University of 20 août 1955 Skikda
Faculty of science
Department of Computer Science

A project to obtain a master's degree and a certificate for a startup project within the framework of the ministerial decision 1275

Smart Soil Analyzer and Crop Guidance System

Speciality:
Artificial Intelligence

Realized by: Marwa Boukhezna

Supervised by Dr.	Samira Hazmoune	Department of Computer Science
Co-supervised by Dr.	Bala Sahima	Department of Agronomic Sciences

Academic Year: 2024/2025

Acknowledgements

All praise is due to Allah, the Most Gracious and the Most Merciful, who granted me strength, patience, and perseverance to complete this work and guided me throughout my academic journey.

I would like to express my deepest gratitude to my two supervisors: Dr. Samira Hazmoune and Dr. Sahima Bala, for their insightful guidance, continuous support, and valuable advice throughout the course of this research. Their encouragement and dedication have been instrumental in shaping the direction and quality of this work.

With heartfelt emotion, I dedicate a special prayer to the souls of my late parents and brothers, may Allah have mercy on them and grant them the highest place in Paradise. Their love, prayers, and belief in me remain a source of strength that continues to inspire me even in their absence.

I also extend my sincere thanks to my extended family my uncles, aunts, cousins, and all relatives whose moral support and kind encouragement have meant a great deal to me throughout my journey.

Finally, I would like to thank all the professors who contributed to my education, and all my colleagues and friends who supported me and stood by me during this academic endeavor.

With gratitude and appreciation to all.

Dedication

I dedicate the humble fruit of my efforts to those who were the reason behind my success:

To the souls of my dear parents may Allah have mercy on them and grant them the highest place in Paradise. Their love, prayers, and values continue to guide and strengthen me even in their absence.

To my sister and my aunts, each for her support, and to their husbands especially my uncle Kaddour, who has been like a father to me throughout all these years.

To my entire extended family uncles, aunts, cousins, and all relatives thank you from the bottom of my heart for your prayers, encouragement, and constant presence in my life. You have truly been a pillar of support and pride.

To my family as a whole, for the care, motivation, and sense of belonging you gave me I am forever grateful.

To all my teachers throughout my educational journey, from elementary school to university thank you for every letter taught and every value passed on.

To my colleagues who shared this academic path with me your companionship was both meaningful and inspiring.

To everyone who helped me with a word or a deed and stood by me in completing this work may you be rewarded with all goodness.

To all those who strive for knowledge and believe in its noble mission.

And to everyone who holds love, respect, and appreciation for me this is for you.

Abstract

This study aims to develop an intelligent crop recommendation system based on soil characteristics using artificial intelligence techniques. The core objective is to classify the most suitable crop for cultivation in a specific soil based on various features such as nitrogen (N), phosphorus (P), potassium (K) levels, pH value, moisture, rainfall, soil type, and other environmental factors.

To achieve this, we applied a set of machine learning and deep learning algorithms, including Naïve Bayes (NB), Decision Tree (DT), Support Vector Machine (SVM), Logistic Regression (LR), as well as LSTM and GRU neural networks. The models were trained using a specialized agricultural dataset collected from Kaggle, with consistent preprocessing and splitting methods to ensure fair performance comparison.

A thorough hyperparameter tuning process was carried out to identify the optimal settings for each algorithm. The experimental results showed that deep learning models (LSTM and GRU) achieved strong classification performance, while the Decision Tree model provided good accuracy with lower computational requirements, making it a suitable option for resource constrained applications.

Keywords: Crop Recommendation, Artificial Intelligence, Soil Classification, Naïve Bayes, Decision Tree, SVM, Logistic Regression, LSTM, GRU, Machine Learning, Deep Learning.

ملخص

يهدف هذا البحث إلى تطوير نظام ذكي لتوصية المحاصيل الزراعية اعتماداً على خصائص التربة، باستخدام تقنيات الذكاء الاصطناعي. يتمثل الهدف الرئيسي في تصنيف أنسب محصول للزراعة في تربة معينة، بناءً على مجموعة من الخصائص مثل مستويات النيتروجين والفوسفور والبوتاسيوم، وقيمة الرقم الهيدروجيني، والرطوبة، وكميات الأمطار، ونوع التربة، بالإضافة إلى عوامل بيئية أخرى.

لتحقيق هذا الهدف، تم تطبيق مجموعة من خوارزميات التعلم الآلي والتعلم العميق، بما في ذلك خوارزمية نايف بايز، وشجرة القرار، وآلة الدعم الناقل، والانحدار اللوجستي، إلى جانب الشبكات العصبية العميقة مثل MTSL وURG. وقد تم تدريب هذه النماذج باستخدام مجموعة بيانات زراعية متخصصة مأخوذة من منصة elggaK، مع اتباع خطوات معالجة أولية وتقسيم موحدة لضمان مقارنة عادلة في الأداء.

كما تم إجراء عملية ضبط دقيق لفرط المعلمات بهدف تحديد الإعدادات المثلى لكل خوارزمية. وقد أظهرت النتائج التجريبية أن نماذج التعلم العميق قد حققت أداءً قوياً في التصنيف، بينما أظهرت شجرة القرار دقة جيدة مع متطلبات حسابية منخفضة، مما يجعلها خياراً مناسباً للتطبيقات التي تعاني من محدودية الموارد.

الكلمات المفتاحية: توصية المحاصيل، الذكاء الاصطناعي، تصنيف التربة، نايف بايز، شجرة القرار، آلة الدعم الناقل، الانحدار اللوجستي، التعلم الآلي، التعلم العميق.

Contents

List of Figures

List of Tables

List of Acronyms

Acc	Accuracy
Adam	Adaptive Moment Estimation
AI	Artificial Intelligence
ANN	Artificial Neural Network
AUC	Area Under Curve
BCE	Binary Cross-Entropy
CCE	Categorical Cross-Entropy
CNN	Convolutional Neural Network
CPU	Central Processing Unit
DL	Deep Learning
DT	Decision Tree
EC	Electrical Conductivity
FNN	Feedforward Neural Network
GIS	Geographic Information System
GPU	Graphics Processing Unit
GRU	Gated Recurrent Unit
IoT	Internet of Things
JS	JavaScript
K	Potassium
KNN	K-Nearest Neighbors
L1	L1 Regularization
L2	L2 Regularization
LR	Logistic Regression

LSTM	Long Short-Term Memory
ML	Machine Learning
MLP	Multi-Layer Perceptron
MSE	Mean Squared Error
N	Nitrogen
NB	Naïve Bayes
NLP	Natural Language Processing
P	Phosphorus
PCA	Principal Component Analysis
pH	Potential of Hydrogen (Soil Acidity)
PReLU	Parametric ReLU
RAM	Random Access Memory
RBF	Radial Basis Function
ReLU	Rectified Linear Unit
RL	Reinforcement Learning
RMSprop	Root Mean Square Propagation
RNN	Recurrent Neural Network
SF24	Smart Farming 2024 Dataset
SGD	Stochastic Gradient Descent
SLP	Single-Layer Perceptron
SMO	Sequential Minimal Optimization
SVM	Support Vector Machine
Test Acc	Test Accuracy
Val Acc	Validation Accuracy
VS Code	Visual Studio Code

General Introduction

Farmers, agricultural workers, and investors in the agricultural sector face numerous challenges, including labor intensive tasks and resource consumption. Over time, this has led to the adoption of many modern technologies in the sector. However, modern agriculture still faces multiple challenges that affect its efficiency and sustainability. These include difficulties in predicting agricultural output due to climate change, poor water resource management, and challenges in the early detection of pests and diseases. Additionally, the sector struggles with the mismatch between crop quality and market standards, weak coordination in supply chains, and the inefficient use of fertilizers and pesticides. These challenges result in resource wastage, increased costs, and reduced productivity. There is an urgent need to develop innovative solutions based on artificial intelligence to intelligently analyze agricultural data, provide accurate recommendations to enhance productivity, preserve natural resources, and contribute to the sustainability of the agricultural sector.

Unlike traditional methods, our thesis project focuses on harnessing the power of Artificial Intelligence (AI) by developing solutions for soil analysis and agricultural improvement. It also involves designing a system that recommends optimal fertilizer and pesticide dosages based on soil and crop data, as well as analyzing weather, soil, and plant data to provide accurate production forecasts. This is achieved through the development of an AI powered smart system that determines the optimal water quantities for each crop. By integrating advanced technologies, our system aims to optimize resource utilization, save valuable time, and enhance overall productivity. The focus on real time product and productivity monitoring sets our project apart, contributing to the advancement of the agricultural sector and strengthening food security.

The significance of this study lies in its contribution to the advancement of the agricultural sector through the integration of cutting edge technologies. By leveraging artificial intelligence and data driven methods, the project aims to introduce innovative solutions that can enhance productivity, optimize resource utilization, and promote sustainable agricultural practices. These efforts contribute to addressing key challenges in agriculture and support the broader goal of improving efficiency and resilience within the sector. Ultimately, this work aspires to foster progress and modernization in agricultural systems through the application of intelligent technologies.

This thesis consists of a general introduction, three main chapters, and a general conclusion.

Chapter 1: An overview of machine learning techniques This chapter provides a general overview of machine learning concepts and techniques, with a focus on classification algorithms as key tools in data processing. It also highlights the evolution from traditional methods to deep learning models, including recurrent neural networks such as GRU and LSTM.

Chapter 2: Latest technologies for identifying soil properties This chapter focuses on farmland soil as a vital component of agricultural productivity. It presents the main physical, chemical, and biological properties of soil, and explores modern techniques used to analyze these characteristics. Special attention is given to the role of artificial intelligence, remote sensing, and GIS in enabling accurate, data driven soil assessment.

Chapter 3: System architecture, experiments, and result analysis This chapter presents the development of the proposed system by outlining the main phases, including data preparation, model design using AI techniques, and experimental implementation. It also includes a performance analysis of the results and a comparison of different models to identify the one that provides the most accurate and efficient recommendations.

The system implementation is presented in Appendix A.

Chapter 1

An overview of machine learning techniques

1.1 Introduction

Artificial Intelligence (AI) is one of the most significant technological advancements of the modern era, becoming an essential part of various fields, from daily life to advanced industrial sectors. AI enables computers to perform tasks that require human like intelligence, such as learning, reasoning, and decision making. Machine Learning (ML) is a key branch of AI that allows computer systems to analyze data and learn from it without explicit programming. This advancement has led to remarkable improvements in multiple fields, including medicine, transportation, commerce, and cybersecurity. With the evolution of AI, Deep Learning has emerged as a powerful technique that relies on artificial neural networks to process vast amounts of data. This has enabled advanced applications such as facial recognition, natural language processing, and autonomous driving systems. Thanks to these technologies, AI has become a powerful tool for solving complex problems and enhancing efficiency across various sectors, paving the way for a future increasingly dependent on intelligent systems and self learning algorithms.

In this chapter, we will explore various types of machine learning and deep learning algorithms, including supervised and unsupervised learning. We will discuss classical machine learning algorithms such as Linear regression ,KNNs clustering, artificial neural networks (ANNs), transfer learning, as well as modern deep learning architectures like Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and Transformers.These techniques are essential for tackling complex tasks and improving real life applications.

1.2 Artificial intelligence

Artificial Intelligence (AI) is a field of study that concentrates on developing systems capable of mimicking human intelligence in decision making and learning. It includes technologies such as machine learning, natural language processing, and computer vision to enable machines to interpret and analyze data intelligently. The term was coined first in 1950 in Alan Turing's article "Computing Machinery and Intelligence,"(Weinert 2014) where he suggested the Turing Test for machine intelligence. In the Dartmouth

Conference of 1956, Marvin Minsky defined AI as ”programs that could carry out complex mental tasks such as learning, and logical reasoning.” AI through the years has evolved to systems that self learn and can make independent decisions. Modern AI is defined by processing and analyzing unstructured data without the need for human intervention.

It’s founded on advanced algorithms, such as deep learning and neural networks, therefore it can mimic human thought. AI is now prevalent in robotics, cybersecurity, medicine, and big data analysis. Due to these advances, AI has become a central part of modern technology, paving the way for a future that relies more on intelligent systems.(Soulez 2018)

The relationship between AI, ML, and deep learning is shown in the following visual form, as seen in Figure 1.1

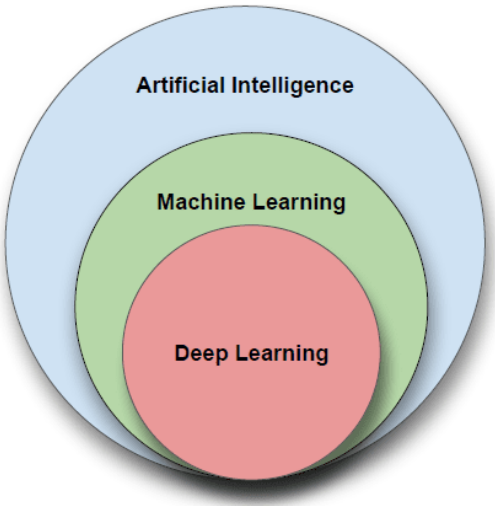


Figure 1.1: Relationship among AI and ML, Deep learning
(Jadhav 2024)

1.3 Machine learning

Machine Learning (ML) is an artificial intelligence discipline that centers on constructing systems capable of learning automatically from data without being specifically programmed. ML was initially termed by Arthur Samuel in 1959 as ”a field of study that gives computers the ability to learn without being explicitly programmed.” (McCarthy et al. 1990) ML taps into algorithms of data analysis to learn patterns, predict outcomes, and make decisions optimally. In contrast to conventional coding, ML allows computers to refine their performance over time by constantly learning from accessible data.

Machine learning has wide usage in fields such as healthcare, where it assists in the examination of medical records and genetics, finance for market research and trend forecasting, and cybersecurity for threat identification. Machine learning also plays a role in smart manufacturing to enhance productivity. In addition, ML powers general applications such as recommendation systems, image recognition, and virtual assistants, enhancing intelligent systems’ ability to solve complex problems and make sound and efficient decisions. (Kreuzberger et al. 2023)

1.3.1 Supervised learning

Supervised learning is a machine learning approach that relies on training a model using labeled data, where each training example consists of inputs paired with known outputs. The model aims to learn a function that maps inputs to outputs, allowing it to make predictions when presented with new, unseen data.

Supervised learning is categorized into two main types: **Classification**, where data is assigned to predefined categories. **Regression**, where continuous values are predicted.

This approach is widely used in applications such as facial recognition, text analysis, financial forecasting, and medical diagnosis. (Uddin et al. 2019)

1.3.1.1 Types of supervised machine learning

Classification

Classification is the process of assigning data to predefined categories using techniques that analyze patterns and extract key features from the data. Algorithms in classification work by defining decision boundaries that separate different categories to ensure accurate classification. This process will be further illustrated in the following Figure 1.2.(Osisanwo et al. 2017)

Stages of classification

1. **Data preparation** Cleaning and preprocessing data to make it suitable for analysis.

2. **Feature extraction** Identifying key characteristics that help distinguish between categories.

3. **Model building** Using algorithms like Decision Trees or Neural Networks to establish relationships between inputs and outputs.

4. **Training and learning**The model learns patterns from labeled data and understands how to classify each instance.

5. **Evaluation and improvement** The model is tested on new data, and its accuracy is measured using a Confusion Matrix or Classification Accuracy.

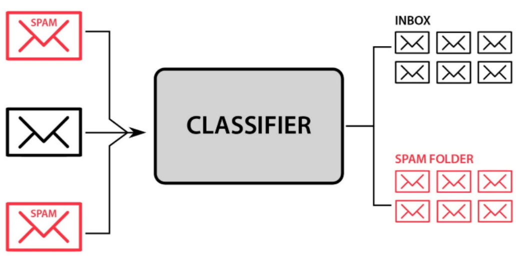


Figure 1.2: Stages of Classification in Machine Learning
(AnalytixLabs 2023a)

Regression

Regression involves predicting continuous numerical outcomes by modeling the relationship between input variables and a target value. Unlike classification, which assigns data to distinct categories, regression focuses on estimating precise values. This technique is commonly applied in scenarios such as forecasting prices, measuring growth, or estimating quantities. An illustration of the regression process is provided in the Fgure 1.3below. (Osisanwo et al. 2017)

Stages of regression:

1. **Data analysis** Identifying independent and dependent variables.
2. **Pattern extraction** Studying relationships between variables and their influence on outputs.
3. **Model building** Creating a mathematical equation that links inputs to outputs, such as **Linear regression**, which uses a straight line as a predictive model.
4. **Error reduction** Improving the model using techniques like **Regularization Regression** to prevent overfitting.
5. **Model testing** Evaluating prediction accuracy using **metrics like R^2 or Mean Squared Error (MSE)**.

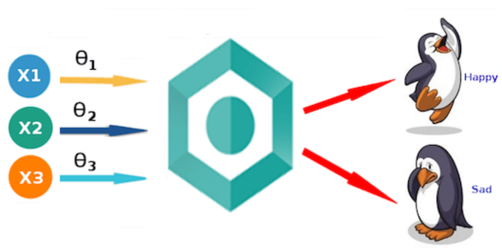


Figure 1.3: Stages of Regression in Machine Learning
(Polamuri 2017)

Key differences between classification and regression

The classification decides to which category a particular object belongs to. It focuses on the execution of input for pattern recognition and decides the category in which the input is to be classified. On the other hand, regression helps to predict the continuous valued output.

