize gray leaf spot, and healthy maize plants.

Table 3-1: Maize Plant Dataset

Category	**Number of images**
Maize Blight	985
Maize Common rust	1,192
Maize Gray leaf spot	573
Maize Healthy	1,162

In order to build the model, the dataset must be divided into various sections which are training, testing and validating to be able to achieve a good outcome

Table 3-2: Dataset Splitting

Category	**Number of images**	**Train**	**Test**	**Validation**
Maize Blight	985	80%	10%	10%
Maize Common rust	1,192	80%	10%	10%
Maize Gray leaf spot	573	80%	10%	10%
Maize Healthy	1,162	80%	10%	10%

3.4 Defining the model architecture

This phase of the deep modeling development process is crucial. The define of which architecture used which is CNN and the requirements. A CNN is a class neural network that particularly adept at processing information or data with a grid-like architecture, such as an image. An image is a presentation of visual data, it contains several series of pixel arranged in a grid like form that contains pixel values to detect how bright and what color each pixel should represent. They are steps needed to be taken to build a CNN layer namely:

i. Convolution layer
ii. Pooling layer
iii. Flattening layer
iv. Dense layer

❖ Convolution layer

The basic component of CNN, which carries the majority of the network's computational burden, is the convolution layer. The layer creates a dot product across two matrices, one of which is the kernel—a collection of learnable parameters—and the other of which is the constrained area of the flexible field. Compared to a picture, the kernel is smaller in space but deeper. This indicates that the kernel height and width will be spatially tiny if the picture is made up of three (RGB) channels, but the depth will go up to all three channels.

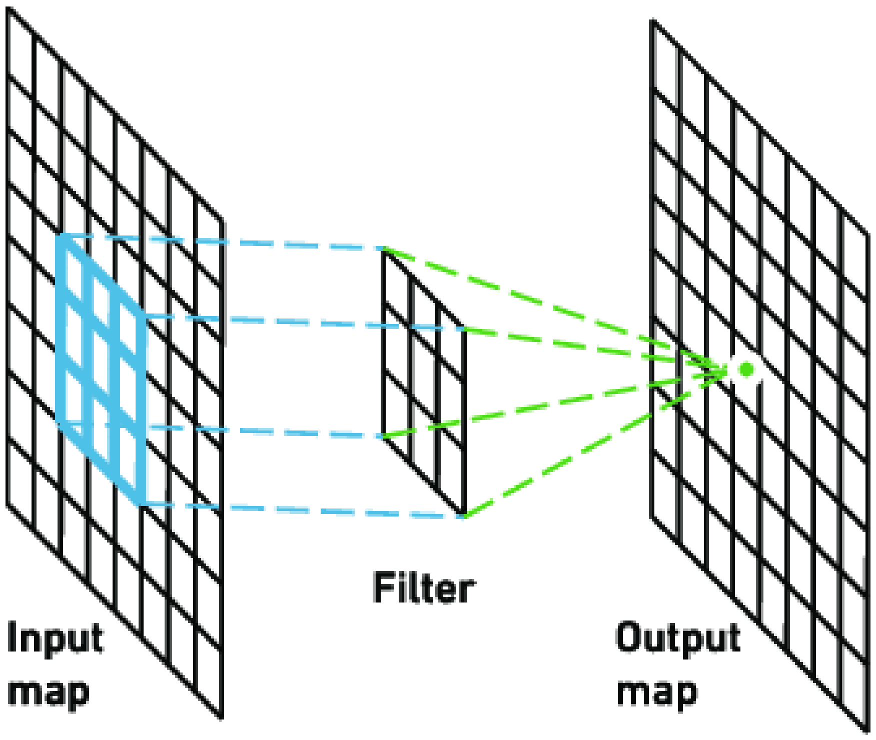


Figure 3-2: Convolutional layer

Source: (Kim Matana, 2016)

❖ Pooling Layer

By replacing a network output at a specified location with a summary statistic of neighboring outputs, the pooling layer reduces the spatial size of the representation, thus lessens overall amount of computing and weights needed. Each slice of the representation is subjected to the pooling procedure separately.

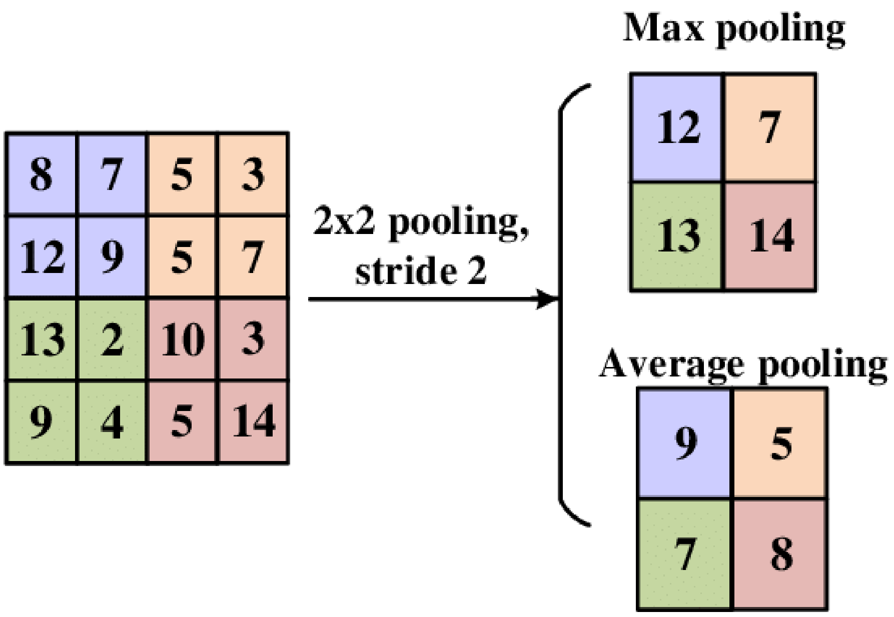


Figure 3-3: Pooling layer

Source: (Jay Avana, 2020)

❖ Flattening Layer

When data is created as a dimensional array to be entered into the next layer, it is flattened. We flatten the convolutional layer output to produce a solitary, lengthy feature vector. Additionally, it is linked to the last classification model, also known as a fully-connected layer. To put it in other words, we link the last layer to the single line containing all the pixel data. And once more.

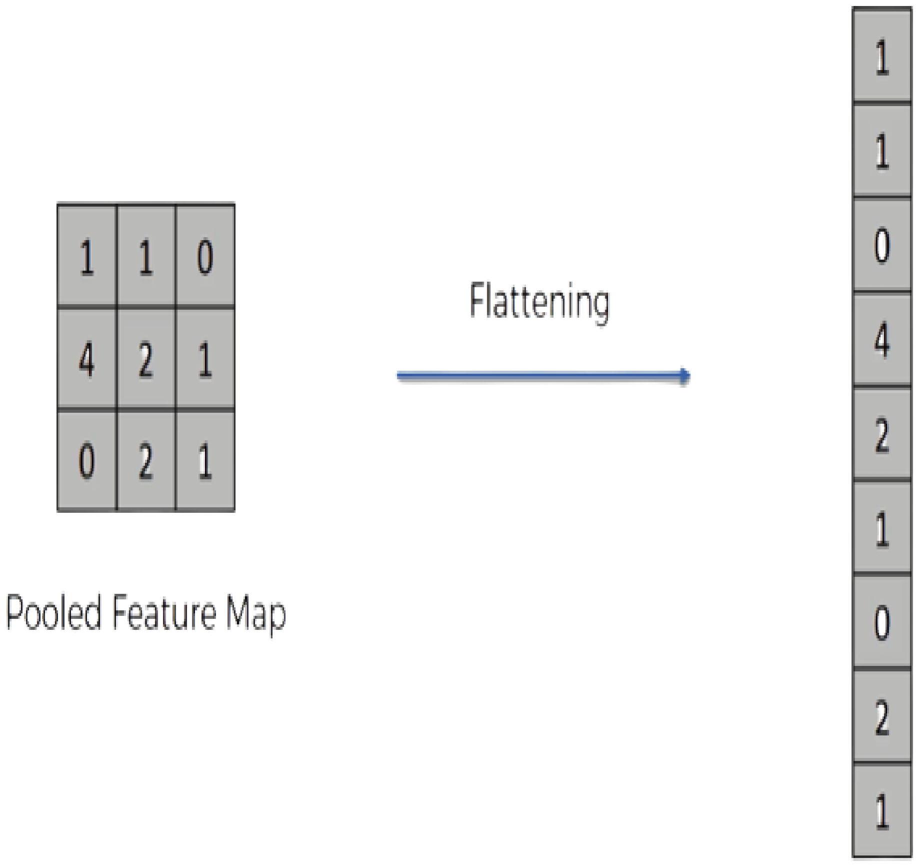


Figure 3-4: Flattening layer

Source: (Jai Khan, 2018)

❖ Dense Layer

A neural network in a dense layer is one in which the levels that came before it are closely related, meaning that the neurons in each layer are connected to the neurons in every other layer. This layer is the one that artificial neural network networks utilize the most.

The neurons in a model's dense layer multiply matrices and vectors and accept input from all the neurons in the layer above them. Make sure row vector of the output from the preceding layers is the same as the column vector of the dense layer when multiplying matrices with vectors.

.

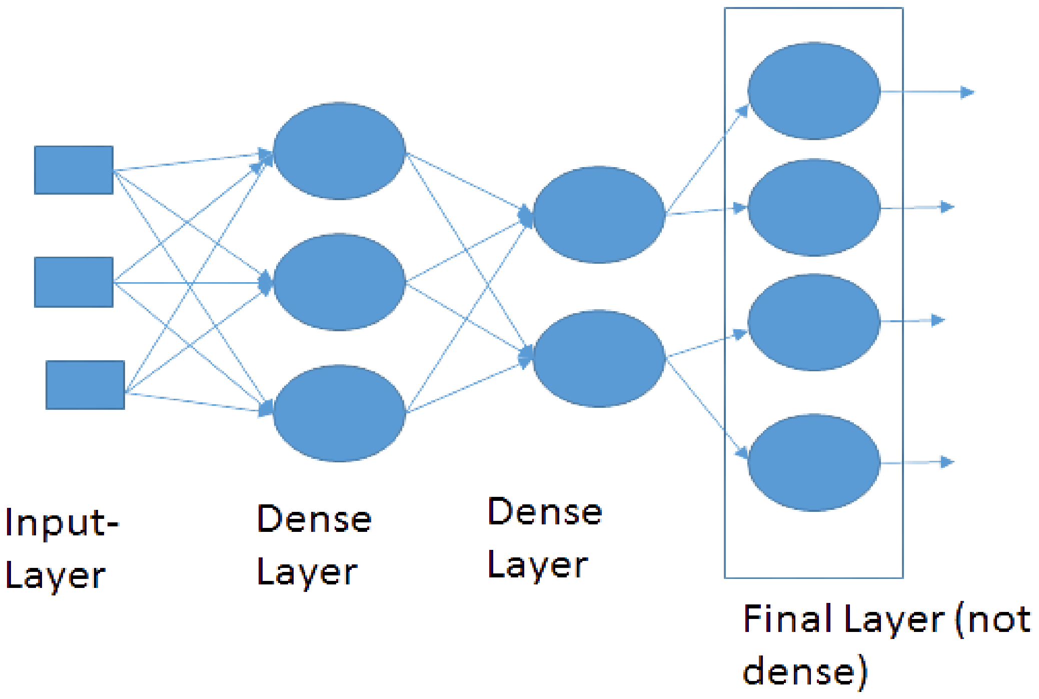


Figure 3-4: Dense layer

Source: (Shashwat Tiwari, 2019)

3.5 Data Flow Diagram

A data flow diagram uses minimum numbers of simple symbols to show the performance of a system and the data flow among the functions.

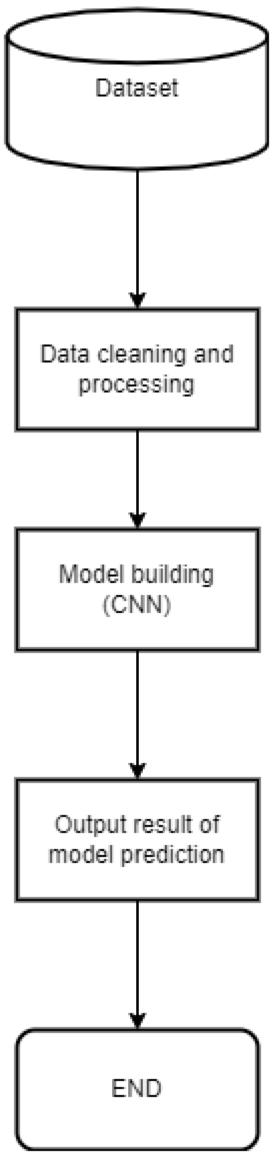

Dataset
Data cleaning and processing
Model building (CNN)
Output result of model prediction
END

Figure 3-5: Data flow diagram of CNN algorithm

Source: Researcher

3.6 Libraries Used

3.6.1 Matplot Library

A Python 2D charting toolkit called Matplotlib creates publication-quality visuals in a range of physical formats and in cross-platform interactive settings. Matplotlib aims to make difficult things feasible and simple things easy. With just a few blocks of code, you can create plots, bar graphs, power spectra, line graphs, error charts, scatterplots, and more. Examples may be displayed in the plot sample and thumbnail galleries.

Particularly when used in conjunction with IPython, the pyplot package offers a MATLAB-like GUI for basic charting. For the power user, there are two options for controlling line styles, font settings, axis properties, etc.: an object-oriented interface or a collection of MATLAB-friendly methods.

3.6.2 NumPy

The cornerstone Python module for scientific computing is called NumPy. NumPy is a sophisticated multi-dimensional data container that has several applications outside of science. Data-type definitions are flexible. As a result, NumPy can quickly and easily interact with a broad range of databases.

3.6.3 TensorFlow

TensorFlow is a free toolkit for numerical computation that is compatible with Python that speeds up and simplifies the creation of neural networks etc. With the help of TensorFlow, deep neural networks can be trained and used for a variety of tasks, including handwritten digit classification, image recognition, word embedding's, recurrent neural networks, sequence-to-sequence models for machine translation, natural language processing, and PDE (partial differential equation)-based simulations. The best part is that TensorFlow uses the same models that were used for training to provide output predicting at volume..

3.7 Evaluation Metrics

Simply put, accuracy reflects how often the classifier predicts correctly. The percentage of the number of accurate forecasts to all of the predictions may be used to determine accuracy..

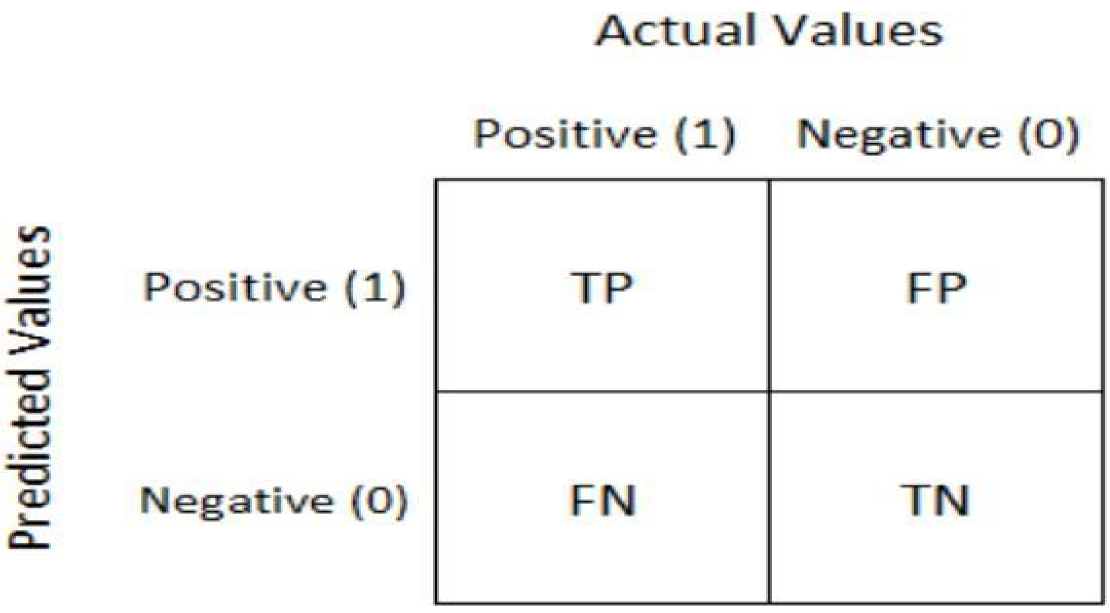
$$\textit{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

You would assume that a model is working well when it reports an accuracy rate of 99%, but this isn't always the case and in certain cases, it might be deceptive. I'll provide an instance to illustrate how this works.Con

Confusion Matrix

A performance indicator for classification involving machine learning issues when the output might be two or more classes is the confusion matrix. It is a table containing combinations of values that were expected and real.

The chart that is frequently used to represent how well a classification model performs on a set of data in which the real values are known is known as a confusion matrix.