1.3.1.2 Common supervised learning algorithms

Linear classifiers

Linear classification aims to find linear decision boundaries that separate distinct categories within the feature space. It works by learning a classification function based on a linear combination of features, making it particularly effective in high dimensional settings. Linear models maximize the margin between classes to achieve proper classification, with optimal parameters determined by algorithms such as Logistic Regression or Support Vector Machines (SVM) with a linear kernel. These models are computationally efficient and easy to interpret. However, they can struggle with nonlinear data or cases where classes overlap significantly. In such situations, more advanced methods like nonlinear transformations or specialized kernels are often required for improved performance. The accompanying figure illustrates these concepts and the behavior of linear classifiers.(Osisanwo et al. 2017)

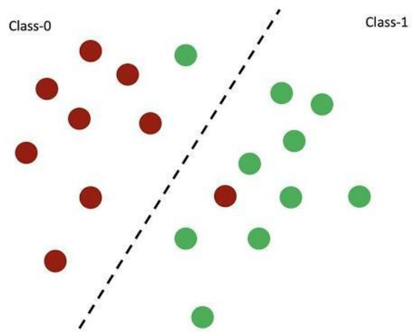


Figure 1.4: Linear Classifiers
(Starmer 2018)

Logistic regression (LR)

Logistic Regression is a supervised classification model that uses a logistic (sigmoid) function to estimate the probability of class membership. It is particularly effective for binary classification tasks, where a threshold (commonly 0.5) determines the predicted class. The model can also be extended to handle multi class problems through multinomial logistic regression. Logistic Regression defines decision boundaries in probabilistic terms, considering the distance of data points from those boundaries, which enhances prediction accuracy. While simple and computationally efficient, Logistic Regression assumes a linear relationship between the independent variables and the log odds of the dependent variable. This assumption can limit its effectiveness with complex, nonlinear data unless combined with feature engineering or transformations. Additionally, it may be prone to overfitting in high dimensional spaces, a challenge that can be addressed using regularization techniques such as L1 (Lasso) and L2 (Ridge). The concept is visually demonstrated in the accompanying figure, which illustrates how Logistic Regression estimates the probability of a data instance belonging to a class. If this estimated probability exceeds a predefined threshold, the model predicts the instance as part of that class otherwise, it does not making it an efficient binary classifier.(Uddin et al. 2019)

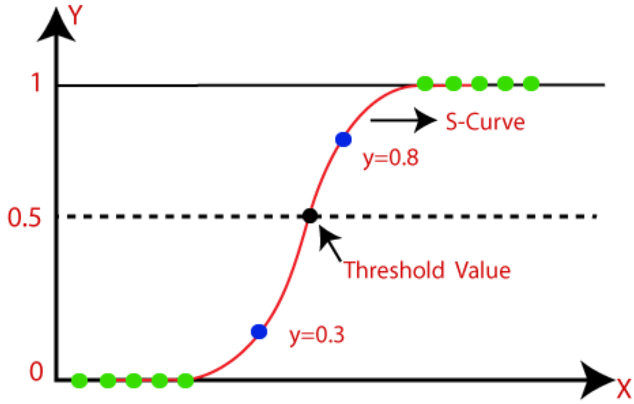


Figure 1.5: Logistic Regression
(FITA Academy 2023)

Naïve bayes algorithm

The Naïve Bayes (NB) algorithm is a probabilistic classification method grounded in Bayes' theorem, with the simplifying assumption that all features are conditionally independent given the class label. This assumption significantly reduces computational complexity, making NB highly efficient and scalable, especially for large datasets. The algorithm works by first estimating the prior probability of each class, then calculating the likelihood of each feature given the class, and finally combining these to compute the posterior probability, which determines the most likely class for a new instance. Despite its simplicity, Naïve Bayes often performs surprisingly well, particularly on datasets where features are genuinely independent or only weakly correlated. However, when strong dependencies exist between features, the model's assumption can lead to reduced accuracy. To mitigate this, enhanced versions like Averaged One Dependence Estimators (AODE) have been developed to partially relax the independence assumption and improve classification performance. This process is visually illustrated in the accompanying figure, which demonstrates how Naïve Bayes treats each feature independently when calculating probabilities, even in the presence of potential correlations. The simplicity and effectiveness of this approach make NB a practical choice for many real world classification tasks.(Uddin et al. 2019)

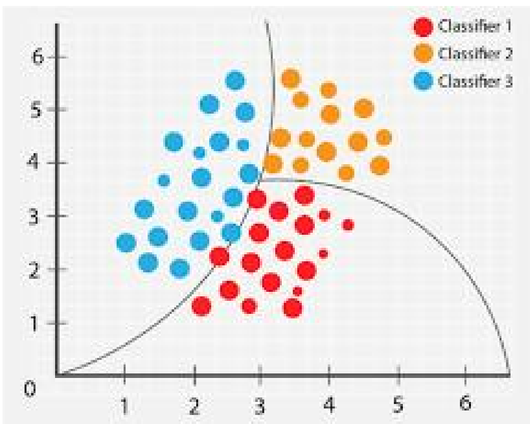


Figure 1.6: Naïve Bayes Algorithm
(FITA Academy 2023)

Support vector machine (SVM)

Support Vector Machine (SVM) is one of the most powerful supervised machine learning algorithms, widely used for both classification and regression tasks. SVM functions by identifying the optimal hyperplane that best separates different classes in a multi dimensional feature space. This hyperplane is chosen to maximize the margin the distance between the hyperplane and the nearest data points from each class, known as support vectors. A wider margin typically results in better generalization and lower error on unseen data. For datasets that are not linearly separable, SVM employs kernel functions such as linear, polynomial, and radial basis function (RBF) kernels to transform the original feature space into a higher dimensional one where a linear separation becomes possible. This kernel trick enhances SVM's flexibility in handling complex data structures. Despite its strong performance, particularly in high dimensional classification problems, SVM can be sensitive to noisy data and overlapping classes, which may affect its accuracy. This behavior is effectively illustrated in the accompanying figure, which shows how SVM constructs the decision boundary and utilizes support vectors to define the margin in both linear and transformed feature spaces.(Osisanwo et al. 2017)

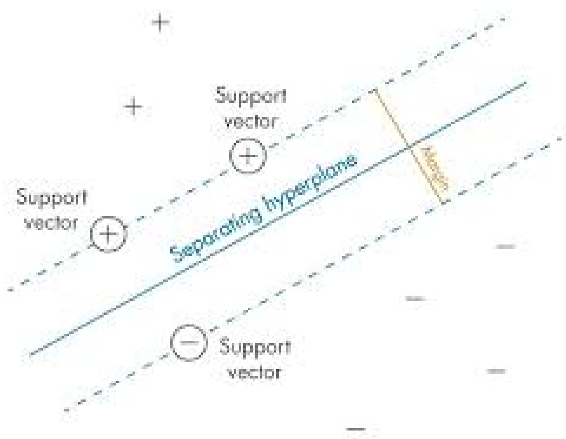


Figure 1.7: Support Vector Machine
(D. e. al. 2022)

K-Nearest neighbors (KNN)

The K-Nearest Neighbors (KNN) algorithm is one of the most intuitive supervised machine learning methods, commonly used for both classification and regression tasks. It operates on the neighborhood principle, where a new data point is classified or predicted based on the values or categories of its K nearest neighbors in the training set. To determine these neighbors, the algorithm calculates distances most often using metrics like Euclidean distance between the new data point and all existing ones. The predicted class is then decided by majority vote among the closest neighbors. KNN is a non parametric algorithm, meaning it does not rely on any underlying assumptions about the data distribution or relationships between variables. Its performance is highly sensitive to the choice of K: a small K may lead to overfitting and high sensitivity to noise, while a large K could result in underfitting by smoothing out significant patterns.(Osisanwo et al. 2017) Despite its conceptual simplicity and versatility, KNN can become computationally expensive, especially with large datasets, due to the requirement to compute distances for every prediction. The figure illustrates how the KNN algorithm functions when K =

4, showing how a new data point is classified based on its four nearest neighbors in the feature space.

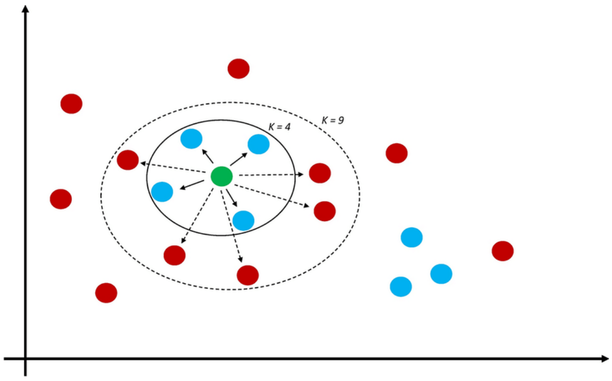


Figure 1.8: K-Nearest Neighbors
(Bicky 2025)

Decision tree (DT)

The Decision Tree (DT) is a widely used supervised machine learning algorithm that supports both classification and regression tasks. It operates by recursively partitioning the dataset into increasingly homogeneous subsets based on decision rules extracted from feature values. The process begins at the root node, which represents the entire dataset, and branches into internal nodes that apply decision rules derived from selected features. These branches continue to divide the data until reaching leaf nodes, which indicate the final predicted output—either a class label or a continuous value. The choice of feature for splitting at each node is guided by criteria such as entropy and information gain, used in algorithms like ID3 and C4.5, or Gini impurity, employed in CART. The recursive splitting continues until a stopping criterion is satisfied, such as achieving homogeneity within a node or hitting a maximum depth. To enhance generalization and reduce overfitting, pruning techniques are applied to remove branches that do not contribute significantly to the model's performance. Decision trees are known for their interpretability, ease of visualization, and rapid learning. However, they can be prone to overfitting, especially when the tree grows too deep or is trained on noisy data. As highlighted in the referenced Figure 1.9, the root node serves as the entry point of the tree structure, guiding subsequent splits and predictions through the decision paths. Well known decision tree algorithms like ID3, C4.5, and CART are widely used in diverse fields, including AI based classification, medical diagnosis, and data mining(Osisanwo et al. 2017)

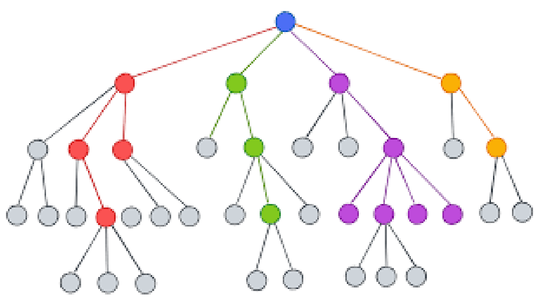


Figure 1.9: Decision Tree
(X 2023)

Artificial neural networks(ANN)

Deep learning algorithms primarily rely on artificial neural networks (ANNs), which are inspired by the architecture and functioning of the human brain. As a subset of machine learning, deep learning excels at identifying patterns and extracting meaningful insights from complex datasets without explicit human intervention. This autonomy makes it especially powerful in applications such as image recognition, natural language processing, and speech analysis. ANNs are composed of interconnected layers of nodes (neurons), typically organized into three main types: the input layer, which receives raw data; one or more hidden layers, where computations and feature extractions take place; and the output layer, which generates predictions or classifications. The learning process involves adjusting the weights of connections between neurons based on error feedback, allowing the network to improve its performance over time.(Ahamed et al. 2016) The referenced Figure 1.10 illustrates a multilayered artificial neural network, highlighting the structure and flow of data through the various layers. As seen in the diagram, the depth and complexity of the network can significantly influence its learning capacity, making deep learning models highly scalable and adaptable to a wide range of problems.

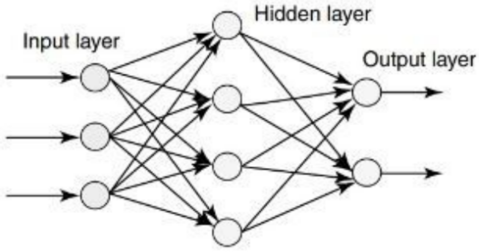


Figure 1.10: Artificial Neural Networks Layer
(B. e. al. 2018)

The network's performance is optimized by adjusting the weights during training to improve accuracy.

Working steps of artificial neural networks

Forward propagation

Forward propagation is the process through which input data is passed sequentially through a neural network, starting from the input layer and moving toward the output layer to generate predictions. Initially, the data is fed into the input layer, where each feature corresponds to a node in the network. Each input is then multiplied by an associated weight, and these weighted inputs are summed together. This sum is passed through an activation function, such as ReLU or Sigmoid, which introduces non linearity to the model, enabling it to learn complex patterns. This process continues through the hidden layers, transforming the data at each step, until it finally reaches the output layer where the network produces the final result based on the learned transformations(Ahamed et al. 2016). The structure of this process can be better understood by visualizing a single artificial neuron, as shown in Figure 1.11 which illustrates how inputs, weights, and activation functions work together in forward propagation.

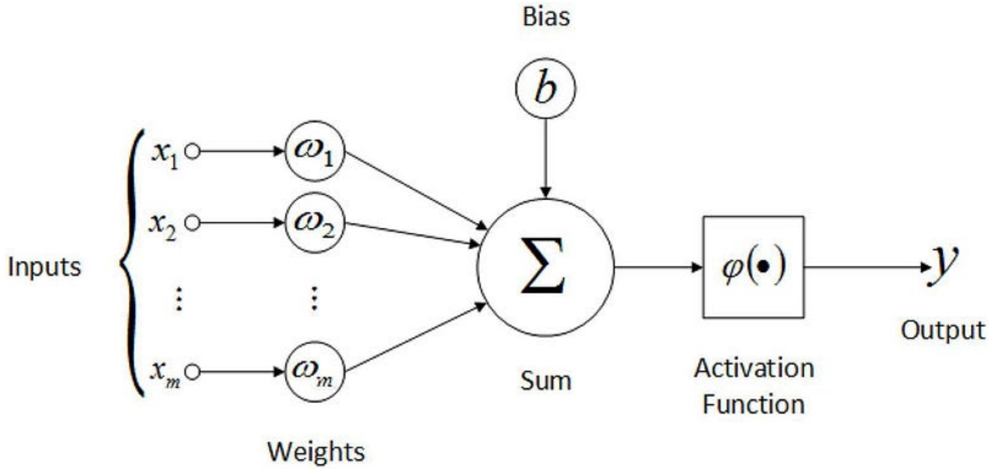


Figure 1.11: Forward Propagation Architecture
(Ashwin 2023)

Backpropagation

Backpropagation is the phase in neural network training where errors are corrected to improve performance by adjusting the weights. Initially, the network's output is compared with the actual values using a loss function, which quantifies the difference between predicted and true values. Then, the gradients of this loss function with respect to each weight are computed, enabling the gradual adjustment of weights through gradient descent to minimize the error and enhance the model's accuracy. This process of forward propagation (making predictions) followed by backward propagation (updating weights) is repeated over multiple epochs during training, allowing the model to progressively learn and capture the underlying patterns in the data.(Ahamed et al. 2016)

The Figure 1.12 provided clearly depicts this cycle, illustrating the flow of data forward through the network, the error computation, and the subsequent backward pass where weight adjustments occur, highlighting the repetitive and dynamic process by which the network learns from data.

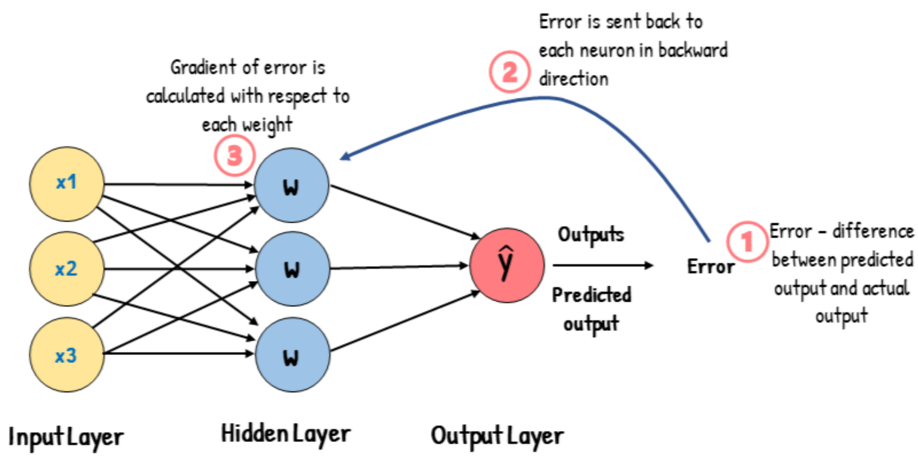


Figure 1.12: Backpropagation Architecture
(Sanchaari 2024)

Activation functions

Activation functions play a crucial role in artificial neural networks, determining whether a neuron should be activated or not. Without these functions, the network would be a simple linear model incapable of learning complex patterns. Below are some common activation functions:

Sigmoid Function

Used in tasks that require an output between 0 and 1, such as binary classification.

Converts values into a fixed range between 0 and 1, making it useful for probability estimation.

Mathematical formula:

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

Softmax Function

Primarily used in output layers for **multi-class classification** problems, converting values into probabilities that sum up to 1.

Allows the model to determine the class to which an input belongs based on computed values.

Unlike other activation functions, **Softmax does not work independently at each neuron level** but rather across all outputs at once, making it ideal for classification tasks.

The mathematical formula is:

$$\sigma(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{n} e^{z_j}}$$

where z_i represents a given input, and n is the number of classes.

The figure below demonstrates the Sigmoid and Softmax activation mechanisms commonly employed in machine learning models.

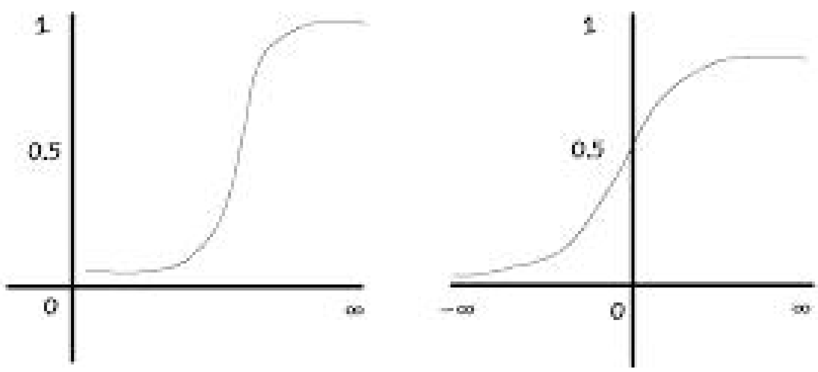


Figure 1.13: Sigmoid and Softmax Activation
(Gao et al. 2020)

ReLU (Rectified Linear Unit)

Keeps positive values unchanged while setting negative values to zero.

Widely used due to its simplicity and computational efficiency, but it may suffer from the **"dying neurons"** problem when all negative values are zeroed out.

The mathematical formula is given by:

$$f(x) = \max(0, x)$$

Leaky ReLU

Similar to ReLU, but allows small negative values instead of setting them to zero.

Helps mitigate the dying neurons issue found in ReLU.

The mathematical formula is given by:

$$f(x) = \max(0.1x, x)$$

PReLU (Parametric ReLU)

An improved version of Leaky ReLU, where the parameter controlling negative values is learned during training.

Provides more flexibility in determining which negative values should be retained.

The mathematical formula is:

$$f(x) = \max(ax, x), \quad \text{where } a \text{ is learned during training.}$$

The following figure illustrates the ReLU, PReLU, and Leaky ReLU activation mechanisms, highlighting their role in addressing the limitations of traditional activation functions.

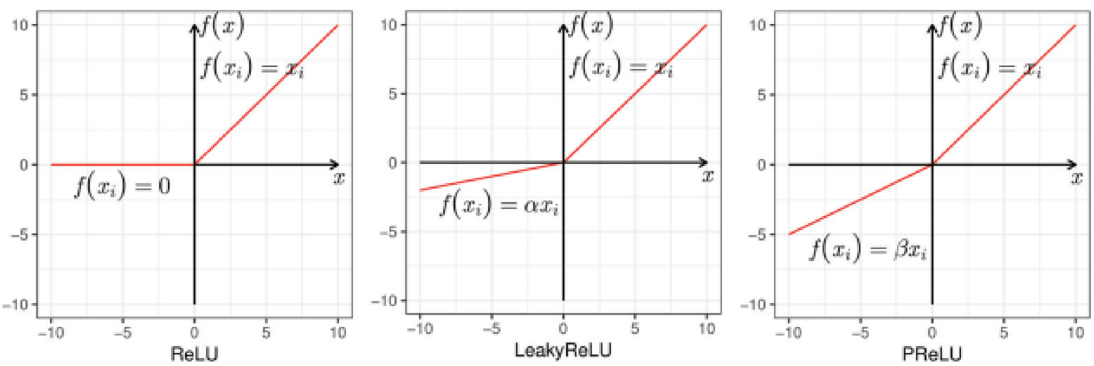


Figure 1.14: ReLU, PReLU, and Leaky ReLUActivation
(Gao et al. 2020)

Loss functions

Loss functions are metrics used to evaluate a model's performance by calculating the difference between the predicted values ($\hat{y}$) and the actual values (y). Each type of loss function is suitable for a specific task, such as regression or classification.

Mean Squared Error (MSE)

Used in **regression tasks**, where the goal is to predict a continuous numerical value.

Calculates the mean of the squared differences between actual and predicted values, making it highly sensitive to outliers.

Mathematical formula:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

where n is the number of samples, y_i is the actual value, and $\hat{y}_i$ is the predicted value.

Binary Cross Entropy (BCE)

Used in **binary classification**, where there are only two classes (e.g., 0 and 1).

Computes the probability that a sample belongs to one of the two classes using a logarithmic distribution, making it suitable for models using **Sigmoid** in the output layer.

Mathematical formula:

$$BCE = -\frac{1}{n}\sum_{i=1}^{n}\left[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)\right]$$

If the true label is $y_i = 1$, the function focuses on $\log(\hat{y}_i)$.
If $y_i = 0$, it focuses on $\log(1 - \hat{y}_i)$.
BCE minimizes the error when predictions are close to actual values (either 0 or 1).

Categorical Cross-Entropy (CCE)

Used in **multi class classification**, where there are more than two classes, such as recognizing handwritten digits (0-9).

Computes the loss across all classes using the **Softmax** function to generate class probabilities.

Mathematical formula:

$$CCE = -\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{C} y_{ij}\log(\hat{y}_{ij})$$

where C is the number of classes, y_{ij} is the correct class indicator (1 if the sample belongs to class j, 0 otherwise), and $\hat{y}_{ij}$ is the predicted probability for class j.

Table 1.1: Comparison of Loss Functions Based on Task Type

Task Type	**Suitable Loss Function**
Regression	MSE
Binary Classification	BCE
Multi Class Classification	CCE

Hyperparameters

Hyperparameters are values set before training the model and are not learned directly from the data. Their selection impacts the neural network's performance and learning efficiency. Below are some of the most important hyperparameters:

Learning rate: To update the weights of a model, a learning rate determines how much changes are needed, and that learning rate is very crucial to decide. Thus, the learning rate that is too high makes the model gradeers very much busy and also very unstable, while the learning rate that is low, makes the learning very slow; again, it might get stuck at the local minimums. So, by properly experimenting or using the proper techniques.