		Actual Values	
		Positive (1)	Negative (0)
Predicted Values	Positive (1)	TP	FP
	Negative (0)	FN	TN

True Optimistic: Our positive prediction came true. We correctly identified the lady in the photograph as being pregnant.

True Negative: Our prediction of a negative outcome came true. We correctly identified the guy in the photograph as not being pregnant.

False Positive (Type 1 Error): We expected a positive result, but it wasn't true. Although we assumed a male in the photograph was pregnant, he is not.

False Negative (Type 2 Error): We expected a negative result, but it wasn't true. We assumed that the lady in the photograph was not pregnant, but she is.

Let's talk about some other confusion matrix measures now that we've covered accuracy.

1. **Precision** – Precision describes the percentage of precisely anticipated situations that really resulted in a good outcome. Precision is helpful when False Positives are more problematic than False Negatives. Precision is crucial in applications like music or video recommendation systems, e-commerce websites, etc. because inaccurate results might cause user attrition and hurt the company's bottom line.

The clarity of a label is based on the ratio of factual positives to expected positives..

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive}$$

2. **Recall (Sensitivity)** – Recall describes the proportion of real positive instances that our model was able to properly anticipate. When False Negative is more important than False Positive, it is a valuable statistic. It is crucial in medical situations because even if we raise a false alarm, the real good examples shouldn't go unnoticed.

The proportion of genuine positives to all other positive results is known as recall for a label.

$$Recall = \frac{TruePositive}{TruePositive + FalseNegative}$$

3. **F1 Score**—It gives a combined idea about Precision and Recall metrics. It is maximum when Precision is equal to Recall.

F1 Score is the harmonic mean of precision and recall.

$$F1 = 2.\,\frac{Precision\ \times Recall}{Precision + Recall}$$

More severe values are penalized by the F1 score. F1 Score may function as a useful assessment statistic in the following circumstances:

i. When the price of FP and FN are comparable.

ii. Increasing the amount of data doesn't significantly alter the result.

iii. High True Negative.

CHAPTER FOUR
IMPLEMENTATION, RESULT AND DISCUSSION

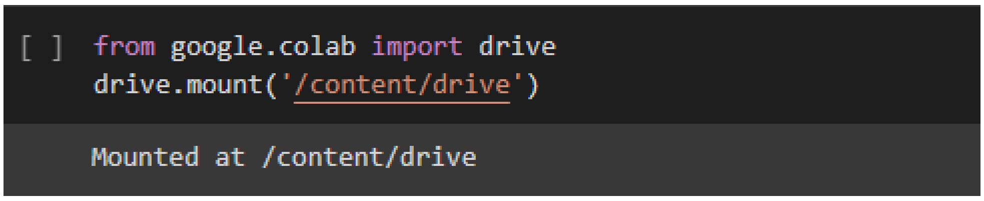
4.1 Introduction

This chapter how's the result of the trained model and its implementation. The result will be shown for each category of maize plant.

4.2 Implementation

4.2.1 Importing Dataset

The dataset was imported form google drive, were the dataset was uploaded.



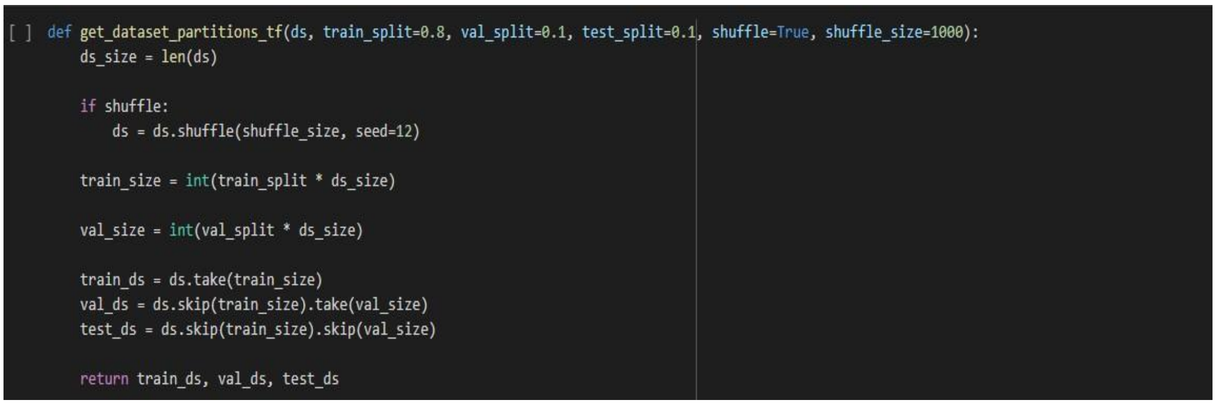
```
[ ]  from google.colab import drive
     drive.mount('/content/drive')

     Mounted at /content/drive
```

Figure 4-1: importing dataset

Source: Researcher

4.2.3 Splitting the Dataset

The data was divided into train, test and validation in which train takes 80 percent of the dataset while test and validation takes 10 percent each.



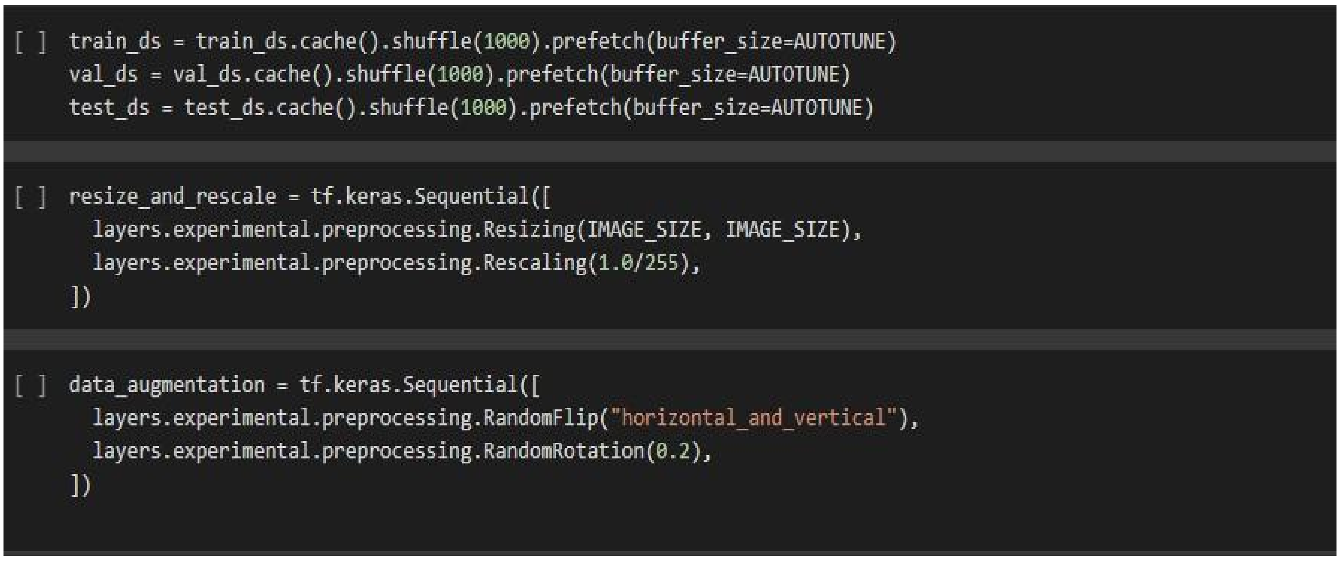
```
[ ]  def get_dataset_partitions_tf(ds, train_split=0.8, val_split=0.1, test_split=0.1, shuffle=True, shuffle_size=1000):
         ds_size = len(ds)

         if shuffle:
             ds = ds.shuffle(shuffle_size, seed=12)

         train_size = int(train_split * ds_size)

         val_size = int(val_split * ds_size)

         train_ds = ds.take(train_size)
         val_ds = ds.skip(train_size).take(val_size)
         test_ds = ds.skip(train_size).skip(val_size)

         return train_ds, val_ds, test_ds
```

Figure 4-2: Splitting the dataset

Source: Researcher

4.2.3 Data augmentation

The dataset was resized, rescaled and flipped randomly in a vertical and horitonzal rotation.



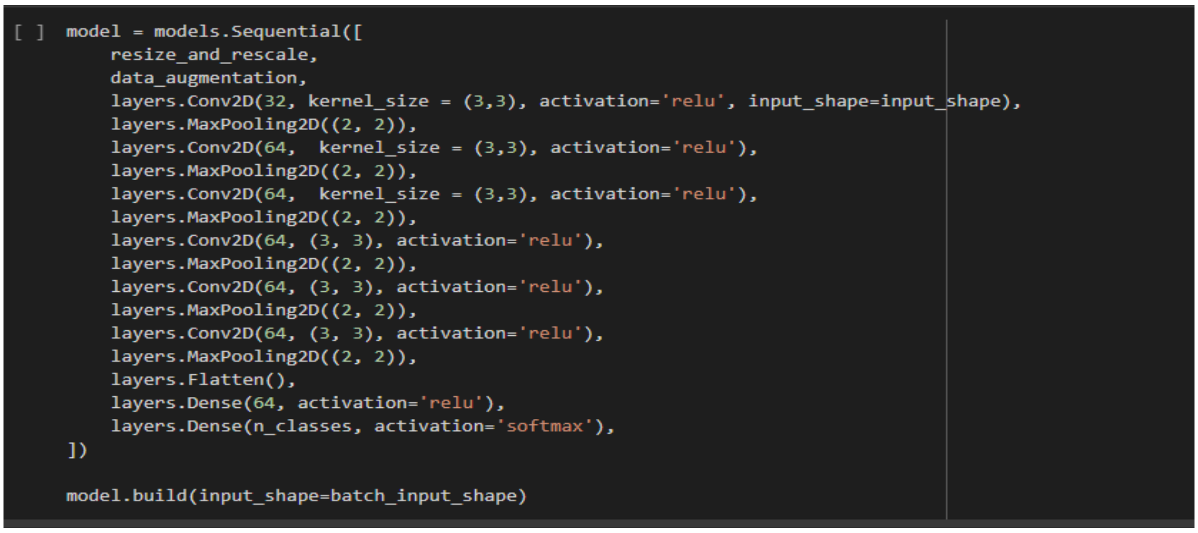
```
[ ] train_ds = train_ds.cache().shuffle(1000).prefetch(buffer_size=AUTOTUNE)
    val_ds = val_ds.cache().shuffle(1000).prefetch(buffer_size=AUTOTUNE)
    test_ds = test_ds.cache().shuffle(1000).prefetch(buffer_size=AUTOTUNE)

[ ] resize_and_rescale = tf.keras.Sequential([
      layers.experimental.preprocessing.Resizing(IMAGE_SIZE, IMAGE_SIZE),
      layers.experimental.preprocessing.Rescaling(1.0/255),
    ])

[ ] data_augmentation = tf.keras.Sequential([
      layers.experimental.preprocessing.RandomFlip("horizontal_and_vertical"),
      layers.experimental.preprocessing.RandomRotation(0.2),
    ])
```

Figure 4-2: Data Augmentation

Source: Researcher

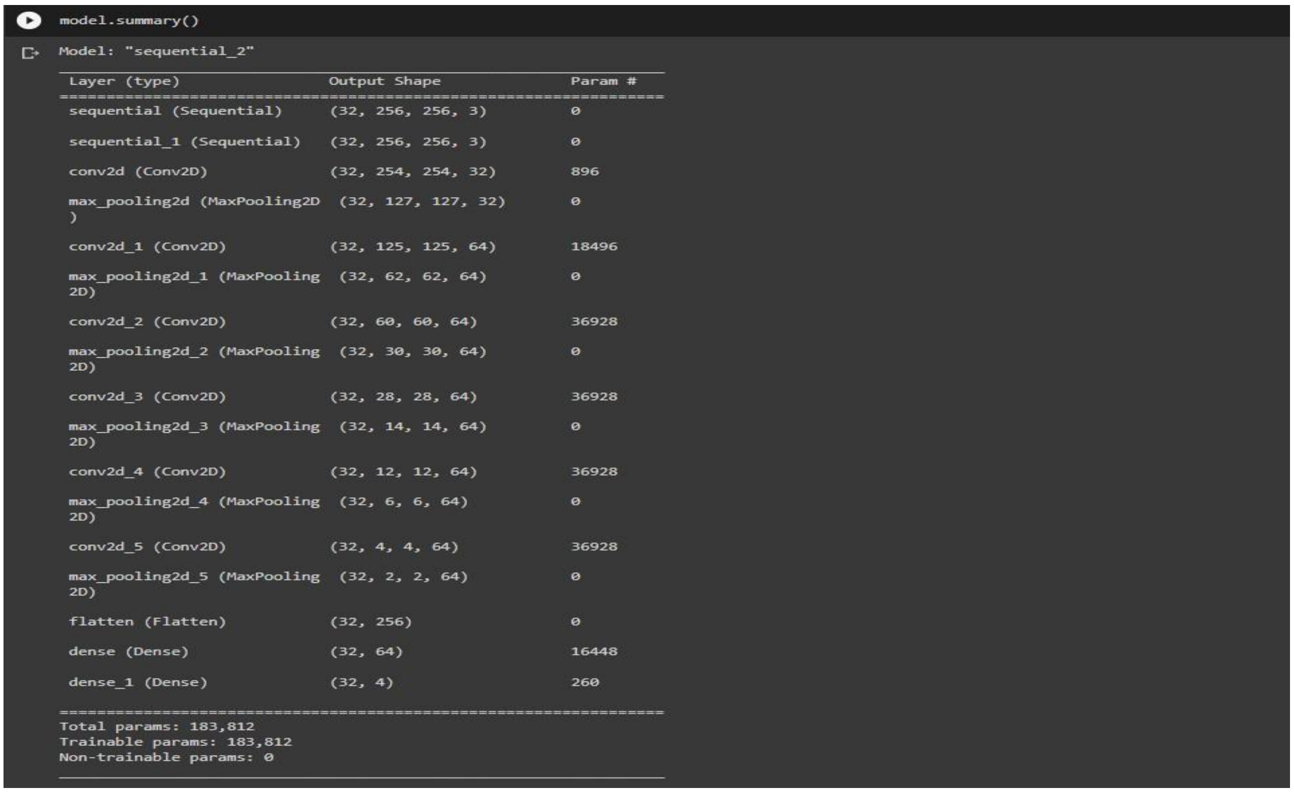
4.2.4 Defining Model

The model architecture used was cnn and the summary of the model output was displayed.



```
[ ] model = models.Sequential([
        resize_and_rescale,
        data_augmentation,
        layers.Conv2D(32, kernel_size = (3,3), activation='relu', input_shape=input_shape),
        layers.MaxPooling2D((2, 2)),
        layers.Conv2D(64,  kernel_size = (3,3), activation='relu'),
        layers.MaxPooling2D((2, 2)),
        layers.Conv2D(64,  kernel_size = (3,3), activation='relu'),
        layers.MaxPooling2D((2, 2)),
        layers.Conv2D(64, (3, 3), activation='relu'),
        layers.MaxPooling2D((2, 2)),
        layers.Conv2D(64, (3, 3), activation='relu'),
        layers.MaxPooling2D((2, 2)),
        layers.Conv2D(64, (3, 3), activation='relu'),
        layers.MaxPooling2D((2, 2)),
        layers.Flatten(),
        layers.Dense(64, activation='relu'),
        layers.Dense(n_classes, activation='softmax'),
    ])

    model.build(input_shape=batch_input_shape)
```

Figure 4-3: Defining the model
Source: Researcher



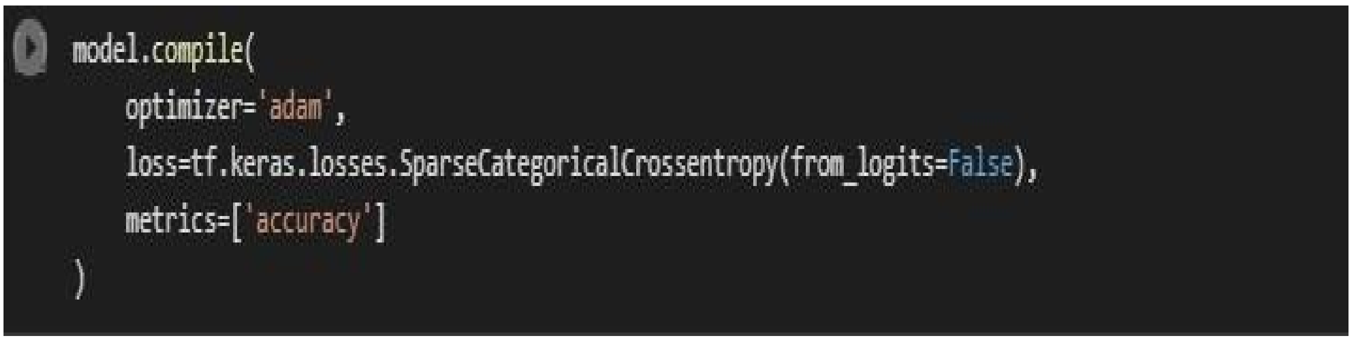
```
model.summary()

Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 sequential (Sequential)     (32, 256, 256, 3)         0

 sequential_1 (Sequential)   (32, 256, 256, 3)         0

 conv2d (Conv2D)             (32, 254, 254, 32)        896

 max_pooling2d (MaxPooling2D  (32, 127, 127, 32)       0
 )

 conv2d_1 (Conv2D)           (32, 125, 125, 64)        18496

 max_pooling2d_1 (MaxPooling  (32, 62, 62, 64)         0
 2D)

 conv2d_2 (Conv2D)           (32, 60, 60, 64)          36928

 max_pooling2d_2 (MaxPooling  (32, 30, 30, 64)         0
 2D)

 conv2d_3 (Conv2D)           (32, 28, 28, 64)          36928

 max_pooling2d_3 (MaxPooling  (32, 14, 14, 64)         0
 2D)

 conv2d_4 (Conv2D)           (32, 12, 12, 64)          36928

 max_pooling2d_4 (MaxPooling  (32, 6, 6, 64)           0
 2D)

 conv2d_5 (Conv2D)           (32, 4, 4, 64)            36928

 max_pooling2d_5 (MaxPooling  (32, 2, 2, 64)           0
 2D)

 flatten (Flatten)           (32, 256)                 0

 dense (Dense)               (32, 64)                  16448

 dense_1 (Dense)             (32, 4)                   260

=================================================================
Total params: 183,812
Trainable params: 183,812
Non-trainable params: 0
_________________________________________________________________
```