Regularization:

- Aims to prevent **overfitting**, where the model becomes overly adapted to the training data, leading to poor performance on new data.
- Common regularization techniques include:

  L1 Regularization (Lasso): Adds constraints on the absolute values of weights, leading to some weights becoming zero, effectively selecting the most important features.

  L2 Regularization (Ridge): Adds constraints on the sum of squared weights, helping to reduce large weight values and improve model generalization.

  Dropout: Randomly disables some neurons during training to prevent the model from relying too much on specific neurons.

Momentum:

- Used in **Gradient Descent** to speed up learning and reduce oscillations when updating weights.
- Helps overcome **local minima** by maintaining the "momentum" of previous updates, allowing it to continue in a specific direction rather than getting stuck at suboptimal points.

Optimizers:

- Algorithms used to update weights and improve model performance during training.
- Some key optimizers include:

  SGD (Stochastic Gradient Descent): Updates weights based on random samples, helping to avoid local minima but can be slow.

Adam Optimizer (Adaptive Moment Estimation): One of the most powerful optimizers as it combines the advantages of **Momentum** and **RMSprop**, using adaptive learning rates for each weight, making it highly efficient in various tasks.

RMSprop: Used to address **vanishing gradient problems** by adjusting the learning rate based on the average of past values.

Types of neural network architectures

Feed-forward neural network (FNN) Feed-forward Neural Networks (FNN) are the simplest type of neural networks, where data flows in only one direction, from the input layer to the output layer, without feedback loops or temporal memory. They are widely used in classification and regression tasks but are ineffective when dealing with sequential or time dependent data.(Iqbal et al. 2019)

Single layer perceptron (SLP) The Single Layer Perceptron (SLP) is one of the simplest neural network models, consisting of only an input layer and an output layer. It uses the Delta Rule to update weights. While it performs well for linearly separable problems, it cannot handle non linear data, making it inadequate for more complex tasks.(Iqbal et al. 2019) The accompanying figure illustrates the basic architecture of a perceptron, showing how input nodes connect directly to the output layer without any hidden layers. This visual highlights the simplicity of the SLP structure and helps explain why it is limited to solving only linearly separable problems.

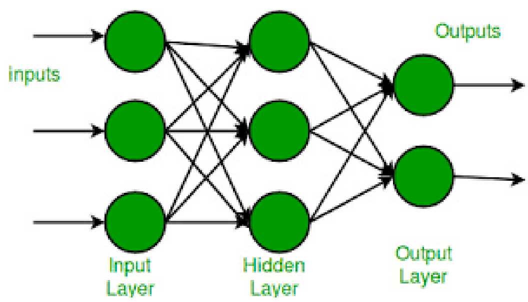


Figure 1.15: Single Layer Perceptron (SLP)
(Majdoubi et al. 2023)

Multi-layer perceptron (MLP) The Multi-layer Perceptron (MLP) is an improvement over the Single Layer Perceptron, incorporating hidden layers between the input and output layers. It utilizes backpropagation to update weights and gradually minimize errors. Thanks to non-linear activation functions like ReLU and Sigmoid, MLP can recognize complex patterns.(Iqbal et al. 2019) The figure shows how multiple layers of neurons are connected, with each layer transforming the input data through weighted connections and activation functions. This layered structure enables the network to learn and represent more intricate features compared to a single-layer model.

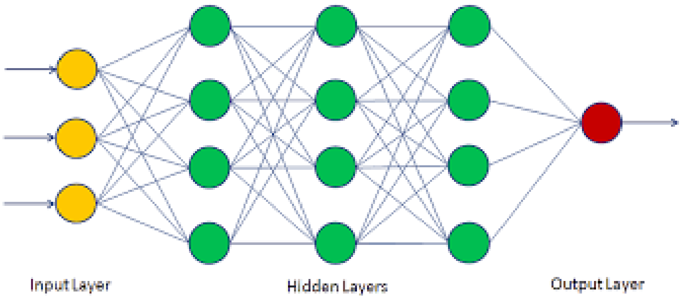


Figure 1.16: Multi-layer Perceptron (MLP)
(Mahmudiono et al. 2022)

1.3.2 Unsupervised learning

Unsupervised Learning is a branch of machine learning used to discover patterns in unlabeled data without the need for predefined outputs. These algorithms automatically analyze data and extract key features, helping to understand the intrinsic structure of the dataset. Unsupervised learning is widely applied in various fields, including dimensionality reduction, social network analysis, fraud detection, and image and text classification. It is also used in genome analysis to identify relationships between genes, providing new insights in medical research. Additionally, unsupervised learning helps in anomaly detection, making it a powerful tool for analyzing big data without human intervention.(Naeem et al. 2023)

1.3.2.1 Types of unsupervised Learning

Clustering

Clustering is an unsupervised learning technique used to group data points that are similar to each other into clusters. The main goal of clustering is to ensure that items in the same group are more similar to each other than to those in other groups. It helps reveal patterns, structures, or trends within the data. Clustering is widely applied in areas such as data analysis, image recognition, and customer segmentation. The primary aim is to organize and simplify large datasets by grouping related data points, often leading to more efficient and meaningful analysis. (Sharma et al. 2020) The figure visually demonstrates how data points can be organized into distinct groups based on their similarity, highlighting the ability of clustering algorithms to separate and define meaningful patterns within a dataset.

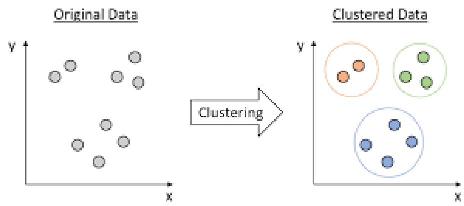


Figure 1.17: Clustering
(guoyee94 2024)

Anomaly detection

Anomaly Detection is the process of identifying unusual patterns or outliers in a dataset that significantly differ from the majority of data points. These anomalies can indicate rare or abnormal events such as network intrusions, hardware malfunctions, or corrupted data. For instance, unusual spikes in network traffic might signal data breaches or unauthorized access. Anomaly detection techniques are crucial in domains like fraud prevention, cybersecurity, and system health monitoring(Sharma et al. 2020). The figure illustrates different types of anomalies, including individual anomalies and collective anomalies where a linked group of data points together deviates from expected patterns highlighting how these unusual occurrences can be detected within complex datasets.

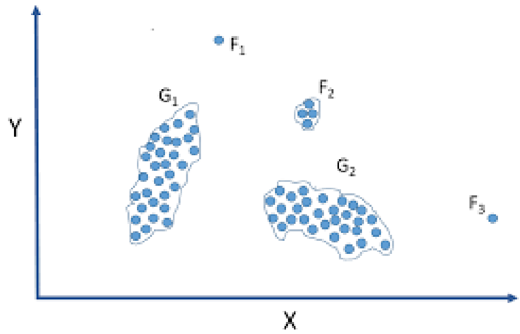


Figure 1.18: Anomaly Detection
(Vaitheswaran 2021)

Dimensionality reduction

In machine learning and data science, handling high dimensional data is a challenging task for researchers and developers. Therefore, dimensionality reduction, which is an unsupervised learning technique, is important as it leads to better human interpretability, reduced computational costs, and helps avoid overfitting and redundancy by simplifying models. Dimensionality reduction is a method for analyzing high dimensional data by reducing the number of variables, either by removing irrelevant features or combining several features into single elements, making the data more understandable. The main benefit of this approach is the removal of unnecessary aspects of the data, enabling easier visualization, such as in two dimensional plots. However, the combined features may become less interpretable, and some information is inevitably lost during the process.(Björklund et al. 2023)

1.3.2.2 Common unsupervised learning algorithms

K-Means

K-means is a popular clustering algorithm that partitions data into a predetermined number of clusters, denoted by K. It works by iteratively assigning data points to the nearest cluster centroid and then recalculating these centroids as the mean of points assigned to each cluster. This process continues until the cluster assignments stabilize and the centroids no longer change significantly. (Naeem et al. 2023])

The figure illustrates how K-means divides data into distinct clusters by grouping points around their closest centroids, visually representing the algorithm's step by step refinement of cluster boundaries.

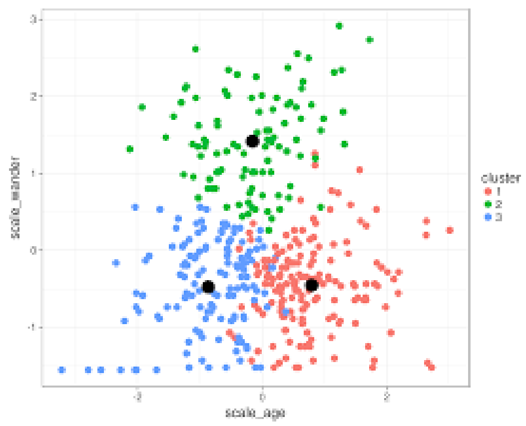


Figure 1.19: K-Means
(Czar 2014)

Principal component analysis (PCA)

Principal Component Analysis (PCA) is a powerful technique used in machine learning and data analysis for reducing the dimensionality of large datasets while preserving as much of the original variance as possible. It works by identifying directions (principal components) along which the variation in the data is maximized, and then projecting the data onto a smaller number of these components. This results in a simplified dataset that maintains the most significant patterns and structures. (Naeem et al. 2023)

The Figure 1.20 illustrates how PCA transforms high dimensional data into a lower dimensional space by extracting key features, enabling more efficient processing and clearer visualization without unnecessary noise or redundant information.

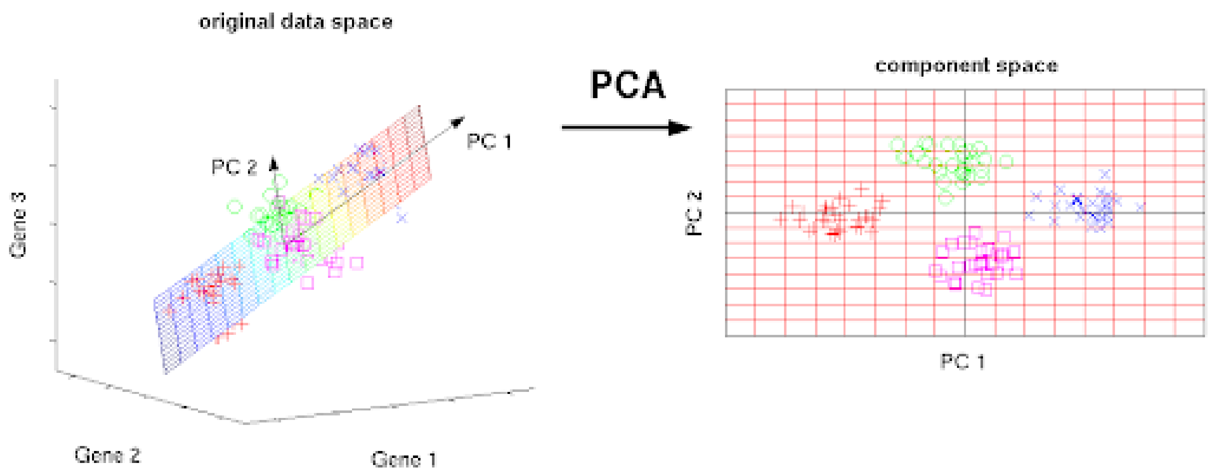


Figure 1.20: Principal Component Analysis
(Kato 2021)

1.3.3 Semi-supervised learning

Semi-supervised learning is an approach that combines both supervised and unsupervised learning, leveraging both labeled and unlabeled data. This method is particularly useful in machine learning and data mining, especially when obtaining labeled data is difficult or expensive, while large amounts of unlabeled data are available. The goal of semi-supervised learning is to improve prediction accuracy compared to using labeled data alone, making it suitable for applications such as anomaly detection, speech analysis, and internet content classification.(Mey et al. 2022)

1.3.4 Reinforcement learning

Reinforcement learning is a key area within machine learning focused on training agents to make sequential decisions by interacting with an environment. The agent learns to maximize cumulative rewards by receiving feedback in the form of rewards or penalties based on its actions.(Kaelbling et al. 1996)

This trial and error process enables the agent to improve its decision making strategy over time. Reinforcement learning has been successfully applied in domains such as robotics, autonomous systems, and industrial automation, where adaptive behavior is essential in dynamic and uncertain environments. (Sah 2020)

See the figure for a representation of the interaction between the agent and its environment.

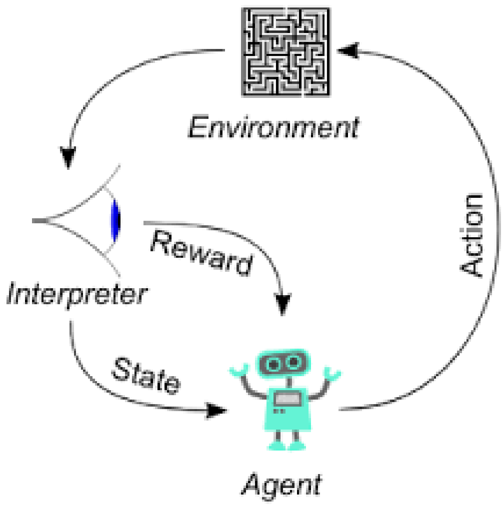


Figure 1.21: Reinforcement learning
(Modasiya 2022)

1.4 Deep learning

Deep learning constitutes a specialized domain within machine learning that leverages deep artificial neural networks, as illustrated in Figure 1.22. These networks consist of multiple hidden layers, enabling the extraction of intricate and high level features from data. Unlike conventional machine learning approaches, which rely on manual feature engineering, deep learning models autonomously learn hierarchical representations through successive nonlinear transformations. Deep learning techniques have been extensively applied across various domains, including cybersecurity, natural language processing, robotics, and image recognition. A principal advantage of deep learning lies in its capacity to process large volumes of data and discern complex patterns without explicit human intervention. Nonetheless, despite its demonstrated efficacy, deep learning demands substantial computational resources and large scale datasets to attain optimal performance.(Hordri et al. 2016)

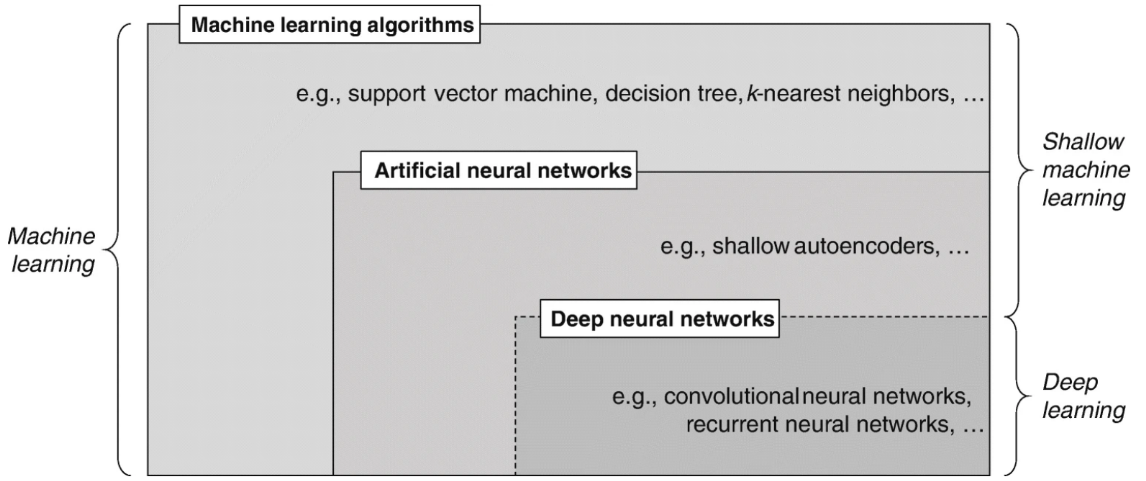


Figure 1.22: Classification Diagram of Machine Learning and Deep Learning Algorithms (Janiesch et al. 2021)

1.4.1 Difference between deep learning and machine learning

Table 1.2: Comparison Between Deep Learning and Machine Learning

Feature	**Deep Learning**	**Machine Learning**
Approach	Relies on artificial neural networks to extract patterns and hidden relationships in data.	Uses statistical algorithms to detect patterns and relationships in data.
Data Requirement	Requires a significantly large amount of data.	Can work with relatively smaller datasets.
Best Use Cases	Suitable for complex tasks like **image processing** and **natural language processing (NLP)**.	Suitable for simpler tasks with **limited data**.
Training Time	Takes longer to train models.	Requires less time to train models.
Feature Extraction	Features are extracted automatically, making it an **end-to-end learning system**.	Features are manually extracted to create a model capable of recognizing patterns.
Complexity & Interpretability	More complex and **less interpretable** due to multiple layers of nonlinear transformations.	Provides **more interpretable** results as it relies on clear rules or simpler models.
Hardware Requirements	Requires **high-performance computers** with **GPUs** for efficient training.	Can run on **CPUs** and requires **less computational power** compared to deep learning.

Deep learning is an advanced evolution of machine learning, offering superior performance in complex tasks but requiring high computational resources and large datasets. In contrast, machine learning is simpler, more interpretable, and works effectively with smaller datasets and lower computing power.

1.4.2 Benefits of deep learning

Deep Learning has revolutionized artificial intelligence, enabling the development of intelligent systems capable of perception, understanding, and interaction with the world in unprecedented ways. It is used in various applications such as self driving cars, virtual assistants, personalized recommendations, and advanced robotics. Below are its key advantages:

Automatic feature learning: Deep learning can extract features directly from data without the need for manual design, making it ideal for complex tasks such as image and speech recognition.[(Bengio et al. 2009)

Ability to handle large and complex Data: Deep learning algorithms efficiently process large, high dimensional datasets, making them a powerful tool for extracting valuable insights from complex data.(Schmidhuber 2015)

Superior performance: Deep learning algorithms have demonstrated outstanding results in various fields, including pattern recognition, computer vision, and natural language processing.(Krizhevsky et al. 2012)

Discovering non linear relationships: Deep learning excels at identifying non linear relationships between data, a capability that traditional methods struggle to achieve. (Goodfellow et al. 2016)

Handling structured and unstructured Data: Deep learning can analyze different types of data, whether structured, like databases, or unstructured, such as images, text, and audio recordings.(Zhang et al. 2020)

Predicting future trends: Deep learning can build models capable of forecasting future trends, aiding businesses and organizations in strategic planning and decision-making.(Kim et al. 2018)

Handling incomplete data: Deep learning models can manage missing or incomplete data, making them highly useful in real world applications where data may be inconsistent or evolving.(Hardy et al. 2018)

Sequential data analysis: Networks like RNN and LSTM excel at processing sequential data such as text, audio, and time series data, enhancing contextual understanding and improving decision-making accuracy.(Nofal et al. 2014)

Scalability and deployment: Deep learning models can be easily scaled to handle increasing amounts of data and can be deployed on cloud platforms or edge devices for real world applications.(Dean et al. 2012)

Adaptability to new scenarios: Deep learning models generalize well and adapt to new situations and datasets, making them highly flexible for solving a variety of problems. Thanks to these advantages, deep learning has become the foundation of many modern intelligent technologies, contributing to the development of systems capable of learning, analyzing, and making decisions with unprecedented efficiency.

1.4.3 Deep neural networks

Artificial Neural Networks (ANNs) are the foundational models of Deep Learning, designed to mimic the way the human brain processes information. In the context of deep learning, ANNs consist of multiple layers of interconnected neurons that enable the network to learn complex, non linear patterns from data,as seen in Figure 1.23.