Figure 4-4: Model summary

Source: Researcher

4.2.5 Model Training and Compiling

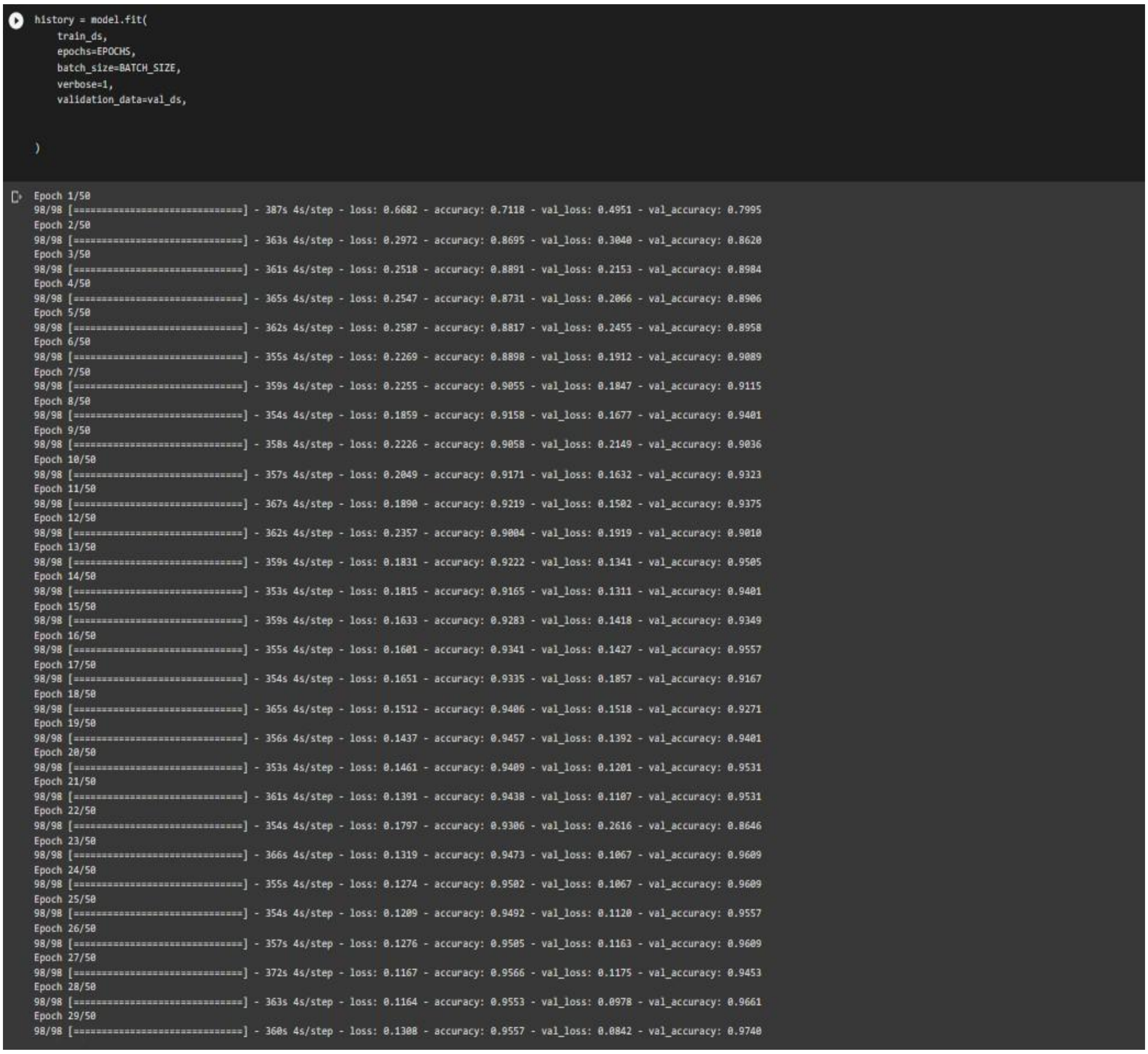
The model was compiled with adam optimizer and the metrics used was accuracy. Inorder to train the model 50 epochs was trained to achieve a satisfactory result.



```
model.compile(
    optimizer='adam',
    loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=False),
    metrics=['accuracy']
)
```

Figure 4-5: Model compile

Source: Researcher



```
history = model.fit(
    train_ds,
    epochs=EPOCHS,
    batch_size=BATCH_SIZE,
    verbose=1,
    validation_data=val_ds,


)
```

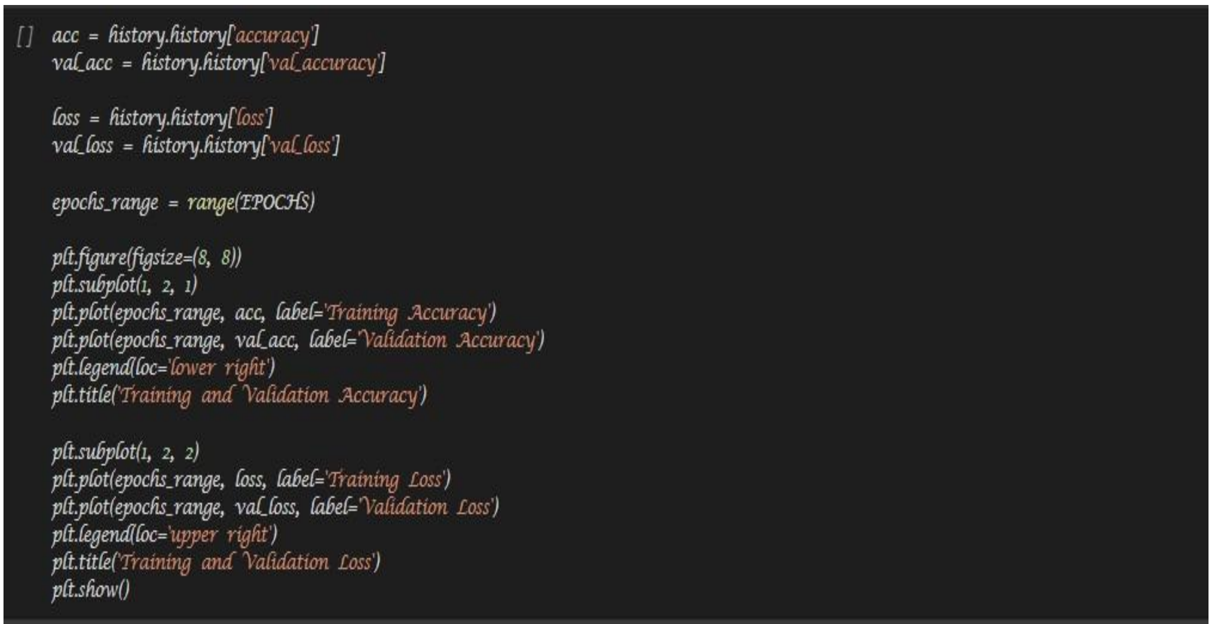
```
Epoch 1/50
98/98 [==============================] - 387s 4s/step - loss: 0.6682 - accuracy: 0.7118 - val_loss: 0.4951 - val_accuracy: 0.7995
Epoch 2/50
98/98 [==============================] - 363s 4s/step - loss: 0.2972 - accuracy: 0.8695 - val_loss: 0.3040 - val_accuracy: 0.8620
Epoch 3/50
98/98 [==============================] - 361s 4s/step - loss: 0.2518 - accuracy: 0.8891 - val_loss: 0.2153 - val_accuracy: 0.8984
Epoch 4/50
98/98 [==============================] - 365s 4s/step - loss: 0.2547 - accuracy: 0.8731 - val_loss: 0.2066 - val_accuracy: 0.8906
Epoch 5/50
98/98 [==============================] - 362s 4s/step - loss: 0.2587 - accuracy: 0.8817 - val_loss: 0.2455 - val_accuracy: 0.8958
Epoch 6/50
98/98 [==============================] - 355s 4s/step - loss: 0.2269 - accuracy: 0.8898 - val_loss: 0.1912 - val_accuracy: 0.9089
Epoch 7/50
98/98 [==============================] - 359s 4s/step - loss: 0.2255 - accuracy: 0.9055 - val_loss: 0.1847 - val_accuracy: 0.9115
Epoch 8/50
98/98 [==============================] - 354s 4s/step - loss: 0.1859 - accuracy: 0.9158 - val_loss: 0.1677 - val_accuracy: 0.9401
Epoch 9/50
98/98 [==============================] - 358s 4s/step - loss: 0.2226 - accuracy: 0.9058 - val_loss: 0.2149 - val_accuracy: 0.9036
Epoch 10/50
98/98 [==============================] - 357s 4s/step - loss: 0.2049 - accuracy: 0.9171 - val_loss: 0.1632 - val_accuracy: 0.9323
Epoch 11/50
98/98 [==============================] - 367s 4s/step - loss: 0.1890 - accuracy: 0.9219 - val_loss: 0.1502 - val_accuracy: 0.9375
Epoch 12/50
98/98 [==============================] - 362s 4s/step - loss: 0.2357 - accuracy: 0.9004 - val_loss: 0.1919 - val_accuracy: 0.9010
Epoch 13/50
98/98 [==============================] - 359s 4s/step - loss: 0.1831 - accuracy: 0.9222 - val_loss: 0.1341 - val_accuracy: 0.9505
Epoch 14/50
98/98 [==============================] - 353s 4s/step - loss: 0.1815 - accuracy: 0.9165 - val_loss: 0.1311 - val_accuracy: 0.9401
Epoch 15/50
98/98 [==============================] - 359s 4s/step - loss: 0.1633 - accuracy: 0.9283 - val_loss: 0.1418 - val_accuracy: 0.9349
Epoch 16/50
98/98 [==============================] - 355s 4s/step - loss: 0.1601 - accuracy: 0.9341 - val_loss: 0.1427 - val_accuracy: 0.9557
Epoch 17/50
98/98 [==============================] - 354s 4s/step - loss: 0.1651 - accuracy: 0.9335 - val_loss: 0.1857 - val_accuracy: 0.9167
Epoch 18/50
98/98 [==============================] - 365s 4s/step - loss: 0.1512 - accuracy: 0.9406 - val_loss: 0.1518 - val_accuracy: 0.9271
Epoch 19/50
98/98 [==============================] - 356s 4s/step - loss: 0.1437 - accuracy: 0.9457 - val_loss: 0.1392 - val_accuracy: 0.9401
Epoch 20/50
98/98 [==============================] - 353s 4s/step - loss: 0.1461 - accuracy: 0.9409 - val_loss: 0.1201 - val_accuracy: 0.9531
Epoch 21/50
98/98 [==============================] - 361s 4s/step - loss: 0.1391 - accuracy: 0.9438 - val_loss: 0.1107 - val_accuracy: 0.9531
Epoch 22/50
98/98 [==============================] - 354s 4s/step - loss: 0.1797 - accuracy: 0.9306 - val_loss: 0.2616 - val_accuracy: 0.8646
Epoch 23/50
98/98 [==============================] - 366s 4s/step - loss: 0.1319 - accuracy: 0.9473 - val_loss: 0.1067 - val_accuracy: 0.9609
Epoch 24/50
98/98 [==============================] - 355s 4s/step - loss: 0.1274 - accuracy: 0.9502 - val_loss: 0.1067 - val_accuracy: 0.9609
Epoch 25/50
98/98 [==============================] - 354s 4s/step - loss: 0.1209 - accuracy: 0.9492 - val_loss: 0.1120 - val_accuracy: 0.9557
Epoch 26/50
98/98 [==============================] - 357s 4s/step - loss: 0.1276 - accuracy: 0.9505 - val_loss: 0.1163 - val_accuracy: 0.9609
Epoch 27/50
98/98 [==============================] - 372s 4s/step - loss: 0.1167 - accuracy: 0.9566 - val_loss: 0.1175 - val_accuracy: 0.9453
Epoch 28/50
98/98 [==============================] - 363s 4s/step - loss: 0.1164 - accuracy: 0.9553 - val_loss: 0.0978 - val_accuracy: 0.9661
Epoch 29/50
98/98 [==============================] - 360s 4s/step - loss: 0.1308 - accuracy: 0.9557 - val_loss: 0.0842 - val_accuracy: 0.9740
```

Figure 4-6: Epochs

Source: Researcher

4.2.6 Traning and Validation

Below shows the graphical representation of the training loss, validation loss and training accuracy, validation accuracy of the model.



```
[ ] acc = history.history['accuracy']
    val_acc = history.history['val_accuracy']

    loss = history.history['loss']
    val_loss = history.history['val_loss']

    epochs_range = range(EPOCHS)

    plt.figure(figsize=(8, 8))
    plt.subplot(1, 2, 1)
    plt.plot(epochs_range, acc, label='Training Accuracy')
    plt.plot(epochs_range, val_acc, label='Validation Accuracy')
    plt.legend(loc='lower right')
    plt.title('Training and Validation Accuracy')

    plt.subplot(1, 2, 2)
    plt.plot(epochs_range, loss, label='Training Loss')
    plt.plot(epochs_range, val_loss, label='Validation Loss')
    plt.legend(loc='upper right')
    plt.title('Training and Validation Loss')
    plt.show()
```

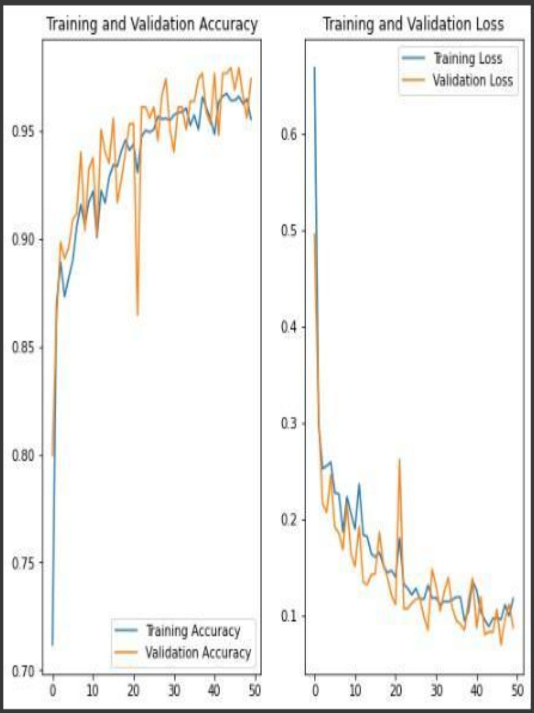
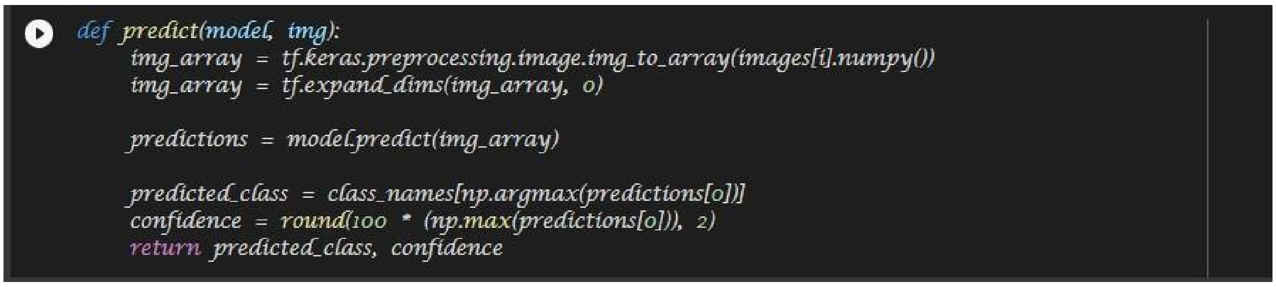