These layers include an input layer, several hidden layers (making it "deep"), and an output layer. Each neuron performs computations involving weighted inputs and activation functions, allowing the model to learn hierarchical features through forward propagation and adjust its internal parameters via backpropagation. This architecture enables ANNs in deep learning to excel at tasks such as image classification, natural language processing, speech recognition, and more, especially when dealing with large scale and high dimensional data.(Santos et al. 2025)

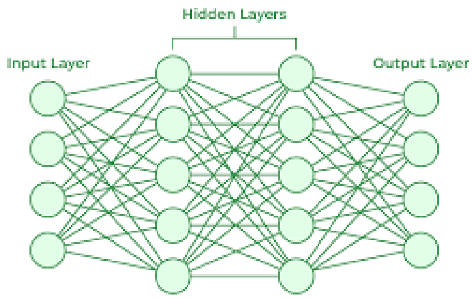


Figure 1.23: Artificial Neural Networks Layer
(Bhatnagar 2023)

1.4.3.1 Convolutional Neural Networks (CNN)

Convolutional neural networks (CNN) are an advanced type of artificial neural network primarily used for **image and video analysis**. They are highly effective in **automatic feature extraction** from visual data, making them ideal for tasks such as **face recognition, scene analysis, and image classification**.(Purwono et al. 2022)

Key components of CNN:

Input Layer: Receives raw image data and converts it into a numerical matrix representing pixel values.

Convolutional Layer: Applies filters to extract features such as edges, textures, and patterns. Each filter acts as a detector for specific features, enabling the network to recognize detailed structures.

Activation Layer: Applies a non linear activation function (such as ReLU) after convolution, highlighting important features while discarding irrelevant ones.

Pooling Layer: Reduces the dimensionality of extracted features to improve efficiency and reduce computational complexity. Common types include:

- **Min Pooling:** Retains the minimum value in the specified region. It is less common but used in certain applications to extract less obvious features.
- **Max Pooling:** Retains the maximum value in a pooling region, preserving essential features while reducing size.
- **Average Pooling:** Computes the average value in a pooling region, simplifying information while minimizing detail loss.
- **Stochastic Pooling:** Randomly selects values from the pooling region instead of using max or average values, enhancing generalization.
- **Global Pooling:** Applied across the entire feature map instead of specific regions, producing a compact representation of extracted features.

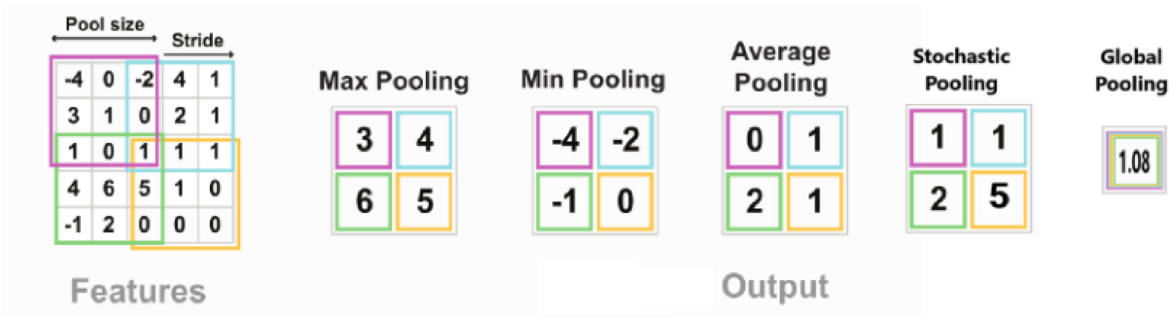


Figure 1.24: CNN pooling types

Dropout Layers: Prevent overfitting by randomly disabling some units during training, improving model stability.

Fully Connected Layers (FC Layers): Transform extracted features into a final representation used for classification or decision making.

The key components of a convolutional neural network (CNN) are best understood by examining its fundamental layers. These layers play a crucial role in the functioning of a CNN and are essential for grasping its architecture and learning process. An overview of these core layers is presented in Figure 1.25.

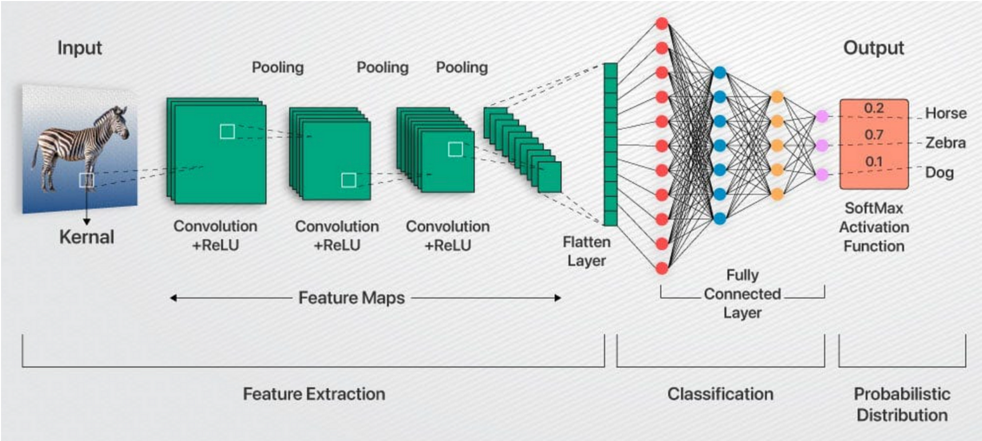


Figure 1.25: CNN architecture
(AnalytixLabs 2023b)

A detailed overview of the most popular Convolutional Neural Network (CNN) architectures.

LeNet-5 (1998): The first successful Convolutional Neural Network (CNN), developed for handwritten digit recognition in the MNIST dataset, consisted of two convolutional layers followed by two pooling layers and fully connected layers; it used the Tanh activation function and was limited to processing low resolution images.

AlexNet (2012): A deep CNN with 5 convolutional layers and 3 fully connected layers introduced key improvements such as ReLU activation for faster training and Dropout to reduce overfitting; it achieved great success in the ImageNet Challenge, significantly contributing to the widespread adoption of CNNs in computer vision.

VGGNet (2014): VGG networks, such as VGG-16 and VGG-19, rely on small sized filters (3×3) but use many layers to enable detailed feature extraction; while their architecture is simple and easy to understand, they require high memory and computational power, making them less efficient compared to newer architectures.

GoogLeNet / Inception (2014): The Inception network introduced Inception Modules, which use filters of different sizes (1×1, 3×3, 5×5) within the same layer to enable diverse and efficient feature extraction; it also utilized 1×1 convolutions to reduce the number of parameters and improve computational speed, resulting in a deeper architecture (22 layers) that is more resource efficient than VGGNet.

ResNet (2015): ResNet introduced Residual Connections, which allow layers to be skipped when necessary, making it easier to train very deep networks like ResNet-50 and ResNet-152; by solving the Vanishing Gradient problem that limited the depth of traditional networks, ResNet enabled the creation of architectures up to 152 layers deep without sacrificing performance or significantly increasing computational complexity.

MobileNet (2017): MobileNet was designed to be lightweight and fast, making it ideal for mobile devices and applications requiring high energy efficiency; it uses Depthwise Separable Convolutions to significantly reduce computational operations compared to traditional networks while maintaining good performance, with further enhancements introduced in MobileNetV2 to optimize data flow and memory efficiency.

EfficientNet (2019): EfficientNet is a highly optimized CNN developed using AutoML techniques to balance network size and depth efficiently; it delivers high performance compared to previous networks while consuming fewer resources, making it ideal for AI applications requiring computational efficiency, and it employs a Compound Scaling approach that balances network width, depth, and resolution to maximize feature extraction.

Comparison of architectures

Table 1.3: Comparison of Popular CNN Architectures

Architecture	Layers	Key Features	Drawbacks
LeNet-5	7	First CNN, suitable for simple images	Not suitable for large scale data
AlexNet	8	First deep CNN, used ReLU & Dropout	High memory consumption
VGG-16/19	16-19	Small 3×3 filters, high accuracy	Slow and resource intensive
GoogLeNet (Inception)	22	Multi-size filters, high efficiency	Complex design
ResNet-50/101/152	50-152	Residual connections, very deep networks	Requires significant computational power
MobileNet	Varies	Lightweight and fast for mobile devices	Lower accuracy than deep networks
EfficientNet	Varies	High performance with optimized efficiency	Requires careful tuning

Advantages of CNN:

Reduces the need for **manual feature engineering**.

Exploits **spatial structure** of data, improving pattern recognition efficiency.

Computationally efficient for image related tasks compared to traditional networks.

Learns from **large datasets**, making it powerful for complex classification tasks.

Challenges of CNN:

Requires **high computational power**, especially for high resolution images.

Needs **large training datasets** to achieve good performance.

Prone to **overfitting** if not properly tuned.

Sensitive to data quality **noise or errors in images** can affect performance.

Not well suited for sequential data, limiting its application in text and audio processing.

1.4.3.2 Recurrent neural networks (RNN)

Recurrent Neural Networks (RNNs) are a class of deep neural networks specifically designed to process sequential or time series data by maintaining a hidden state that captures information from previous time steps. This architecture enables RNNs to model temporal dependencies, making them particularly effective for applications such as natural language processing (NLP), speech recognition, and time-series forecasting. By transmitting information across time steps, RNNs learn patterns that unfold over sequences. However, standard RNNs face significant limitations, most notably the vanishing gradient problem, which impedes their ability to capture long range dependencies. To address this challenge, advanced variants like Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) were introduced. These architectures incorporate gating mechanisms that selectively retain or discard information over time, thereby improving performance on complex, long-sequence tasks (Sherstinsky 2020). The structure and functioning of RNNs are outlined in Figure 1.26.

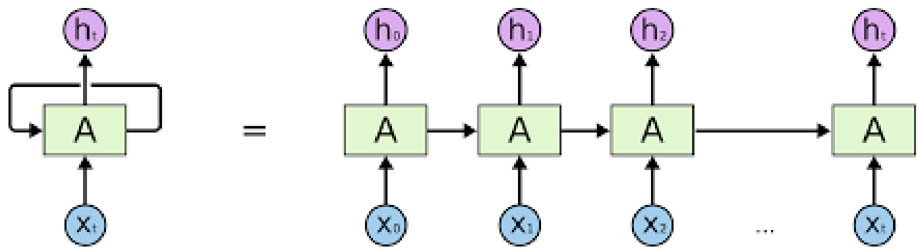


Figure 1.26: RNN Architecture
(Sung Kim 2016)

Advantages of RNN:

Suitable for **sequential data** such as text, speech, and video.

Retains **context information** from previous inputs.

Useful for applications like **machine translation, sentiment analysis, and time series forecasting**.

Challenges of RNN:

Suffers from the **Vanishing Gradient Problem**, making deep RNNs hard to train.

Struggles with long range dependencies due to **information loss over time**.

Long Short-Term Memory (LSTM) Networks

LSTM is an advanced version of RNN designed to solve the **short-term memory loss problem**. It features **long-term memory storage**, allowing it to handle sequential data more effectively. LSTM uses a set of **memory cells** that regulate information flow through **three main gates**:

Input Gate: Controls how much new information is added to memory based on its importance.

Forget Gate: Determines how much old information should be discarded to maintain optimal performance.

Output Gate: Defines what information should be passed to the next layers for future predictions.

How LSTM Works:

1. **Receiving Inputs:** The unit receives sequential data values and processes them through the input gate.
2. **Identifying Important Information:** The forget gate decides whether past information is still relevant or should be discarded.
3. **Updating Stored State:** The internal state of the cells is adjusted based on new accepted data and retained information.
4. **Producing Outputs:** The output gate determines which data is sent to the next layers, allowing the model to predict future outcomes based on contextual understanding.

Gated Recurrent Units (GRU)

GRU is an improved version of RNN, similar to LSTM in handling sequential data but **simpler in design**, using only **two gates** instead of three. This makes it **faster and less resource intensive**.

Update Gate: Controls how much new information should be retained and what should be forgotten.

Reset Gate: Helps regulate how much past information influences the current state.

How GRU Works:

1. **Receiving Data:** Upon new input, the update gate determines which information should be retained from previous iterations.
2. **Controlling Temporal Influence:** The reset gate decides how much past values should influence current predictions.

3. **Computing the New State:** The remaining values are merged to update the cell state.

4. **Producing Outputs:** The updated state generates the final result and passes it to the next layers.

Below, we outline the primary architectural contrasts between LSTM and GRU networks.

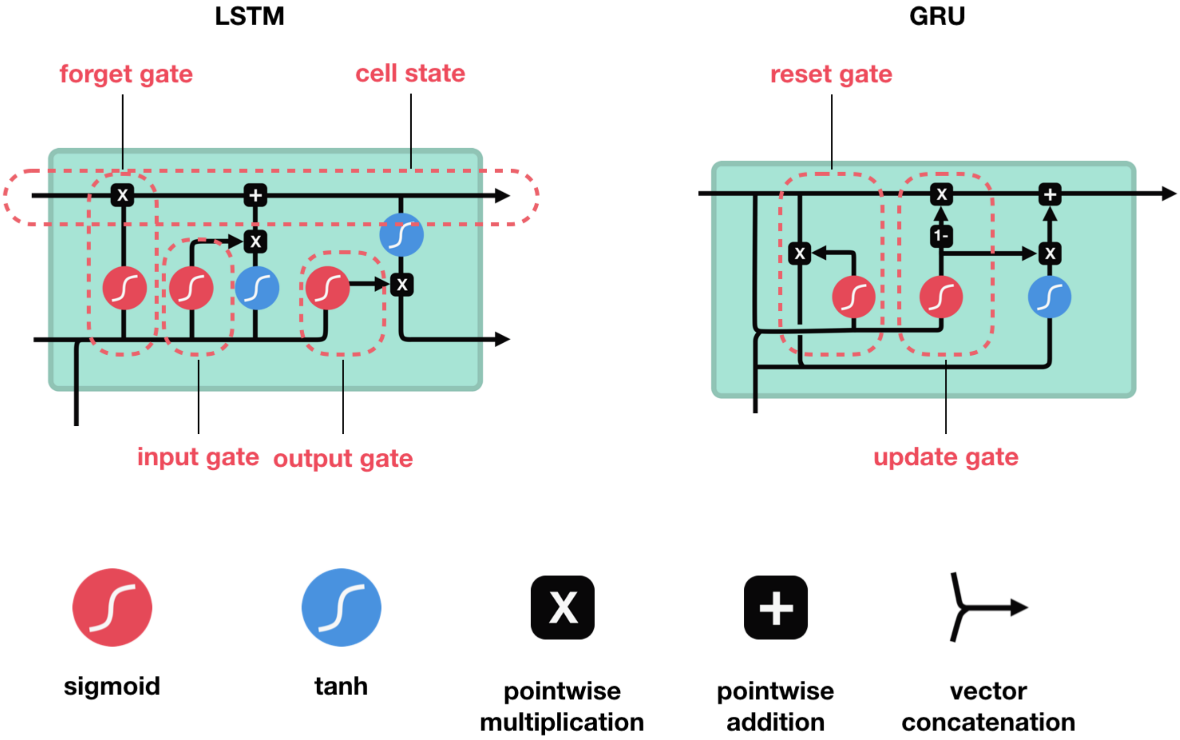


Figure 1.27: LSTM and GRU architecture
(Wang 2024)

Comparison Between LSTM and GRU:

Criterion	LSTM	GRU
Number of Gates	3 (Input, Forget, Output)	2 (Update, Reset)
Training Speed	Slower due to complexity	Faster due to simplicity
Resource Consumption	Higher	Lower
Handling Long Sequences	More efficient	Slightly less efficient

Key applications of LSTM and GRU:

Text analysis and sentiment recognition in artificial intelligence.

Neural machine translation for improving language understanding.

Time-series forecasting in economic predictions and financial market movements.

Speech processing and voice recognition, enhancing smart assistants like Siri and Google Assistant.

GRU is widely used in applications such as **text processing, machine translation, time-series forecasting, and speech recognition**. It is often preferred when **higher execution speed and lower resource consumption** are required.

1.4.4 Transformers

Transformers are an advanced type of deep learning model, first introduced in the Attention is All You Need paper in 2017 by Vaswani et al. Originally designed for sequence based tasks such as machine translation, they have become fundamental in various fields including Natural Language Processing (NLP), Computer Vision (CV), and speech signal analysis. The Transformer architecture is based on self-attention mechanisms, which allow the model to assign different weights to each input element, enabling a better understanding of context. This structure eliminated the need for Recurrent Neural Networks (RNNs), improving processing speed and reducing issues such as vanishing gradients. The model consists of an encoder-decoder architecture, where the encoder processes the input and transforms it into an internal representation, while the decoder uses this representation to generate the desired output, such as translated text or predictive sentences. The emergence of Transformers led to the development of powerful models such as BERT, GPT, and ViT, which have demonstrated outstanding performance across multiple domains.(Islam et al. 2024)

This is visually represented in Figure 1.28, which shows the Transformer architecture.

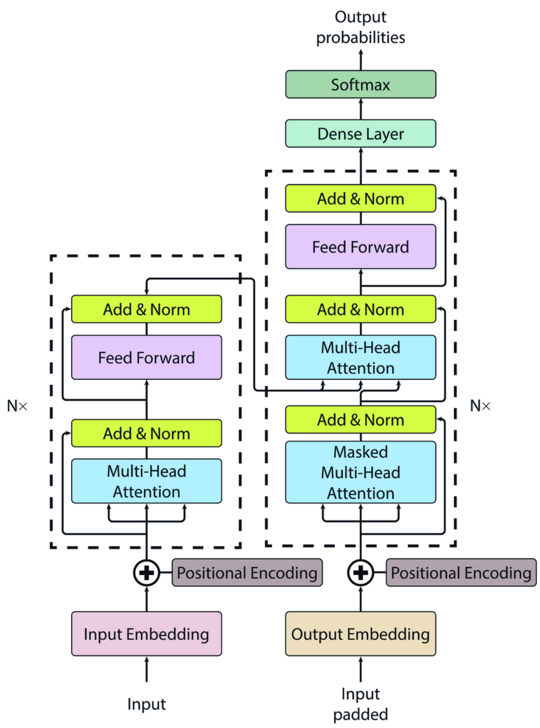


Figure 1.28: Transformer Architecture
(Zhumagambetov et al. 2021)

1.5 Ensemble learning

Ensemble Learning is a machine learning technique that enhances model accuracy and reduces errors by combining the predictions of multiple models rather than relying on a single one, thus reducing bias, variance, and improving generalization. Effective ensemble learning depends on appropriate data sampling, either through Independent Datasets training models on separate subsets or Dependent Datasets where subsets rely on previous ones, as in Boosting.

The training of baseline classifiers follows either Sequential Learning (Boosting) to correct errors progressively or Parallel Learning (Bagging, Random Forest) to reduce variance via independent models. Fusion methods determine how predictions are merged, using approaches like Hard Voting (majority vote), Soft Voting (probability averaging), or Weighted Voting (based on model performance), while Meta-Learning, such as Stacking, uses a meta-learner to optimally combine base model outputs.

Common ensemble methods include Bagging, which reduces variance by training models on bootstrapped data and combining results via voting or averaging Random Forest is a key example but may not lower bias due to shared weaknesses among models. Boosting, including AdaBoost, Gradient Boosting, and XGBoost, builds models sequentially to correct errors and improve accuracy, though it risks overfitting and is computationally intensive. Stacking leverages diverse models and a meta learner to integrate predictions, yielding high performance but requiring careful tuning and resources. Voting, either Hard or Soft, is simple and beneficial for aggregating predictions of independent models but may falter when model quality varies.

Each method suits different needs: Bagging excels in reducing variance, Boosting focuses on bias reduction, Stacking combines diverse strengths for peak performance, and Voting offers a straightforward ensemble strategy.

Table 1.4: Comparison of Different Ensemble Learning Methods

Method	**Data Dependency**	**Primary Goal**	**Advantages**	**Disadvantages**
Bagging	Independent Samples	Reduce variance	Improves stability	Does not reduce bias
Boosting	Dependent Samples	Reduce bias	Enhances accuracy	Prone to overfitting
Stacking	Diverse Samples	Model combination	Strongest performance	Complex and slow
Voting	Independent Samples	Simple combination	Easy to implement	Does not analyze model relationships

1.6 Conclusion

In this chapter, we provided a comprehensive analysis of learning techniques in artificial intelligence, starting from traditional machine learning approaches to advanced deep learning techniques. These include artificial neural networks (ANNs), deep architectures such as recurrent neural networks (RNNs) and convolutional neural networks (CNNs), as well as transformer models. We also discussed the concept of transfer learning and its significance in enhancing model performance across various tasks. Additionally, we explored practical applications of these techniques, such as sentiment analysis, machine translation, and pattern recognition, demonstrating the significant impact of AI in solving complex problems.

In the next chapter We will explore various techniques and methods to identify soil properties.

Chapter 2

Latest technologies for identifying soil properties

2.1 Introduction

Soil of farmland is a central element of agriculture, directly affecting crop output and sustainable environment. Science of soil or the science to investigate the soil has evolved significantly since early 1800s based on contributions from other sciences.(Brevik et al. 2015)

Farm soil refers particularly to soil used for crops and animal care and requires close monitoring and care. It possesses physical, chemical, and biological properties which affect the growth of crops such as texture, nutrient availability, pH value, and microbial density. Identification and management of these properties are crucial for sustainable agriculture.(Lal 2011)

Visual inspection and manual touch have been the conventional methods practiced by farmers over time, while modern-day technologies like GIS, remote sensing, spectroscopy, and AI driven software now enable precise, real-time soil analysis to aid effective decision making and resource management.(Wadoux et al. 2020)

In this chapter, we explore the definition and classification of agricultural soils, discuss their key physical, chemical, and biological properties, and present both classical and modern techniques used to analyze these properties. Furthermore, we highlight how artificial intelligence and machine learning are being leveraged to improve soil analysis and crop recommendations. Finally, we review relevant datasets and recent research works that support smart agriculture and sustainable soil management.

2.2 Definition of agricultural soil

Soil science, established in the 1800s, is the study of soils and their properties. It has developed a specialized language to describe soil, reflecting advances in disciplines like geology and microbiology. The definition of soil has evolved over time as scientific knowledge has expanded. Research in soil science now extends to various fields, influencing policy and public understanding. Agriculture, derived from Latin terms for ”field cultivation,” encompasses soil cultivation, crop growing, and animal raising. It includes practices such as horticulture, pastoralism, and mixed farming. The field has seen shifts in terminology to reflect different farming methods.(Brady et al. 2008)

Agricultural soil refers to the type of soil used specifically for farming, which requires careful management to maintain soil fertility and structure. It plays a critical role in crop production and sustains agricultural productivity. Agricultural soil can be classified into different types, such as sandy, clay, or loamy, each with distinct characteristics affecting drainage, nutrient content, and suitability for different crops.(Harris et al. 2013)

Proper management of agricultural soil includes practices like crop rotation, fertilization, and soil conservation to prevent degradation and ensure long-term sustainability. Soil science plays a vital role in agricultural productivity and environmental sustainability.(Harris et al. 2013)

2.3 Type of agricultural soil properties

Soil is one of the essential elements in agriculture, as its physical, chemical, and biological properties directly impact the growth and quality of crops. Understanding these properties helps farmers and researchers improve agricultural practices and ensure sustainability.

2.3.1 Physical properties of soil

The physical properties of soil include its mineral composition, which refers to the proportions of sand, clay, and silt in the soil. Clay soils are excellent for retaining water, while sandy soils drain water effectively. Loamy soils strike a balance between water retention and aeration, making them ideal for most plants. Permeability and aeration describe the soil's ability to allow water and air to pass through, with soils of high permeability being favorable for root growth. Water holding capacity determines how much water the soil can store for later use, with clay soils retaining water longer than sandy soils.(Akinde et al. 2020)

2.3.2 Chemical properties of soil

Chemically, the soil's pH level significantly influences nutrient availability, with most plants thriving in soils with a pH between 6 and 7. The cation exchange capacity (CEC) is another important chemical property, indicating the soil's ability to hold positively charged nutrients. Soils rich in organic matter tend to have a high CEC, aiding in nutrient supply for plants. Soil salinity, the concentration of salts in the soil, can negatively impact plants' ability to absorb water when it is too high.(Akinde et al. 2020)

2.3.3 Biological properties of soil

Biologically, organic content plays a vital role in improving soil structure and increasing its water retention capacity. This organic matter consists of decomposed plant and animal matter that enriches the soil. Microbial activity, which includes the work of bacteria and fungi, is crucial for breaking down organic matter and recycling nutrients. Additionally, soil biodiversity the variety of living organisms in the soil enhances its overall health and quality, contributing to its fertility and sustainability.(Bai et al. 2018)

2.4 Identifying soil properties

2.4.1 Classical methods for identifying soil properties

2.4.1.1 Visual inspection

Visual soil assessment is an important tool for evaluating the quality of agricultural soil, relying on the visual inspection of properties such as structure, root depth, and moisture to determine the extent of degradation. Systems used in this evaluation include VESS (Visual Evaluation of Soil Structure), which focuses on soil porosity and the presence of aggregates or compacted areas, VSA (Visual Soil Assessment), which incorporates tests like the drop test to assess soil clod disintegration, and M-SQR (Müller Soil Quality Rating), which evaluates soil based on a range of properties affecting its agricultural performance. Assessment methods can target only the topsoil layer (VESS), only the subsoil layer (SubVESS), or both layers through comprehensive evaluation, helping farmers make informed decisions regarding soil management practices such as fertilization and irrigation. Additionally, visual soil assessment plays a role in identifying environmental risks, including nitrous oxide emissions and soil related degradation.(Ball et al. 2017) An overview of the various soil types is presented in Figure 2.1.

Figure 2.1: Soil Classification
(Student Projects 2025)

2.4.1.2 Touch and feel method

The Touch and Feel Method is one of the oldest techniques used to identify soil properties, relying on the physical senses rather than instruments. In this method, soil scientists use their hands to assess soil texture by rubbing, squeezing, and rolling moistened soil samples. They estimate clay content based on cohesion and plasticity, feeling how easily the soil forms ribbons or balls. Sand content is judged by the grittiness and how much volume remains when particles are separated by washing. Loam, silt, and clay soils each exhibit distinct tactile sensations. The method requires experience to develop accuracy, as soil moisture, organic matter, and mineral content can affect feel. Traditional flowcharts help guide beginners through the process by offering step by step touch based observations. Although considered subjective, this method remains valuable in fieldwork due to its simplicity and speed. It provides a quick preliminary assessment, especially when lab testing is unavailable.(Santanello Jr et al. 2007)

The soil hand texturing process is demonstrated in Figure 2.2.

Figure 2.2: Touch and Feel Method
(UBC Wiki 2025)

2.4.1.3 Soil pH testing with litmus paper

Soil pH testing with litmus paper is a simple and cost-effective method to measure the acidity or alkalinity of soil. By creating a soil water suspension and dipping the litmus paper into the solution, the paper changes color to indicate the pH level, helping to determine if the soil is acidic (below 7) or alkaline (above 7). This test is useful for gardeners and farmers to assess soil health, as pH affects nutrient availability and overall plant growth. While not as precise as laboratory tests, it provides a quick way to decide if soil amendments, such as lime or sulfur, are needed to adjust the pH for optimal crop production.(McLean 1982)

The figure depicts the procedure for measuring soil pH

Figure 2.3: Testing Soil pH with pH Papers
(STEMpedia 2025)

2.4.1.4 Smell test

The Smell Test in agricultural soil is a sensory method used to assess soil health based on the aroma emitted by the soil. Research has shown that healthy soils, often managed with soil health-promoting practices, tend to produce a more pleasant and distinct smell compared to unhealthy soils. This smell is linked to volatile organic compounds (VOCs) released from the soil, which vary depending on soil management and moisture levels. By analyzing these VOCs through techniques like gas chromatography mass spectrometry olfactometry (GC-MS-O), scientists can identify patterns that correlate with soil health. Higher soil moisture reduces the intensity and variety of soil aromas over time, with optimal VOC expression found at around 25% soil moisture. Thus, the Smell Test provides a quick, intuitive, and research-backed way to help evaluate the biological quality of soil.(Bonds et al. 2022)

2.4.2 Modern methods for identifying soil properties

2.4.2.1 Laboratory soil testing

Laboratory soil testing in agriculture is a standardized method used to evaluate soil fertility, pH levels, lime requirements, and nutrient availability to guide effective crop management. Over the past 30 years, soil testing has become essential for farmers and growers, helping them make informed decisions about fertilizer and lime application. Different soils require different testing methods, making standardization critical to ensure accurate and reliable results. Resources like the Soil Analysis Handbook of Reference Methods provide detailed laboratory procedures for testing soil properties, including newer areas like nitrate levels, heavy metals, and quality assurance practices. By following these standard techniques, soil testing laboratories support sustainable farming and improved plant nutrition management around the world.(Jones 2018)

Figure 2.4: Reliable Soil Testing
(Hayatok 2025)

2.4.2.2 Geographic Information System (GIS) and remote sensing

Geographic Information Systems (GIS) and Remote Sensing (RS) are essential technologies in modern agriculture, especially for monitoring soil health and crop conditions. Remote sensing collects data about the Earth's surface using satellite imagery, aerial photography, and other sensors. These technologies help monitor large areas, offering insights into factors like soil fertility, plant health, and land use changes. GIS is a computer based tool that captures, stores, analyzes, and displays geographic data. It integrates spatial data from remote sensing, field surveys, and historical data for better analysis and visualization. In vineyards, GIS and RS enable precise monitoring of soil, weather, and crop growth. These technologies help optimize irrigation, predict crop yields, and improve land use. Advances in remote sensing, like high-resolution satellites and multispectral drones, enhance data accuracy. By integrating GIS and RS, farmers can make informed, data driven decisions to improve sustainability and crop yield (Grishin et al. 2020). The Figure 2.5 demonstrates how Variable Rate Technology (VRT) operates in conjunction with GIS and RS systems.

Figure 2.5: Applications of (GIS) Geoinformatics in Agriculture
(Piddubna 2022)

2.4.2.3 Portable soil sensors and probes

Portable soil sensors and probes are advanced technologies designed for on site, real-time soil nutrient analysis, making them essential for precision agriculture. These devices allow for the quick and cost effective determination of soil macronutrients and micronutrients, which are critical for optimal crop growth. By offering portability, these sensors enable rapid nutrient testing and mapping of soil nutrient distribution across fields, addressing issues like over or under application of fertilizers. Unlike conventional soil testing methods, which are time-consuming and expensive, portable sensors provide an efficient alternative, offering low cost, on the go solutions. With advancements in technology, these devices require minimal sample preparation, and their real-time data capabilities allow farmers to make informed decisions and optimize fertilizer use. Additionally, these sensors are integrated into the Internet of Things (IoT), enabling continuous monitoring and predictive analytics for better resource management and sustainable farming practices. Techniques such as optical, electrochemical, and capillary electrophoretic sensors are commonly used, with machine learning and artificial intelligence enhancing their performance (Pal et al. 2024). Figure 2.6 shows a soil nutrients testing sensor device used to measure key soil parameters, enabling precise soil health monitoring and informed agricultural decision making.

Figure 2.6: Portable Soil Analyzer with Multi Probes
(Alibaba n.d.)

2.4.2.4 Soil spectroscopy

Soil spectroscopy is an emerging analytical technique in agricultural soil management that uses visible, near infrared, and mid infrared (Vis-NIR-MIR) reflectance to rapidly and non destructively assess soil properties. Over the past 30 years, it has proven to be a fast, cost effective, and environmentally friendly alternative to traditional soil testing methods. Unlike conventional laboratory analyses, which are time consuming and expensive, soil spectroscopy allows for the quick assessment of important soil parameters like organic matter, nutrient content, texture, and moisture levels. Its application both in the laboratory and increasingly in the field, using portable devices and airborne or satellite sensors, has the potential to revolutionize large scale soil health monitoring. However, challenges remain in reducing prediction errors, standardizing methods across different operators, and building robust, diverse spectral databases. By addressing these issues, soil spectroscopy could become a routine tool for soil assessment at field, national, and even continental scales, helping to support sustainable land management and tackle global challenges like food security, climate change, and environmental degradation.(Nocita et al. 2015)

2.5 Utilization of machine learning and artificial intelligence

AI and machine learning (ML) are revolutionizing agricultural soil management by enabling more precise and efficient practices for soil analysis and irrigation. Traditional soil analysis methods were time consuming and costly, relying on lab based techniques. However, AI-driven tools now process complex data from sensors, satellites, and geographical systems, predicting soil properties like texture and moisture content with greater accuracy and speed. (Awais et al. 2023)

ML models, including regression algorithms like XGBoost and random forest, help predict nutrient requirements, moisture levels, and optimize irrigation systems, ensuring crops receive the right amount of water at the right time. This reduces water waste, improves crop yields, and makes irrigation smarter and more resource efficient. Additionally, AI and remote sensing create comprehensive soil property datasets, enabling precision agriculture that supports data-driven decision making for improved productivity and sustainability. ML techniques like decision trees, support vector machines (SVM), K-nearest neighbors (KNN), and neural networks further enhance soil management by classifying soil types, forecasting moisture for irrigation, detecting soil erosion or contamination, and recommending targeted fertilizer and pesticide use. These innovations not only boost crop yield predictions but also reduce environmental impact, promoting resource conservation and paving the way for sustainable agriculture worldwide.(Dey et al. 2024)

2.6 Review of related research

2.6.1 Datasets

Agricultural soil management data sets are structured collections of soil measurements gathered over diverse ecosystems for research and practical applications in agriculture and environmental conservation. The typical data sets include such soil features as texture, water content, fertility, pH, and organic matter, supplemented by remote sensing or climate parameters. Each data point is tagged with suitable agronomic or environmental metadata so that machine learning and AI models can be trained, tested, and validated. Due to their size, scope, and generality, such datasets have assumed added significance as a mandatory tool for researchers, agronomists, and machine learning experts. They are essential to building precision agriculture by enabling the development of predictive models that assess soil health, optimize fertilization, detect degradation, and recommend sustainable agriculture practices. With the use of such abundant datasets, AI-powered soil management systems become more robust and efficient, ultimately improving crop yield, reducing environmental impact, and enabling improved agricultural decision making.

2.6.1.1 Smart Farming Data 2024 (SF24)

The SF24 dataset comprises 4,800 records with 28 features (23 original and 5 derived) covering soil, climate, and crop-related parameters. It includes labeled data for various crops under diverse environmental conditions. The dataset is organized in tabular format and includes features such as soil nutrients (N, P, K), temperature, humidity, pH, rainfall, and derived indices like THI, NBR, and SFI. Basic preprocessing was applied; no data augmentation was performed. This dataset is suitable for applications in crop classification, environmental stress analysis, and predictive modeling for smart agriculture.(Engineer 2024)

2.6.1.2 Crop Recommendation dataset

This dataset contains 2,200 records with 7 features related to soil and climate. It includes data for 22 crops, with about 100 samples each. The data is organized in tabular form. Basic preprocessing was applied. No data augmentation was performed. It is suitable for use in crop prediction, recommendation systems, and machine learning model training.(Ingle 2021)

2.6.1.3 Crop Recommendation using Soil Properties and Weather Prediction

This dataset includes soil and climate features collected from multiple Ethiopian sources. Soil data contains attributes like pH, color, nutrients, and coordinates. Climate data includes temperature, rainfall, humidity, wind, and pressure, sourced from NASA cloud infrastructure. Crop types are mostly cereals. Data was collected based on location and season. No augmentation was applied. It is suitable for use in crop prediction, recommendation systems, and machine learning model training.(TmleynCodes 2021)

2.6.2 Recent works

2.6.2.1 The work of Cao et Al.2021

In this study, the authors developed a county level rice yield prediction model for China by integrating diverse publicly available datasets using the Google Earth Engine (GEE) platform. They combined satellite-derived vegetation indices such as Enhanced

Vegetation Index (EVI) from MODIS and Solar-Induced Chlorophyll Fluorescence (SIF) from GOSIF, along with their combination (ESI), to capture both canopy structure and photosynthetic activity. Meteorological data including minimum and maximum temperature, precipitation, PDSI, PET, VPD, and degree day metrics were obtained from CHIRPS, ERA5, and TerraClimate. Soil attributes like pH, texture, and organic matter were accessed via SoilGrids. Ground-truth rice yield data (2000 to 2015) were collected at the county level from government statistical yearbooks. Prior to modeling, extensive preprocessing and data trimming were performed to remove outliers, align spatial temporal resolutions, handle missing values, and normalize variables.

Three models, LASSO regression, Random Forest (RF), and Long Short-Term Memory (LSTM) networks, were evaluated. LSTM achieved the highest prediction accuracy, with R^2 values ranging between 0.77 and 0.87, which is prediction accuracy of between 77% and 87%, and RMSE ranging from 298.11 to 724 kg/ha. RF followed with slightly lower values, while LASSO performed the worst with R^2 values between 0.33 and 0.42. The combination of EVI and SIF (ESI) marginally improved performance, as it better captured drought and heat stress. Optimal predictions were achievable one to two months before rice maturity. This framework offers a scalable, efficient, and timely approach for large-area yield estimation, adaptable to other regions with limited observational records. (Cao et al. 2021)

2.6.2.2 The work of Ali et Al. 2021

In this study, a Pakistan specific crop recommendation system is proposed to improve agricultural productivity using machine learning techniques. The system focuses on assisting farmers in selecting the most suitable crop before planting based on temperature and seasonal trends. Agriculture plays a major role in Pakistan's economy, and a large portion of land is cultivated for both local use and export. The proposed model helps prevent losses from planting unsuitable crops and supports seasonal classification for better decision-making. The system uses a custom built, localized dataset collected specifically for the Nawabshah district (Sindh, Pakistan).

The dataset includes crop yield data for wheat, cotton, sugarcane, and rice from local agricultural departments and research institutes. Temperature readings were obtained from government weather stations, and seasonal classifications (late winter, summer) were added for context. Soil properties such as texture, fertility, and chemical composition were gathered from local soil testing labs. Machine learning models were trained on this dataset with temperature as the key predictive input. Experimental results showed that the proposed model achieved 90% accuracy, which is higher than existing approaches. This proves its value for local farmers and national food security. The system is practical, scalable, and adaptable to other regions facing similar challenges with limited data availability.(Ali et al. 2021)

2.6.2.3 The work of Ahmad et Al. 2023

Soybean (Glycine max) is a proteinaceous oilseed crop used extensively for cooking oil and poultry feed but is subjected to significant challenges as a result of unfavorable global climatic conditions exacerbated by the current climate crisis. The purpose of this research was to determine optimal areas for soybean production in a semi-arid terrestrial environment to facilitate effective land use planning. Using geostatistical interpolation, some data layers such as soil characteristics, quality of irrigation water, land use and land cover, topography, and climate were generated and merged with weight factors derived from the Analytic Hierarchy Process. To construct these layers, the study relied on several publicly accessible datasets, including historical climate data (temperature and rainfall from Open Data Pakistan, covering a 116-year series from 1901–2016), Sentinel 2 satellite imagery for land-use and vegetation classification, administrative boundaries and GIS layers from PakData/GISData and OSM shapefiles, and soybean related NDVI and soil moisture insights from the USDA Soybean Explorer. Land use and land cover classification overall accuracy was 70% and Kappa value was 0.61, indicating that it has good precision. Soybean suitability confirmation using the Receiver Operating Characteristic (ROC) curve showed an excellent area under the curve (AUC) of 0.738. The research revealed that out of 172,618.66 hectares, almost 47.46% is highly suitable (S1), 21.36% moderately suitable (S2), 11.91% marginally suitable (S3), 7.00% not presently suitable (N1), and 12.28% not suitable at all (N2). Overall, the results reveal that the study location has favorable climatic conditions, healthy soils, and quality irrigation water availability, conducive to soybean production with improved agronomic practices.

The research provides valuable information for farmers and policymakers to maximize the management of irrigation, increase agricultural yields, and minimize soil degradation.(N. Ahmad et al. 2023)

2.6.2.4 The work of Pokhariyal et Al. 2023

This article presents a systematic review exploring how machine learning (ML) techniques are being integrated with remote sensing (RS) data to support agricultural monitoring across India. It highlights the effectiveness of ML algorithms in extracting meaningful insights from various RS datasets to improve agricultural practices in areas such as crop classification, yield prediction, disease detection, weed monitoring, nutrient mapping, soil salinity and moisture estimation, as well as drought and water stress assessment.

The research surveyed a wide range of RS inputs including data from Landsat, Sentinel 2, PRISMA, SAR (Synthetic Aperture Radar), and thermal sensors, often used alongside vegetation indices like NDVI and additional climate parameters. These data sources were processed using ML models such as Support Vector Machines (SVM), Random Forests (RF), Artificial Neural Networks (ANNs), Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), Support Vector Regression (SVR), and Gradient Boosting Regression (GBR). Results across the reviewed studies were highly promising for example, CNNs achieved crop classification accuracies of up to 99%, and GBR models for yield prediction reported R^2 values as high as 0.94. High accuracy was also observed in disease identification, weed detection (over 82% accuracy), water stress monitoring (over 90% accuracy), and soil quality assessment (R^2 values exceeding 0.9).