Figure 4-9: Graph of training and validation

Source: Researcher

4.2.7 Prediction

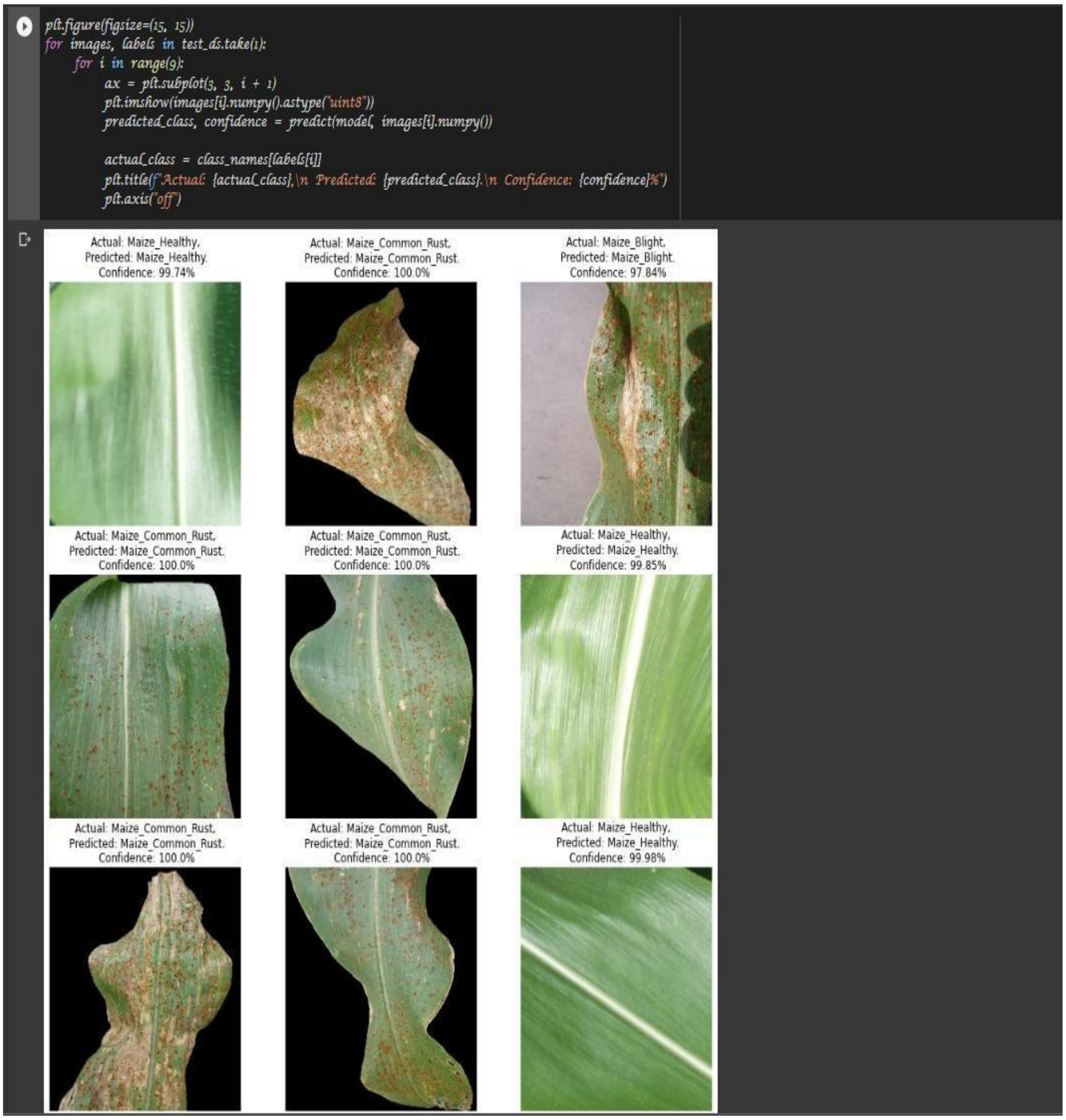
After the evaluation of the model, the model was used to prediction some of maize plant disease.



```
def predict(model, img):
    img_array = tf.keras.preprocessing.image.img_to_array(images[i].numpy())
    img_array = tf.expand_dims(img_array, 0)

    predictions = model.predict(img_array)

    predicted_class = class_names[np.argmax(predictions[0])]
    confidence = round(100 * (np.max(predictions[0])), 2)
    return predicted_class, confidence
```

Figure 4-10: Function for Predicting

Source: Researcher



```
plt.figure(figsize=(15, 15))
for images, labels in test_ds.take(1):
    for i in range(9):
        ax = plt.subplot(3, 3, i + 1)
        plt.imshow(images[i].numpy().astype("uint8"))
        predicted_class, confidence = predict(model, images[i].numpy())

        actual_class = class_names[labels[i]]
        plt.title(f"Actual: {actual_class},\n Predicted: {predicted_class}.\n Confidence: {confidence}%")
        plt.axis("off")
```

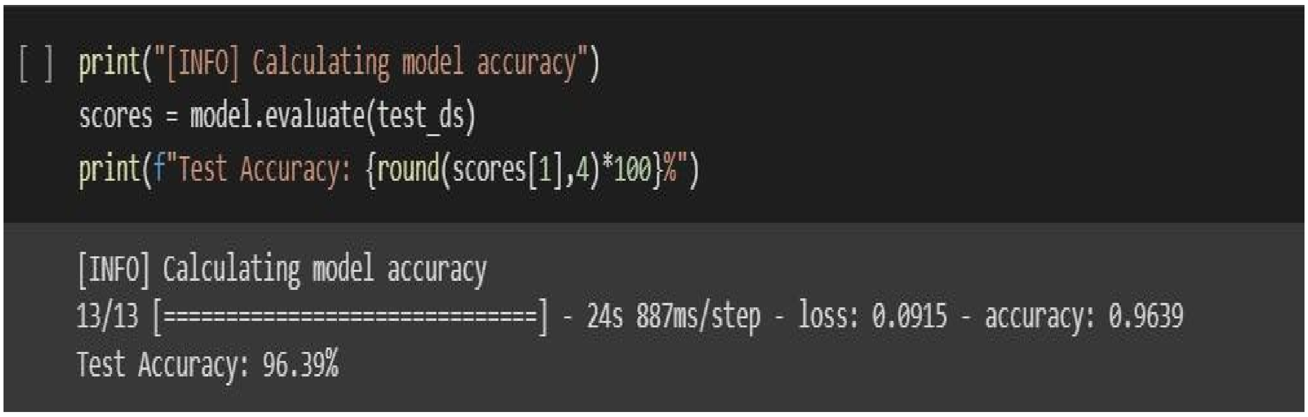
```
Actual: Maize_Healthy,
Predicted: Maize_Healthy.
Confidence: 99.74%

Actual: Maize_Common_Rust,
Predicted: Maize_Common_Rust.
Confidence: 100.0%

Actual: Maize_Blight,
Predicted: Maize_Blight.
Confidence: 97.84%

Actual: Maize_Common_Rust,
Predicted: Maize_Common_Rust.
Confidence: 100.0%

Actual: Maize_Common_Rust,
Predicted: Maize_Common_Rust.
Confidence: 100.0%

Actual: Maize_Healthy,
Predicted: Maize_Healthy.
Confidence: 99.85%

Actual: Maize_Common_Rust,
Predicted: Maize_Common_Rust.
Confidence: 100.0%

Actual: Maize_Common_Rust,
Predicted: Maize_Common_Rust.
Confidence: 100.0%

Actual: Maize_Healthy,
Predicted: Maize_Healthy.
Confidence: 99.98%
```

Figure 4-11: Prediction

Source: Researcher

4.2.8 Model Evaluation

After the model was trained an evaluation was carried out, checking loss and accuracy of the model.



```
[ ]  print("[INFO] Calculating model accuracy")
     scores = model.evaluate(test_ds)
     print(f"Test Accuracy: {round(scores[1],4)*100}%")

     [INFO] Calculating model accuracy
     13/13 [==============================] - 24s 887ms/step - loss: 0.0915 - accuracy: 0.9639
     Test Accuracy: 96.39%
```

Figure 4-7: Model Evaluation

Source: Researcher

Table 4-1: Evaluation Metrics

Category	**Number of images**	**Accuracy**	**Precision**	**F1 score**	**Recall**
Maize Blight	985	0.967	0.955	0.965	0.984
Maize Common rust	1,192	0.924	0.923	0.945	0.934
Maize Gray leaf spot	573	0.962	0.956	0.932	0.941
Maize Healthy	1,162	0.956	0.945	0.922	0.963