The findings underscore the transformative potential of integrating RS technologies with advanced ML algorithms to enhance precision agriculture. This integration supports timely, accurate, and scalable decision making, offering significant benefits for agricultural productivity and resource management in India.(Pokhariyal et al. 2023)

2.6.2.5 The work of Deyet Al. 2024

Machine learning (ML) can sufficiently utilize agricultural data for crop production under various levels of soil nutrients and climatic conditions to recommend suitable crops or additional nutrient requirements for enhanced productivity. In this study, the performance of five ML models was evaluated using a publicly available dataset from Kaggle titled Crop Recommendation Dataset. The dataset includes parameters such as NPK values, soil pH, and climatic factors namely temperature, rainfall, and humidity collected under real field conditions.

The ML models tested include Support Vector Machine (SVM), XGBoost, Random Forest, K-Nearest Neighbors (KNN), and Decision Tree. These models were trained using yield related data from 11 farm crops, 10 horticultural crops, and a combined dataset incorporating both categories. Results revealed that separate training on farm and horticultural datasets yielded better prediction accuracies compared to the combined dataset. Among all models, XGBoost consistently outperformed the rest, achieving 99.09% accuracy (AUC 1.0) for farm crops, 99.3% (AUC 1.0) for horticultural crops, and 98.51% (AUC 0.99) on the combined dataset.

This non intrusive, data-driven ML approach demonstrates the potential of integrating AI in agriculture through user friendly, cloud based systems to enable rapid and precise decisions for optimal crop selection and fertilizer application across diverse agro-environmental scenarios. The outcomes contribute to advancing sustainable productivity and supporting precision agriculture practices through intelligent crop management. (Dey et al. 2024)

2.7 Conclusion

Effective management of agricultural soil is essential for ensuring long term agricultural productivity and environmental health. Understanding the physical, chemical, and biological characteristics of soil enables farmers to adopt sustainable practices tailored to specific conditions. While traditional methods still provide valuable insights, modern technologies such as portable sensors, soil spectroscopy, and AI-powered models have transformed how soil properties are monitored and utilized. These advancements support precision agriculture, enhance crop yields, and reduce environmental impact. By integrating conventional knowledge with innovative tools, the future of soil management can meet global food demands while conserving natural resources.

In the following chapter, we will explain how machine learning can be used to develop a system that identifies the most suitable crop type for a specific soil, highlighting the ongoing advancements in leveraging artificial intelligence technologies to build reliable and efficient systems.

Chapter 3

Organizational Structures, Results, and Reflections

3.1 Introduction

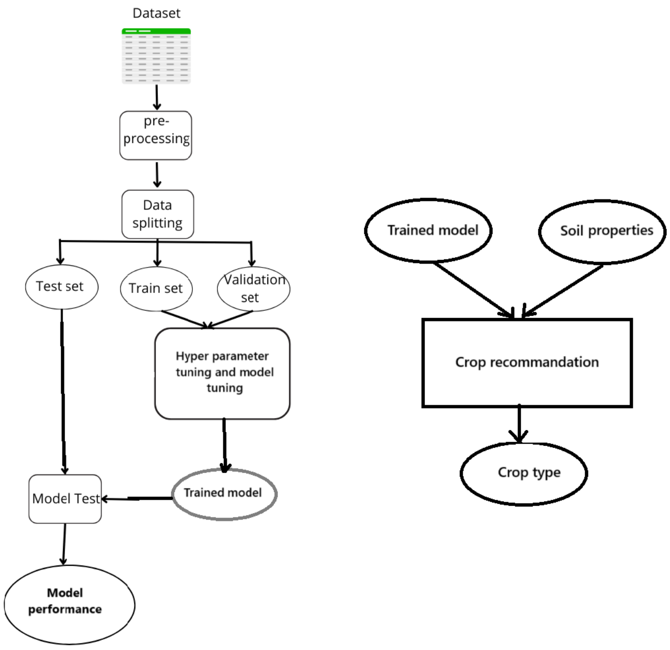
With the continuous advancement of artificial intelligence and smart agriculture technologies, it has become possible to develop systems that analyze soil data to provide precise agricultural recommendations. Our project aims to create an intelligent system capable of identifying the most suitable type of crop that can be cultivated on a given plot of land by analyzing soil characteristics such as pH level, moisture, nutrient content, and other physical properties. This data is processed using AI models, including machine learning techniques, to draw accurate conclusions about the most appropriate crop for the current soil conditions. The system contributes to improved agricultural decision making, increased productivity, and reduced resource waste.

In this chapter, we present the components of the proposed system, starting with data collection and preprocessing, followed by the design and testing of the model, and concluding with the analysis of results and comparison of different models to select the one that delivers the highest recommendation accuracy.

3.2 General system architecture

Our system follows a typical workflow in machine learning, which is illustrated through a diagram showing four main stages, as depicted in (Figure 3.1). At the beginning, we use a training database that contains soil elements and materials, such as soil type and temperature, necessary for training the model. Next, we perform data preprocessing. Then, the model is trained on a portion of the processed training data. The resulting model is later used in the testing phase to determine which type of plant is suitable for that land.

3.3 Detailed presentation of our system



(a) Training (b) Classification

Figure 3.1: General schema of the proposed approach

3.3.1 Data preparation

3.3.1.1 Data preprocessing

In our system, preprocessing of tabular data is an essential step to ensure that the dataset is clean, consistent, and suitable for training machine learning models. This process involves handling missing or duplicate values, encoding categorical variables, and normalizing or scaling numerical features to ensure they are within a comparable range. We also check for outliers and perform feature selection or transformation as needed. These steps are critical for improving the quality of the data, reducing noise, and enabling the model to learn patterns more effectively and make accurate predictions.

3.3.1.2 Data splitting

After completing data preprocessing, it is essential to divide the dataset into three distinct parts: the **training set**, **validation set**, and **test set**. This separation is critical to ensure the generalizability of our approach and to accurately assess model performance using data the model has not seen during training.

Training set

This portion of the dataset (60%) is used to train the model. The model learns directly from this data by adjusting its internal parameters. A sufficiently large training set is important to ensure the model can learn meaningful patterns and achieve strong performance.

Validation set

The validation set (20%) is used during the training process to fine-tune the model's hyperparameters (learning rate, batch size). It helps evaluate the model's ability to generalize to unseen data and detect issues like overfitting. By monitoring performance on the validation set, we can make necessary adjustments to improve the model's robustness.

Test set

The test set (20%) is reserved for final evaluation after the model has been fully trained and validated. It provides an unbiased assessment of the model's performance on completely unseen data, simulating how it would perform in real-world scenarios. Keeping the test set separate ensures a fair and accurate measurement of the model's ability to generalize and make reliable predictions.

3.3.2 Model training

Once the dataset is divided into training, validation, and test sets, the model is trained using the training data. During this process, the model learns patterns from the input features in order to make accurate predictions. The quality of these predictions is measured using a loss function, which guides the overall learning process. Training occurs over several iterations or structured procedures, depending on the model architecture, allowing the model to improve its performance. The validation set is used to monitor how well the model is learning and to adjust key settings, known as hyperparameters, to avoid overfitting. These may include learning rate, regularization strength, or model specific structural constraints. The training continues until the model's performance stabilizes. Final evaluation is conducted on the test set to assess the model's ability to generalize to new, unseen data.

Each classification model undergoes a distinct training process tailored to its underlying learning strategy:

Naïve bayes classifier

This probabilistic model is trained by calculating prior and conditional probabilities directly from the training data. It assumes feature independence and uses Bayes' theorem to compute the posterior probabilities for classification. Model training primarily involves estimating likelihoods from feature distributions.

Decision tree classifier

The model is trained by recursively partitioning the feature space based on information gain, Gini index, or entropy. At each node, the best feature and threshold are selected to maximize the purity of resulting subsets. Training continues until stopping criteria such as maximum depth or minimum samples per leaf are met, with pruning techniques applied to avoid overfitting.

Support Vector Machine (SVM)

SVM training involves finding the optimal hyperplane that maximizes the margin between classes. This is achieved by solving a convex optimization problem, often using techniques like Sequential Minimal Optimization (SMO). For non linear data, kernel functions are employed to map input features into higher dimensional spaces.

Logistic regression classifier

This model is trained using gradient descent or its variants to minimize the binary (or multi-class) cross entropy loss. The sigmoid (or softmax for multi-class) activation is applied to predict class probabilities. During training, the model iteratively updates weights to best fit the decision boundary.

Long Short-Term Memory (LSTM)

LSTM networks are trained on sequential data using sequence based optimization. Input sequences are processed, predictions are generated, and errors are measured by a loss function. Optimization algorithms like Adam or RMSprop are used to update the network's weights. Training occurs over multiple iterations (epochs), often with techniques like dropout and early stopping to reduce overfitting.

Gated Recurrent Unit (GRU)

Similar to LSTM, GRU networks are also trained on sequential data. GRUs simplify the internal structure while maintaining the ability to learn temporal dependencies. The model is optimized by minimizing a loss function over sequences, with weights updated using gradient-based methods and training conducted across multiple iterations.

3.3.3 Model test

After training our crop recommendation model, we evaluate its performance on the test set to assess its effectiveness on unseen data. We use various performance metrics such as accuracy, precision, recall, and F1-score to measure the model's capabilities. This thorough evaluation compares the model's predictions with the actual crop recommendations, ensuring its reliability and adaptability to new data. The use of these metrics guarantees optimal performance in crop prediction, providing a balanced assessment of the model's strengths and weaknesses.

3.4 Experimental results and discussion

In this section, we first present the dataset, which is used to train and evaluate the crop recommendation model. This dataset contains important agricultural information such as temperature, humidity, pH level, nutrient content in the soil, and rainfall, along with the appropriate crop type for each case. After introducing the data, we conduct a series of experiments aimed at identifying the best combination of hyperparameters for the model.

3.4.1 Datasets used

The choice of dataset plays a pivotal role in the success of any agricultural recommendation system. For our crop recommendation model, we selected a dataset from Kaggle that provides essential agricultural data such as temperature, humidity, pH levels, nutrient content in soil, and rainfall, along with the recommended crop type for each set of conditions. This dataset includes a large number of samples, reflecting various agricultural environments. By utilizing this dataset, we aim to derive insights that help us determine the most suitable model for recommending crops based on soil and environmental factors, see Figure 3.2.

N	P	K	temperature	humidity	ph	rainfall	soil_moisture	soil_type	sunlight_exposure	...	irrigation_frequency	crop_density	pest_pressure	fertilizer_usag
90	42	43	20.879744	82.002744	6.502985	202.935536	29.446064	2	8.677355	...	4	11.743910	57.607308	188.19495
85	58	41	21.770462	80.319644	7.038096	226.655537	12.851183	3	5.754288	...	4	16.797101	74.736879	70.96362
60	55	44	23.004459	82.320763	7.840207	263.964248	29.363913	2	9.875230	...	1	12.654395	1.034478	191.97607
74	35	40	26.491096	80.158363	6.980401	242.864034	26.207732	3	8.023685	...	1	10.864360	24.091888	55.76138
78	42	42	20.130175	81.604873	7.628473	262.717340	28.236236	2	8.120512	...	3	13.852910	38.811481	185.25970

Figure 3.2: Sample data from the crop recommendation dataset.

3.4.2 Hyperparameter tuning and models evaluation

In this section, we present the experimental results obtained through the application of various machine learning and deep learning models, including Logistic Regression, Support Vector Machine (SVM), Decision Tree, Naïve Bayes, LSTM, and GRU. To ensure optimal performance, a thorough hyperparameter tuning process was conducted for each model. The key hyperparameters explored include learning rate, batch size, dropout rate, and the number of epochs in the case of deep learning models, while parameters such as regularization strength (C) for Logistic Regression and kernel type for SVM were also investigated. These experiments aimed to identify the most effective configuration for each algorithm when applied to crop recommendation tasks. The evaluation was performed using consistent preprocessing and training methodologies, allowing us to fairly compare model performance and determine the hyperparameter settings that lead to the best predictive accuracy.

To optimize the performance of the applied models, we conducted hyperparameter tuning for both traditional machine learning and deep learning algorithms. For Logistic Regression and SVM, we experimented with parameters such as the regularization strength (C), solver types, and kernel functions. In the Decision Tree model, parameters like maximum depth, minimum samples split, and the criterion (Gini or entropy) were tuned. Although Naïve Bayes requires minimal tuning, we adjusted smoothing parameters to evaluate their impact. For the deep learning models, LSTM and GRU, we focused on optimizing batch size, number of epochs, learning rate, and dropout rate. This tuning process was essential to identify the best parameter configurations that yield the highest accuracy and generalization ability in crop recommendation tasks.

3.4.2.1 Naïve bayes classifier

In Gaussian Naïve Bayes, the varsmoothing parameter plays a crucial role in ensuring numerical stability during probability calculations. This parameter adds a small constant to the variance of each feature, preventing division by zero especially important when features exhibit very low variance. In our experiments, we evaluated the model using various input combinations like (-1, -6, 10), (-2, -6, 10), (-1, -2, 10), and (-15, -1, 15). These tests demonstrated consistent prediction results, highlighting the model's stability and robustness.

Table 3.1: Hyperparameter Tuning for Naïve Bayes Classifier

	Var Smoothing Range	**Train Acc**	**Val Acc**	**Test Acc**
Range	{-15,-9.5,-8.5,-6,-2, -1,10,15,20,25}	/	/	/
Best Value	$(-15, -1, 15)$	99.69%	98.86%	99.54%

The performance metrics indicated strong results across all combinations: training accuracies were 99.39%, 99.62%, 99.39%, and 99.69%, validation accuracies reached 98.63%, 98.86%, 98.63%, and 98.86%, and test accuracies were 99.09%, 99.54%, 98.86%, and 99.54%. Among these, the input combination (-15, -1, 15) achieved the highest training accuracy (99.69%) and tied for the highest validation and test accuracy (98.86% and 99.54%), demonstrating the most robust performance. These findings confirm that careful adjustment of the var smoothing parameter enhances the model's generalization ability, making it highly effective for recommending crops under diverse soil and environmental conditions.

3.4.2.2 Decision tree classifier

For the Decision Tree classifier, several rounds of hyperparameter tuning were carried out to identify the best combination of parameters that enhance the model's ability to recommend crops accurately. The parameters explored included the splitting criterion (**gini** or **entropy**), the maximum depth of the tree, the minimum number of samples required to split a node, and the minimum number of samples required at a leaf node. Various configurations were tested systematically across different ranges, including deeper trees and tighter split thresholds to understand their effect on model generalization.

Among the various configurations tested, a model using the Gini criterion with a maximum depth of 25, minimum samples split of 3, and minimum samples leaf of 1 achieved strong results, with a training accuracy of 99.92%, validation accuracy of 97.73%, and test accuracy of 97.95%. While this configuration showed excellent performance, it was slightly outperformed in terms of generalization by another setup using the entropy criterion with a maximum depth of 15, min samples split of 5, and min samples leaf of 1, which achieved a training accuracy of 100%, validation accuracy of 97.27%, and the highest test accuracy of 99.55

Other configurations were also explored to assess the trade off between model complexity and generalization. For instance, a Gini-based model with a depth of 15, split of 10, and leaf of 4 resulted in lower performance (99.02% train, 97.50% val, 97.50% test), showing that overly conservative splits may hinder generalization. Two trials of a Gini model with depth 12, split 5, and leaf 1 produced consistent results 99.70% training accuracy, 97.95% and 97.73% validation accuracies, and 98.64% test accuracy suggesting that shallower trees can still deliver competitive performance with appropriate tuning.

Overall, despite the strong performance of Gini-based models across various depths, the entropy-based configuration with a depth of 15, split of 5, and leaf of 1 delivered the highest test accuracy, demonstrating superior generalization capability. This highlights the importance of fine tuning hyperparameters and comparing multiple configurations to identify the most effective decision tree model.

Table 3.2: Hyperparameter Tuning for Decision Tree Classifier

	Criterion	**Max Depth**	**Min Split**	**Min Leaf**	**Train Acc**	**Val Acc**	**Test Acc**
Range	{gini, entropy}	{3,5,10,15, 21,25,31,35}	{1,2,3,5, 10,12,20}	{1,2, 4,5,6}	/	/	/
Best Value	entropy	15	5	1	100%	97.27%	99.55%

3.4.2.3 SVM classifier

For the Support Vector Machine (SVM) classifier, several rounds of hyperparameter tuning were conducted to determine the optimal configuration for accurate crop recommendation. The key parameters evaluated included the regularization parameter C, the kernel type, and the gamma value (γ) for non-linear kernels. A wide range of values was tested to explore how these hyperparameters influence the model's margin, decision boundaries, and generalization ability.

The regularization parameter C was tested with values such as 0.01, 0.1, 1, 10, and 100 to observe the trade off between margin maximization and classification errors. Various kernel types including linear, RBF, polynomial, and sigmoid were compared, with linear and RBF consistently outperforming the others in terms of both validation and test accuracy. For the RBF, polynomial, and sigmoid kernels, several γ values were evaluated, including `scale`, `auto`, and fixed values such as 0.01, 0.1, and 1.

The performance metrics indicated strong results across multiple configurations: training accuracies were 99.24%, 99.24%, 100%, 99.55%, and 100%; validation accuracies were 94.77%, 94.77%, 94.77%, 94.32%, and 76.14%; and test accuracies reached 94.77%, 94.77%, 95.23%, 95%, and 78.41%. Among these, the configuration using the RBF kernel with $C = 100$ and $\gamma = 0.01$ achieved the highest test accuracy of 95.23%, along with a perfect training score (100%) and a validation accuracy of 94.77%, indicating strong generalization capability.

Another competitive configuration was the RBF kernel with $C = 10$ and $\gamma = 0.01$, which achieved a training accuracy of 99.55%, validation accuracy of 94.32%, and test accuracy of 95%, reflecting slightly less generalization than the top performing setup. The linear kernel with $C = 1$ and $\gamma =$ `scale` showed consistent and stable performance with a training accuracy of 99.24%, and both validation and test accuracies of 94.77%, making it a reliable and balanced choice.

In contrast, the polynomial kernel with $C = 10$, degree $= 3$, and $\gamma =$ `auto` resulted in the poorest generalization, with a perfect training accuracy (100%) but significantly lower validation (76.14%) and test accuracy (78.41%).

Overall, the best-performing and most generalizable SVM configuration in this study was the RBF kernel with $C = 100$ and $\gamma = 0.01$, highlighting the importance of kernel selection and regularization strength in optimizing SVM for crop recommendation tasks under diverse input conditions.

3.4.2.4 Logistic regression classifier

For the Logistic Regression classifier, a series of hyperparameter tuning experiments were conducted to identify the most effective parameter combinations for accurate crop recommendation. The primary parameters tested included the regularization strength (C), the type of regularization penalty (l1 or l2), and the optimization solver (liblinear,

Table 3.3: Hyperparameter Tuning for SVM Classifier

	C	Gamma	kernel	Train Acc	Val Acc	Test Acc
Range	{0.01,0.1, 1,2,10,100}	{scale,auto, 0.01, 0.1, 1}	{linear,rbf, poly,sigmoid}	/	/	/
Best Value	100	0.01	rbf	100%	94.77%	95.23%

lbfgs, or saga). The inverse regularization strength parameter C was varied across a wide range from 0.001 to 100.0 to assess the impact of both strong and weak regularization on the model's performance. The solvers were selected based on compatibility with the chosen penalty type.

Among the various configurations tested, the setup using the liblinear solver with C = 0.5 and penalty = l1 yielded the best overall performance, achieving a training accuracy of 96.44%, a validation accuracy of 93.64%, and the highest test accuracy of 93.18%. This configuration demonstrated excellent generalization, likely due to the effectiveness of l1 regularization in promoting sparsity and reducing overfitting.