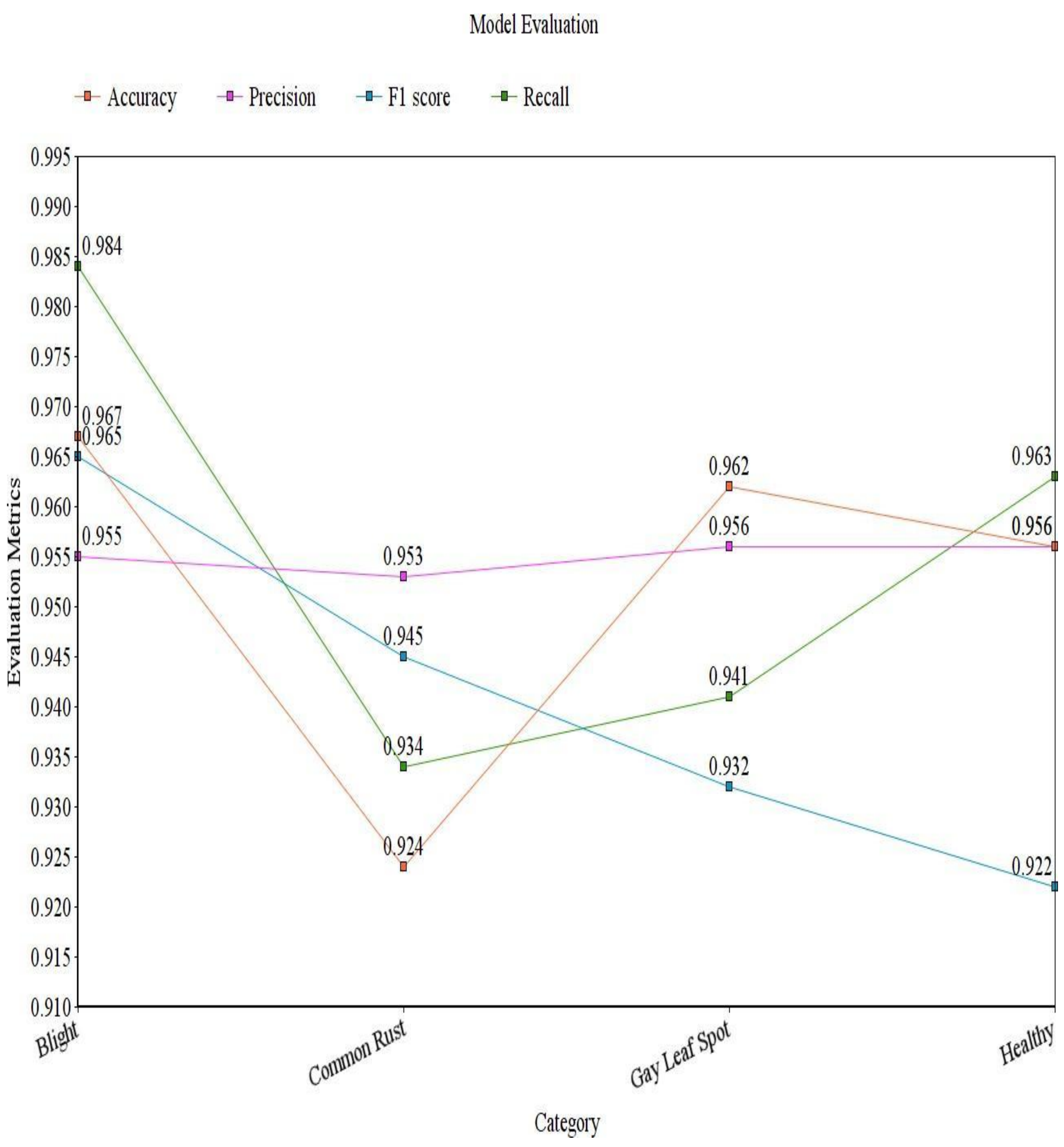


Figure 4-8: Graph of Evaluation metrics

Source: Researcher

4.2.8.1 Discussion

After building the model successfully, evaluation metrics were implemented on the model to check each category of the dataset. The accuracy, precision, f1 score and recall output are shown above with this I was able to properly quantify the performance of the model and understand how well the model performed on the input data that was supplied to it.

CHAPTER FIVE

SUMMARY, CONCLUSION AND RECOMMENDATION

5.1 Summary

As it was proposed in chapter one, the principle focus of the study was to develop a machine learning based decision support system to detect disease in maize plant with the purpose of designing the model and implementing the model. The model was built on google colab using convolutional neural network. The study identifies means via which maize plant disease can be easily detected without the need of specialist. In carrying out the study, most of the information was gathered from journals, books and past research papers related to the subject

5.2 Conclusions

In this project, a system which can be implemented in the field with simplicity and is capable of identifying early indicators of illness from maize leaf photos, thereby providing pre-emptive remedial activities prior to major yield loss. Our solution employs deep learning employing CNN to identify specific disease in maize. However, as the gathered image encloses compressed information that is highly demanding for the system to interpret, it requires a pre-processing phase to extract a certain feature (e.g., colour and shape) that is manually determined by specialists. In such circumstances, deep learning is typically applied since it allows the computer to independently discover the best suited feature without human interaction. Among different network architectures used in deep learning, convolutional neural networks (CNN) are commonly utilized in image identification, detection and classification.

5.3 Recommendations

This project suggests that extra initiatives be put into the identification of maize plant disease. Additionally, information regarding datasets for other maize plant disease should be made available and be collected for machine learning algorithms which may be used to enhance various models.

.

References

Chunxia, Z., Xiuqing , W., & Xudong, L. (2010). Design of Monitoring and Control Plant Disease System Based on DSP&FPGA. *Second International Conference on Networks Security, Wireless Communications and Trusted Computing.*

F, B., & C.W, H. (2008). Handbook on Decision Support Systems. Berlin: Springer Verlag.

Fakorede, M. (2001). Revolutionizing Nigerian Agriculture with Golden seed. *Inaugural lecture series Obafemi Awolowo University Press Limited Ile-Ife*.

FAO. (2018). *Maize production statistics in the Crop Production Year Book.* (Food and Agriculture Organization of the United Nations).

Fatima, B., & Adbul , S. (2005). Fungi associate with maize.

H, A.-H., S, B.-A., M, R., M, B., & Z, A. (March 2011). "Fast and Accurate Detection and Classification f Plant Disease39s. *International Journal of Computer Applicaion*, (0975 – 8887)Volume 17– No.1.

Hackathorn, R., & P.G, G. (1981). Organizational Strategies for Personal Computing in Decision Support Systems. *MIS Quarterly*.

Henk G, S. (1987). Expert systems and artificial intelligence in decision support systems:. *proceedings of the Second Mini Euroconference.* Nehelands.

Holsapple, C., & A.B, W. (1996). Decision Support Systems: A Knowledge-Based Approach. St. Paul: West Publishing.

IITA. (2005). *International Institute of Tropical Agriculure Annual report.*

Iken, J., & Amusa, N. (2004). Review: Maize research and production in Nigeria. 302-307.

Keen, P. (1978). Decision support systems: an organizational perspective. Reading, Mass., Addison-Wesley.

Monica, J., Ashwani, K., & Rushikesh, B. (2013). Image Processing For Smart Farming: Detection Of Disease And Fruit Grading. *IEEE Second International Conference on Image Information Processing.*

Mrunalini , R., & Prashant , R. (2012). Infected leaf Analysis and Comparison by Otsu Threshold and k-Means Clustering. *International Journal of Advanced Research in Computer Science and Software Engineering*.

Papadopoulos, A., Shipp, J., Jarvis, W., Jewett, T., & Clarke, N. 1995. The Harrow Expert System for Greenhouse Vegetables. *HortScience. American Society for Horticultural Science*.

Romney, D., Thorne, P., Lukuyu, B., & Thornton, P. (2003). Maize as food and feed in intensive smallholder . *management options for improved integration in mixed farming systems of East and Southern Africa*.

Sprague, R., & E.D, C. (1982). Building effective decision support systems. *Englewood Cliffs, N.J., Prentice-Hall.*

Wikimedia Foundation, I. (2011). *Wikepidia*. Retrieved from https://en.wikipedia.org/wiki/Decision_support_system

Wright, A., & Sittig, D. (2008). A framework and model for evaluating clinical decision support architectures . *Journal of Biomedical Informatics*.

Zhang, S., & Babovic, V. (2011). An evolutionary real options framework for the design and management of projects and systems with complex real options and exercising conditions..

Zulkifi, B. H., Abdul, H. A., Ali, Y., Shakaff, R. B., & S, M. F. (2012). Feasibility Study on Plant Chili Disease Detection Using Image Processing Techinques. *Third International Conference on Intelligent Systems Modelling and Simulation.* .