Other configurations were also examined to explore the trade-offs in model complexity and generalization. For example, a model using liblinear with C = 0.5 and penalty = l2 produced lower results, with a training accuracy of 92.05%, validation accuracy of 83.64%, and test accuracy of 85.45%. Increasing the C value to 1.0 with l2 penalty modestly improved test accuracy to 88.41%, but still did not match the performance of the l1 configuration.

Additionally, the lbfgs solver with C = 0.5 and penalty = l2 performed competitively in multiclass settings. It achieved a training accuracy of 97.95%, validation accuracy of 91.59%, and test accuracy of 91.82%, showing strong overall performance.

Table 3.4: Hyperparameter Tuning for Logistic Regression Classifier

	solver	C	penalty	max_iter	Train Acc	Val Acc	Test Acc
Range	{liblinear, lbfgs,saga}	{0.5,1}	{l1,l2}	{200,500}	/	/	/
Best Value	liblinear	0.5	l1	200	96.44%	93.64%	93.18%

In conclusion, while several logistic regression configurations performed well, the best generalization and highest test accuracy were achieved using the liblinear solver with l1 penalty and C = 0.5, demonstrating the importance of regularization type and solver choice in optimizing Logistic Regression for crop recommendation tasks.

3.4.2.5 Long Short-Term Memory

For the LSTM-based crop recommendation model, multiple rounds of hyperparameter tuning were conducted to optimize its predictive performance. The key parameters examined included the number of LSTM units, dropout rates for regularization, batch size, and the number of training epochs. These configurations were designed to explore how deeper architectures, lower dropout, and extended training periods affect the model's ability to generalize effectively.

Among the different configurations tested, the model with 128 and 64 LSTM units, dropout rate of 0.2, batch size of 32, and trained for 50 epochs yielded the best overall results. This setup achieved 100% training accuracy, and the highest validation and test accuracy of 95.91%. A similar configuration, using a dropout rate of 0.1 and batch size of 16, also reached perfect training accuracy and maintained strong generalization with 95.91% validation and 95.00

Other configurations involving 64 and 32 units generally showed slightly lower performance, particularly with smaller batch sizes or fewer epochs. For instance, using a batch size of 16 and 20 epochs achieved a training accuracy of 98.33%, with test accuracy dropping to 93.41%. Additionally, models trained with larger batch sizes, such as 64, tended to experience marginal drops in performance unless paired with lower dropout rates to reduce overfitting effects.

One notable configuration using 64 and 64 LSTM units with dropout 0.1 achieved a balanced result of 95.68% for both validation and test accuracy, reflecting the potential of shallow yet well-regularized models. Another configuration using 128 and 32 units, dropout 0.1/0.2, and batch size of 64 also performed competitively, with test accuracy of 94.55%.

Overall, the best performing LSTM configuration included 128 and 64 LSTM units, dropout rate of 0.2, batch size of 32, and 50 training epochs. This model not only achieved perfect training accuracy but also demonstrated superior generalization with 95.91% accuracy on both validation and test sets. These findings underscore the importance of selecting the right depth, regularization, and training duration in LSTM networks to achieve optimal results in crop recommendation systems.

Table 3.5: LSTM best accuracy results

	Units	**Dropout (1/2)**	**Batch Size**	**Epochs**	**Train Acc**	**Val Acc**	**Test Acc**
Range	{32,64,128}	{0.1,0.2}	{16,32,64}	{20,50}	/	/	/
Best Value	128 / 64	0.2 / 0.2	32	50	100.00%	95.91%	95.91%

3.4.2.6 Gated recurrent units

For the GRU-based crop recommendation model, multiple rounds of hyperparameter tuning were conducted to optimize its predictive performance. The key parameters examined included the number of GRU units, dropout rates for regularization, batch size, and the number of training epochs. These configurations were selected to investigate the effect of deeper architectures, dropout variations, and extended training periods on the model's generalization ability.

Among the different configurations tested, the model with 128 and 64 GRU units, dropout rate of 0.2, batch size of 32, and trained for 50 epochs achieved the highest overall results. This setup delivered 100% training accuracy, and strong validation and test accuracy of 96.82% and 96.14%, respectively. A nearly identical configuration using batch size of 16 also reached 100% training accuracy, with 96.59% validation accuracy and 96.36% test accuracy.

Configurations with 64 and 32 GRU units generally performed slightly lower than the deeper models, particularly when using smaller batch sizes or fewer epochs. For example, training with batch size of 16 and 20 epochs using GRU(64, 32) achieved a training accuracy of 98.79%, with validation and test accuracy around 94.77% and 95.00%, respectively. Increasing the batch size to 64 without adjusting dropout led to slight performance drops, but applying lower dropout rates (0.1) helped maintain strong accuracy.

Other competitive setups included GRU(128, 32) with dropout 0.1, batch size of 16, and 50 epochs, which attained 99.85% training accuracy, 96.82% validation accuracy, and 96.14% test accuracy closely matching the best configuration. Another variation using GRU(64, 32), dropout 0.2, and batch size of 64 also performed well, achieving 99.24% training, 96.59% validation, and 96.14% test accuracy.

Overall, the best-performing GRU configuration consisted of 128 and 64 GRU units, trained over 50 epochs, using a dropout rate of 0.2 and a batch size of 32. This model demonstrated robust generalization capability with 100% training accuracy and 96.82% and 96.14% accuracy on validation and test sets, respectively. These findings highlight the effectiveness of deeper GRU architectures and balanced regularization for enhancing performance in crop recommendation tasks.

Table 3.6: GRU Best Accuracy Results

	Units	**Dropout (1/2)**	**Batch Size**	**Epochs**	**Train Acc**	**Val Acc**	**Test Acc**
Range	{32,64,128}	{0.1,0.2}	{16,32,64}	{20,50}	/	/	/
Best Value	128 / 32	0.2 / 0.2	16	50	100.00%	96.59%	96.36%

3.5 Comparison of results

Through this experimental study, we aim to identify the most suitable model architecture that achieves optimal performance for our system. In the following section, we conduct a comparative analysis of different models trained using the best combinations of hyperparameters. The performance of each model was evaluated based on training, validation, and testing accuracy, as summarized in the table below:

Table 3.7: Accuracy of our models

Model	**Train Acc**	**Val Acc**	**Test Acc**
NB	99.69%	98.86%	99.54%
DT	**100%**	**97.27%**	**99.55%**
SVM	100%	94.77%	95.23%
LR	96.44%	93.64%	93.18%
LSTM	100%	95.91%	95.91%
GRU	100%	96.59%	96.36%

The Decision Tree (DT) model clearly stands out as the most effective, achieving the highest test accuracy of 99.55%, alongside strong validation accuracy (97.27%) and perfect training accuracy, demonstrating excellent generalization to unseen data and offering an excellent balance between accuracy and model stability, making it the most suitable architecture for our system.

3.6 Comparison with related work

To assess the robustness and efficiency of our system, we conducted a comparative analysis with a recent state of the art study that also used the Smart Farming 2024 (SF24) dataset. The selected reference is the work by (Aldhahri et al. 2025), which introduced an advanced deep learning ensemble architecture known as ResXceNet-HBA. Their model integrated ResNet blocks, Xception modules, and optimization via the Honey Badger Algorithm, achieving a test accuracy of 98.5%, precision of 98.2%, recall of 98.7%, and F1-score of 98.4% on the SF24 dataset.

In contrast, our system utilized several classical machine learning models, among which the Decision Tree (DT) model demonstrated the most outstanding performance. Prior to testing, the DT model achieved a perfect training accuracy of 100% and a validation accuracy of 97.72%, reflecting its strong capacity for generalization. Upon evaluation on the test set, the model attained a test accuracy of 99.32%, along with a precision of 99.36%, recall of 99.31%, and an F1-score of 99.32%, thus outperforming more complex deep learning architectures in terms of both accuracy and consistency.

This comparison highlights that while deep learning models such as ResXceNet-HBA offer scalability, robustness, and adaptability across various environmental conditions, simpler models like Decision Trees when combined with effective preprocessing and well structured datasets can achieve even higher predictive performance. Our findings underscore the practicality and efficiency of classical approaches in smart farming applications, especially when computational simplicity and interpretability are also desired.

Table 3.8: Performance Comparison on the SF24 Dataset

Model	Test Accuracy	Precision	Recall	F1 Score
Decision Tree (Ours)	**99.32%**	**99.36%**	**99.31%**	**99.32%**
ResXceNet-HBA (Aldhahri et al. 2025)	98.5%	98.2%	98.7%	98.4%

3.7 Conclusion

In this chapter, we presented a comprehensive framework for an intelligent system based on artificial intelligence techniques to analyze soil properties and provide optimal recommendations regarding the most suitable crop type for cultivation. We reviewed the system design stages, starting from soil data collection and preprocessing, to developing and testing machine learning models used for data analysis. Through a series of experiments, we fine-tuned the models' hyperparameters to achieve the highest prediction accuracy. The system demonstrated significant effectiveness in delivering precise recommendations that support agricultural decision making. Our results highlight the potential of artificial intelligence to enhance agricultural productivity and improve resource efficiency, opening wide prospects for future applications in smart farming.

Conclusion

In conclusion, this work highlights the growing importance of integrating artificial intelligence technologies into the agricultural sector, particularly in the analysis of soil properties and the generation of precise recommendations to improve productivity and ensure resource sustainability. The results of this study demonstrate that the combination of machine learning and deep learning algorithms with agricultural data enables a deeper understanding of soil composition and its impact on crop performance.

By developing an intelligent model capable of analyzing soil and recommending optimal agricultural practices, we have taken a significant step toward data driven precision farming an essential approach for modern, efficient, and sustainable agriculture.

The outcomes obtained confirm the effectiveness of the proposed models in accurately predicting the suitable agricultural inputs based on soil characteristics and environmental conditions. This represents a promising direction for expanding such systems on a broader scale.

This research also opens the door to several future prospects, such as:

Data Expansion: Incorporating larger and more diverse datasets including satellite imagery, real time climate data, and geospatial information will improve model robustness and generalization.

Multimodal Integration: Combining soil data with plant growth patterns, pest activity, and weather forecasts can lead to holistic farming solutions.

Scalability: Deploying the system as a mobile or cloud-based application could empower farmers in remote or underserved areas to access AI-driven insights easily.

Ethical and Environmental Considerations: As AI systems are adopted more widely in agriculture, issues such as data privacy, ecological impact, and digital inequality must also be addressed.

Model Transparency: Future work should aim at improving model interpretability so that recommendations can be better understood and trusted by end users.

In summary, this work provides a practical example of how artificial intelligence can be used to revolutionize agriculture. By transforming raw data into actionable insights, AI becomes a vital tool not only for boosting crop yield and farm profitability but also for addressing some of the world's most pressing challenges: food insecurity, climate change, and resource scarcity. The findings presented herein serve as a strong foundation for continued research and real world implementation of AI powered solutions in agriculture.

Bibliographie

Ahamed, K Ishthaq and Shaheda Akthar (2016). “A study on neural network architectures”. In: *Comput. Eng. Intell. Syst* 7, pp. 1–7.

Ahmad, Noman et al. (2023). “Soybean (Glycine max) Cropland Suitability Analysis in Subtropical Desert Climate through GIS-Based Multicriteria Analysis and Sentinel-2 Multispectral Imaging”. In: *Land* 12.11, p. 2034.

Akinde, Bamikole Peter et al. (2020). “Selected physical and chemical properties of soil under different agricultural land-use types in Ile-Ife, Nigeria”. In: *Heliyon* 6.9.

Aldhahri, Eman Ali et al. (2025). “Smart Farming: Enhancing Urban Agriculture through Predictive Analytics and Resource Optimization”. In: *IEEE Access*.

Ali, Sayed Mazhar, Bhagwan Das, and Dileep Kumar (2021). “Machine learning based crop recommendation system for local farmers of pakistan”. In: *Revista Geintec-Gestao Inovacao E Tecnologias* 11.4, pp. 5735–5746.

Awais, Muhammad et al. (2023). “AI and machine learning for soil analysis: an assessment of sustainable agricultural practices”. In: *Bioresources and Bioprocessing* 10.1, p. 90.

Bai, Zhanguo et al. (2018). “Effects of agricultural management practices on soil quality: A review of long-term experiments for Europe and China”. In: *Agriculture, ecosystems & environment* 265, pp. 1–7.

Ball, Bruce C et al. (2017). “Visual soil evaluation: a summary of some applications and potential developments for agriculture”. In: *Soil and Tillage Research* 173, pp. 114–124.

Bengio, Yoshua et al. (2009). “Learning deep architectures for AI”. In: *Foundations and trends® in Machine Learning* 2.1, pp. 1–127.

Björklund, Anton, Jarmo Mäkelä, and Kai Puolamäki (2023). “SLISEMAP: Supervised dimensionality reduction through local explanations”. In: *Machine Learning* 112.1, pp. 1–43.

Bonds, Darcy et al. (2022). “Can soil smell tell us something about soil practices?” In: *2022 ASABE Annual International Meeting*. American Society of Agricultural and Biological Engineers, p. 1.

Brady, Nyle C, Ray R Weil, and Ray R Weil (2008). *The nature and properties of soils*. Vol. 13. Prentice Hall Upper Saddle River, NJ.

Brevik, Eric C et al. (2015). “The interdisciplinary nature of SOIL”. In: *Soil* 1.1, pp. 117–129.

Cao, Juan et al. (2021). “Integrating multi-source data for rice yield prediction across China using machine learning and deep learning approaches”. In: *Agricultural and Forest Meteorology* 297, p. 108275.

Dean, Jeffrey et al. (2012). “Large scale distributed deep networks”. In: *Advances in neural information processing systems* 25.

Del Sole, Alessandro and Del Sole (2019). *Visual Studio Code Distilled*. Springer.

Dey, Biplob, Jannatul Ferdous, and Romel Ahmed (2024). “Machine learning based recommendation of agricultural and horticultural crop farming in India under the regime of NPK, soil pH and three climatic variables”. In: *Heliyon* 10.3.

DiMarzio, Jerome (2016). *Beginning android programming with android studio*. John Wiley & Sons.

Gao, Hongyang, Lei Cai, and Shuiwang Ji (2020). “Adaptive convolutional relus”. In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 34. 04, pp. 3914–3921.

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville (2016). “Regularization for deep learning”. In: *Deep learning*, pp. 216–261.

Grishin, Igor and Rena Timirgaleeva (2020). “Remote sensing: The method of GIS application for monitoring the state of soils”. In: *E3S Web of Conferences*. Vol. 175. EDP Sciences, p. 06009.

Hardy, Jean et al. (2018). “User acceptance of location-tracking technologies in health research: implications for study design and data quality”. In: *Journal of biomedical informatics* 79, pp. 7–19.

Harris, Robert H et al. (2013). “Can nitrogen fertiliser and nitrification inhibitor management influence N 2 O losses from high rainfall cropping systems in South Eastern Australia?” In: *Nutrient cycling in agroecosystems* 95, pp. 269–285.

Hordri, Nur Farhana, Siti Sophiayati Yuhaniz, and Siti Mariyam Shamsuddin (2016). “Deep learning and its applications: A review”. In: *Conference on postgraduate annual research on informatics seminar*, pp. 1–5.

Hutri, Hugo (2023). “Comparison of React Native and Expo”. In.

Iqbal, Ahmed and Shabib Aftab (2019). “A feed-forward and pattern recognition ANN model for network intrusion detection”. In: *International Journal of Computer Network and Information Security* 14.4, p. 19.

Islam, Saidul et al. (2024). “A comprehensive survey on applications of transformers for deep learning tasks”. In: *Expert Systems with Applications* 241, p. 122666.

Janiesch, Christian, Patrick Zschech, and Kai Heinrich (2021). “Machine learning and deep learning”. In: *Electronic markets* 31.3, pp. 685–695.

Jones, Jr (2018). *Soil analysis handbook of reference methods*. CRC press.

Kaelbling, Leslie Pack, Michael L Littman, and Andrew W Moore (1996). “Reinforcement learning: A survey”. In: *Journal of artificial intelligence research* 4, pp. 237–285.

Kim, Ha Young and Chang Hyun Won (2018). “Forecasting the volatility of stock price index: A hybrid model integrating LSTM with multiple GARCH-type models”. In: *Expert Systems with Applications* 103, pp. 25–37.

Kreuzberger, Dominik, Niklas Kühl, and Sebastian Hirschl (2023). “Machine learning operations (mlops): Overview, definition, and architecture”. In: *IEEE access* 11, pp. 31866–31879.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton (2012). “Imagenet classification with deep convolutional neural networks”. In: *Advances in neural information processing systems* 25.

Lal, Rattan (2011). “Soil health and climate change: an overview”. In: *Soil health and climate change*, pp. 3–24.

Linares-Vásquez, Mario et al. (2014). “How do api changes trigger stack overflow discussions? a study on the android sdk”. In: *proceedings of the 22nd International Conference on Program Comprehension*, pp. 83–94.

McCarthy, John and Edward A Feigenbaum (1990). “In memoriam: Arthur samuel: Pioneer in machine learning”. In: *Ai Magazine* 11.3, pp. 10–10.

McKinney, Wes and PD Team (2015). “Pandas-Powerful python data analysis toolkit”. In: *Pandas—Powerful Python Data Analysis Toolkit* 1625.

McLean, EO (1982). “Soil pH and lime requirement”. In: *Methods of soil analysis: Part 2 Chemical and microbiological properties* 9, pp. 199–224.

Mey, Alexander and Marco Loog (2022). “Improved generalization in semi-supervised learning: A survey of theoretical results”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45.4, pp. 4747–4767.

Naeem, Samreen et al. (2023). “An unsupervised machine learning algorithms: Comprehensive review”. In: *International Journal of Computing and Digital Systems*.

Nocita, Marco et al. (2015). “Soil spectroscopy: an alternative to wet chemistry for soil monitoring”. In: *Advances in agronomy* 132, pp. 139–159.

Nofal, Samer, Katie Atkinson, and Paul E Dunne (2014). “Algorithms for argumentation semantics: labeling attacks as a generalization of labeling arguments”. In: *Journal of Artificial Intelligence Research* 49, pp. 635–668.

Osisanwo, FY et al. (2017). “Supervised machine learning algorithms: classification and comparison”. In: *International Journal of Computer Trends and Technology (IJCTT)* 48.3, pp. 128–138.

Pal, Abhishesh et al. (2024). “Portable sensors in precision agriculture: Assessing advances and challenges in soil nutrient determination”. In: *TrAC Trends in Analytical Chemistry*, p. 117981.

Pokhariyal, Shweta, NR Patel, and Ajit Govind (2023). “Machine Learning-Driven Remote Sensing Applications for Agriculture in India—A Systematic Review”. In: *Agronomy* 13.9, p. 2302.

Prasetya, Rangga Prasetya Rangga and Bambang Sujatmiko (2025). “Comparison Of Web Load Speed Between Babel Transpiler and SWC On Website-Based Applications”. In: *Journal of Emerging Information System and Business Intelligence (JEISBI)* 6.1, pp. 40–48.

Presa Käld, Marcus and Oskar Svensson (2021). *React Native and native application development: A comparative study based on features & functionality when measuring performance in hybrid and native applications.*

Purwono, Purwono et al. (2022). “Understanding of convolutional neural network (cnn): A review”. In: *International Journal of Robotics and Control Systems* 2.4, pp. 739–748.

Resig, John (2007). *Pro JavaScript Techniques*. Apress.

Sah, Shagan (2020). “Machine learning: a review of learning types”. In: *Preprints*.

Santanello Jr, Joseph A et al. (2007). “Using remotely-sensed estimates of soil moisture to infer soil texture and hydraulic properties across a semi-arid watershed”. In: *Remote Sensing of Environment* 110.1, pp. 79–97.

Santos, Celso Augusto Guimarães et al. (2025). “Estimating water levels through smartphone-imaged gauges: a comparative analysis of ANN, DL, and CNN models”. In: *Water Resources Management* 39.4, pp. 1639–1654.

Schmidhuber, Jürgen (2015). “Deep learning in neural networks: An overview”. In: *Neural networks* 61, pp. 85–117.

Sharma, Ritu, Kavya Sharma, and Apurva Khanna (2020). “Study of supervised learning and unsupervised learning”. In: *International Journal for Research in Applied Science and Engineering Technology* 8.6, pp. 588–593.

Sherstinsky, Alex (2020). “Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network”. In: *Physica D: Nonlinear Phenomena* 404, p. 132306.

Soulez, Marie (2018). “Questions juridiques au sujet de l’intelligence artificielle”. In: *Annales des Mines-Enjeux numériques* 1.1, pp. 81–85.

Sudiartha, IKG et al. (2020). “Data structure comparison between mysql relational database and firebase database nosql on mobile based tourist tracking application”. In: *Journal of Physics: Conference Series*. Vol. 1569. 3. IOP Publishing, p. 032092.

Uddin, Shahadat et al. (2019). “Comparing different supervised machine learning algorithms for disease prediction”. In: *BMC medical informatics and decision making* 19.1, pp. 1–16.

Wadoux, Alexandre MJ-C, Budiman Minasny, and Alex B McBratney (2020). “Machine learning for digital soil mapping: Applications, challenges and suggested solutions”. In: *Earth-Science Reviews* 210, p. 103359.

Weinert, Alexander (2014). *The Turing Test*.

Zhang, Dongdong et al. (2020). “Combining structured and unstructured data for predictive models: a deep learning approach”. In: *BMC medical informatics and decision making* 20, pp. 1–11.

Zhumagambetov, Rustam et al. (2021). “Transmol: repurposing a language model for molecular generation”. In: *RSC advances* 11.42, pp. 25921–25932.

Webographie

al., Bashar et (2018). *Multilayered Artificial Neural Network*. Consulté en mai 2025. URL: `https://www.researchgate.net/figure/Multilayered-artificial-neural-network-The-Artificial-Neural-Network-characteristics-are_fig3_326462095`.

al., Dener et (2022). *Depiction of SVM Model*. Consulté en mai 2025. URL: `https://www.researchgate.net/figure/Depiction-of-SVM-Model-Dener-et-al-2022_fig2_384434138`.

Alibaba (n.d.). *Cheap Soil Nutrients Testing Sensor Equipment*. Accessed on June 1, 2025. URL: `https://arabic.alibaba.com/product-detail/Cheap-soil-nutrients-testing-sensor-equipment-1600232566751.html`.

Anaconda, Inc. (2024). *Anaconda AI Navigator*. Consulté le 12 juin 2025. URL: `https://www.anaconda.com/docs/tools/ai-navigator/main`.

AnalytixLabs (2023a). *Classification in Machine Learning: Types and Methodologies*. Consulté en mai 2025. URL: `https://www.analytixlabs.co.in/blog/classification-in-machine-learning/`.

— (2023b). *Convolutional Neural Network: Layers, Types, & More*. Consulté le 1 juin 2025. URL: `https://www.analytixlabs.co.in/blog/convolutional-neural-network/`.

Ashwin (2023). *Feed Forward Neural Networks – Intuition on Forward Propagation*. Consulté en mai 2025. URL: `https://www.analyticsvidhya.com/blog/2021/10/feed-forward-neural-networks-intuition-on-forward-propagation/`.

Bhatnagar, Daksh (2023). *Building a Basic Neural Network from Scratch: A Step-by-Step Guide*. Consulté le 1 juin 2025. URL: `https://bhatnagar91.medium.com/building-a-basic-neural-network-from-scratch-a-step-by-step-guide-7a6f97979ddd`.

Bicky (2025). *( )* . Consulté en mai 2025. URL: `https://vocus.cc/article/67892472fd89780001af7e59`.

Czar (2014). *K-Means Clustering Tutorial*. Consulté le 1 juin 2025. URL: `https://rpubs.com/cyobero/k-means`.

Databricks (2025). *Jupyter Notebook*. Consulté le 12 juin 2025. URL: `https://www.databricks.com/glossary/jupyter-notebook`.

Engineer, Dataset (2024). *Smart Farming Data 2024 (SF24)*. Consulté en juin 2025. URL: `https://www.kaggle.com/datasets/datasetengineer/smart-farming-data-2024-sf24`.

FITA Academy (2023). *Top 10 Machine Learning Algorithms For Novices: Guided and More*. Consulté en mai 2025. URL: `https://www.fita.in/top-ten-machine-learning-algorithms-for-novices-guided-and-more/`.

guoyee94 (2024). *[TIL] [KT AIVLE School] DX 8 1 . - (1). (PCA, t-SNE) (K-Means)*. Consulté en mai 2025. URL: `https://guoyee94.tistory.com/m/50`.

Hayatok (2025). *Geology: Fundamentals and Branches*. Accessed: June 1, 2025. URL: https://hyatok.com/%D8%B9%D9%84%D9%85_%D8%A7%D9%84%D8%AC%D9%8A%D9%88%D9%84%D9%88%D8%AC%D9%8A%D8%A7:_%D8%A3%D8%B3%D8%A7%D8%B3%D9%8A%D8%A7%D8%AA%D9%87_%D9%88%D9%81%D8%B1%D9%88%D8%B9%D9%87.

Ingle, Atharva (2021). *Crop Recommendation Dataset*. Consulté en mai 2025. URL: https://www.kaggle.com/datasets/atharvaingle/crop-recommendation-dataset.

Jadhav, Shridhan (2024). *Artificial Intelligence*. Consulté en mai 2025. URL: https://medium.com/@shridhanj5/artificial-intelligence-710cecd764b0.

Kato, Shoya (2021). *DP -* . Consulté le 1 juin 2025. URL: https://www.jsae.or.jp/files_publish/page/528/4-7.pdf.

Mahmudiono, Trias et al. (2022). *A simple diagram of neural networks*. Consulté en mai 2025. URL: https://www.researchgate.net/figure/A-simple-diagram-of-neural-networks_fig3_359996805.

Majdoubi, Oumaima, Achraf Benba, and Ahmed Hammouch (2023). *Comprehensive Machine Learning and Deep Learning Approaches for Parkinson's Disease Classification and Severity Assessment*. Consulté en mai 2025. URL: https://www.researchgate.net/publication/376678940.

Modasiya, Kishan (2022). *Types of Machine Learning Algorithm You Should Know*. Consulté le 1 juin 2025. URL: https://kishanmodasiya.medium.com/types-of-machine-learning-algorithm-you-should-know-73c0e5fa8451.

Piddubna, Alina (2022). *8 Applications of GIS in Agriculture for Predictability & Sustainable Growth*. Accessed on June 1, 2025. URL: https://intellias.com/gis-in-agriculture/.

Polamuri, Saimadhu (2017). *How the Logistic Regression Model Works in Machine Learning*. Consulté en mai 2025. URL: https://opendatascience.com/how-the-logistic-regression-model-works-in-machine-learning/.

Sanchaari, Antarjaala (2024). *A Conversation with a Neural Network*. Consulté en mai 2025. URL: https://blog.stackademic.com/a-conversation-with-a-neural-network-e8678a73640c.

Scikit-learn Developers (2025). *Scikit-learn API Reference*. Consulté le 12 juin 2025. URL: https://scikit-learn.org/stable/api/sklearn.html.

Starmer, Josh (2018). *Linear Classification: Understanding the Fundamentals and Theory*. Consulté en mai 2025. URL: https://www.youtube.com/watch?v=Ojy03zDyXi8.

STEMpedia (2025). *Measure pH value of Soil*. Consulté le 1 juin 2025. URL: https://ai.thestempedia.com/docs/evive-iot-kit/how-to-measure-ph-value-of-soil/.

Student Projects (2025). *Soil Classification*. Consulté le 1 juin 2025. URL: https://studentprojects.in/civil-engineering/soil-mechanics/soil-classification/.

Sung Kim (2016). *Titre inconnu - YouTube Video*. Consulté le 1 juin 2025. URL: https://www.youtube.com/watch?v=-SHPG_KMUkQ.

TensorFlow (2025a). *TensorFlow: An end-to-end open source machine learning platform*. Consulté le 12 juin 2025. URL: https://www.tensorflow.org/?hl=fr.

— (2025b). *tf.keras.Layer | TensorFlow v2.0*. Consulté le 12 juin 2025. URL: https://www.tensorflow.org/api_docs/python/tf/keras/Layer.

— (2025c). *tf.keras.Model | TensorFlow v2.0*. Consulté le 12 juin 2025. URL: https://www.tensorflow.org/api_docs/python/tf/keras/Model.

— (2025d). *tf.keras.utils.to_categorical | TensorFlow v2.0*. Consulté le 12 juin 2025. URL: https://www.tensorflow.org/api_docs/python/tf/keras/utils/to_categorical.

TmleynCodes (2021). *Crop Recommendation Using Soil and Weather*. Consulté en mai 2025. URL: https://www.kaggle.com/datasets/tmleyncodes/crop-recommendation-using-soil-and-weather.

UBC Wiki (2025). *Soil Texture - Hand Texturing Method*. Consulté le 1 juin 2025. URL: https://wiki.ubc.ca/Soil_Texture_-_Hand_Texturing_Method.

Vaitheswaran, Sivashankari (2021). *Understanding Anomaly Detection*. Consulté le 1 juin 2025. URL: https://sivashankarivaitheswaran.medium.com/understanding-anomaly-detection-6c1e018db159.

Van Rossum, Guido (2001). *The History of Python*. Consulté le 12 juin 2025. URL: https://www.python.org/doc/essays/blurb/.

Wang, Lin (2024). *69 Classic Concepts in the 10-Year Journey of NLP — Part 1 (2014–2018)*. Consulté le 1 juin 2025. URL: https://blog.cubed.run/69-classic-concepts-in-the-10-year-journey-of-nlp-part-1-2014-2018-d0d34553df65.

X, Crystal (2023). *Build a Decision Tree Classifier with JAX*. Consulté en mai 2025. URL: https://python.plainenglish.io/build-a-decision-tree-classifier-with-jax-80c838f3aebc.

Appendix A

ANNEX: Development Tools and Implementation

A.1 Introduction

In this annex, we focus on the development and implementation of an AI-based soil analysis application designed to recommend the most suitable crops for agricultural lands, transforming the conceptual model presented earlier into a fully functional and practical system. The process involves selecting the appropriate tools and frameworks, importing the necessary libraries, preparing and splitting the dataset, and building a classification model whether based on traditional machine learning algorithms or deep learning techniques. This is followed by training the model and evaluating its performance on unseen test data to ensure accuracy, reliability, and the system's overall effectiveness in real-world agricultural scenarios.

A.2 Development tools

A.2.1 Artificial intelligence

A.2.1.1 Anaconda navigator

Anaconda AI Navigator is a platform that allows users to download and run Large Language Models (LLMs) directly on their local machines. It includes a built-in llama.cpp API server, making it ideal for developers who want to test AI models within their existing applications. With access to Anaconda's curated list of trusted models, users can evaluate and choose the best model for their needs. Running models locally also ensures full control over personal and proprietary data, removing reliance on external servers.(Anaconda, Inc. 2024)

Figure A.1: Anaconda logo

A.2.1.2 Jupyter

Jupyter Notebooks are interactive, web based tools widely used in data science and machine learning for tasks such as data exploration, cleaning, visualization, statistical modeling, and deep learning. They allow users to write and execute code in individual cells, combining programming, text, images, and visualizations in a single document. Powered by a front end interface and a back end kernel (usually Python), Jupyter makes it easy to test and explain code step by step. Notebooks are also flexible, supporting export to various formats like HTML, PDF, and Python scripts for easy sharing.(Databricks 2025)

Figure A.2: Jupyter logo

A.2.1.3 Development language

Python is an interpreted, object oriented, high level programming language with dynamic semantics. It features built in data structures, dynamic typing, and dynamic binding, which make it particularly attractive for rapid application development and as a scripting or glue language to connect existing components. Its clear and easy to learn syntax emphasizes readability and reduces program maintenance costs. Python is open source and belongs to the family of interpreted languages, allowing developers to focus on problem solving rather than dealing with complex syntax or compilation. It supports modules and packages that promote modularity and code reuse, and its extensive standard library is freely available for all major platforms. Python is dynamically typed, with optional type annotations for better code clarity. It is widely used in various domains such as web development, data analysis, artificial intelligence, neural networks, and scientific computing. The fast edit test debug cycle and powerful built in debugger increase developer productivity, while its flexibility allows it to be used in both simple and highly complex projects. Without a doubt, Python has become one of the most appreciated and in demand programming languages in the tech world.(Van Rossum, Guido 2001)

Figure A.3: Python logo

A.2.1.4 Used library

Pandas: A BSD-licensed, open-source library, it offers a powerful and versatile set of data structures, notably DataFrame and Series, specifically designed for efficient manipulation of structured data, time series, and labeled datasets within the Python environment. Its capabilities extend across the entire data workflow, making it an indispensable tool for tasks such as reading diverse data formats, performing essential data cleaning operations, transforming data into suitable structures, seamlessly merging disparate datasets, and conducting in depth analysis. This comprehensive functionality, combined with its high performance and user friendly interface, positions it as an ideal solution for data professionals and researchers working with complex data in Python.(McKinney et al. 2015)

Numpy: This statement describes NumPy, which serves as the foundational library for numerical computing in Python. It provides powerful N-dimensional array objects, along with a comprehensive suite of mathematical functions and routines for linear algebra. Its highly optimized array handling capabilities are so crucial that nearly all other scientific libraries in Python rely on NumPy for their underlying data structures and operations.

Sklearn model selection: The sklearn.model selection module provides essential tools for preparing and evaluating machine learning models. It includes functions like train test split, which quickly divides a dataset into training and testing subsets using simple parameters such as test size, shuffle, and stratify. It also offers GridSearchCV, a powerful utility that performs an exhaustive search over specified parameter values using cross-validation to determine the best model configuration. Additionally, it includes splitters like StratifiedKFold and other cross-validation tools to ensure robust model evaluation.(Scikit-learn Developers 2025)

Sklearn svm: The sklearn svm module contains various implementations of Support Vector Machines (SVMs), powerful algorithms highly effective for both classification and regression tasks. These models operate by identifying optimal separating hyperplanes within the feature space, thereby distinguishing between different classes or predicting continuous outcomes.(Scikit-learn Developers 2025)

Sklearn naive bayes: sklearn naive bayes provides a collection of probabilistic classifiers, including GaussianNB. These classifiers apply Bayes' theorem, making the assumption of feature independence to perform straightforward yet effective classification.(Scikit-learn Developers 2025)

Sklearn tree: ithin sklearn.tree, users can find decision tree algorithms. These algorithms model decision making processes in a tree like structure and are capable of performing both classification and regression tasks.(Scikit-learn Developers 2025)

Sklearn linear model: The sklearn linear model module houses various linear models, prominently featuring Logistic Regression. This model is primarily designed to predict discrete outcomes, making it a powerful classification algorithm. It is particularly well suited for both binary and multi-class classification problems.(Scikit-learn Developers 2025)

Sklearn metrics: The sklearn.metrics module provides a comprehensive suite of evaluation functions, including accuracy score, precision score, recall score, and f1 score, which are essential for thoroughly assessing and understanding a machine learning model's performance.(Scikit-learn Developers 2025)

Slearn preprocessing: The sklearn preprocessing module offers essential tools for data preparation, such as StandardScaler for feature scaling to achieve zero mean and unit variance, and LabelEncoder for transforming categorical variables into a suitable numerical format for machine learning algorithms.(Scikit-learn Developers 2025)

Tensorflow: TensorFlow, an open source framework from Google, is widely used for building and training deep learning models. It offers both high-level APIs like Keras for easy model construction and low-level APIs for fine grained control, catering to diverse development needs. Its flexibility and extensive capabilities make it a powerful tool for a broad spectrum of AI applications.(TensorFlow 2025a)

Tensorflow keras models: The tensorflow keras models module's Sequential class offers a convenient way to build neural networks. It enables developers to easily stack layers in a linear order. This simplifies model construction, especially for feedforward architectures.(TensorFlow 2025c)

Tensorflow keras layers: The tensorflow.keras.layers module includes essential neural network layers widely used in deep learning models. The Dense layer is a fully connected layer where each input neuron is connected to every output neuron, commonly used in feedforward neural networks. The LSTM (Long Short-Term Memory) layer is a type of recurrent neural network (RNN) that excels in processing and learning from sequential data by preserving long-term dependencies. The GRU (Gated Recurrent Unit) is a simplified version of LSTM that is computationally more efficient while still effective for sequence-based tasks. Lastly, the Dropout layer is a regularization technique that randomly drops neurons during training to help prevent overfitting and improve generalization of the model.(TensorFlow 2025b)

Tensorflow keras utils: The tensorflow.keras.utils module provides essential utility functions, such as to categorical. This function converts integer labels into a one-hot encoded format, which is crucial for preparing data for multi-class classification tasks in deep learning models. This ensures labels are in the necessary binary vector form for proper model training.(TensorFlow 2025d)

A.2.2 Mobile

A.2.2.1 Visual Studio Code

Visual Studio Code (VS Code) is a lightweight, open-source, and cross-platform code editor developed by Microsoft. It is designed to support a wide variety of programming languages and development scenarios, including web, mobile, and cloud applications. VS Code provides rich features such as syntax highlighting, code completion, integrated debugging, and a built-in terminal, all aimed at enhancing productivity and streamlining the development process.

It is more than a simple text editor it is a powerful, code focused development environment that supports the full application lifecycle with integrated Git version control and a vast library of extensions. These extensions enable customization of the coding environment with added language support, themes, linters, and collaboration tools like Live Share for real-time teamwork. Its versatility makes VS Code popular for both quick scripting and complex software projects on any operating system.(Del Sole et al. 2019)

Figure A.4: VS Code logo

A.2.2.2 Android studio

Android Studio is the official integrated development environment (IDE) for Android app development, created by Google. It is built on JetBrains' IntelliJ IDEA and offers a comprehensive suite of tools for building, testing, and debugging Android applications. Android Studio provides advanced features such as a smart code editor, visual layout editor, real-time profilers, and a Gradle-based build system.(DiMarzio 2016)

A key component of Android Studio is the Android Software Development Kit (SDK), which is bundled with the IDE. The SDK includes essential tools, libraries, and APIs required for developing Android apps. These include a code debugger, device emulator, and platform-specific tools that allow developers to write applications using Java, Kotlin, or C++, and optimize them for different Android devices and OS versions.(Linares-Vásquez et al. 2014) Android Studio, along with the Android SDK, delivers a full-featured environment for building Android apps across various device types.

Figure A.5: Android Studio logo

A.2.2.3 Expo

Expo is an open source platform built on top of React Native that simplifies mobile app development for both Android and iOS using JavaScript. It provides a comprehensive set of preconfigured tools and services that automate many routine tasks like building, updating, and submitting apps to app stores. By handling these processes, Expo allows developers to focus more on writing code and designing user experiences rather than managing complex configurations. Its development environment includes an easy to use CLI, over-the-air updates, and built-in integrations for common native features. Expo also boasts a supportive community and direct access to the development team for assistance. While it may have some limitations for apps requiring deep native customization, Expo is ideal for rapid development and prototyping. Overall, Expo accelerates the app creation process by offering a streamlined, developer friendly ecosystem. It is especially suited for small to medium projects that benefit from fast iteration and deployment.(Hutri 2023)

Figure A.6: Expo logo

A.2.2.4 Development language

JavaScript is a versatile, high-level programming language used to build dynamic and interactive experiences on the web. Originally introduced by Netscape in 1995, it has grown to support both front-end and back-end development through platforms like Node.js. As a foundational technology of the web, alongside HTML and CSS, JavaScript enhances how users interact with websites and applications. The language embraces multiple paradigms, including object-oriented, functional, and imperative styles. Its features such as dynamic typing, event-based execution, and first-class functions enable the creation of rich, complex web functionalities. Governed by the ECMAScript standard, JavaScript behaves consistently across different browsers. Its wide ecosystem features powerful frameworks like Vue.js, Angular, and React, which streamline app development. It also supports specialized applications like online experiments using tools such as jsPsych. Formal models of JavaScript improve its understanding and help ensure security. JavaScript continues to be an essential and widely adopted language in modern software development.(Resig 2007)

Figure A.7: JavaScript logo

A.2.2.5 Framework

React Native React Native is an open-source framework developed by Facebook that enables the development of native mobile applications using JavaScript and React. It allows developers to build apps for both Android and iOS from a single shared codebase, significantly reducing development time. React Native uses a component-based architecture inspired by React, making UI development modular and efficient. Unlike hybrid approaches, it renders UI components as native platform widgets, delivering near-native performance and appearance. It also permits integration with native modules using Java, Swift, or Objective-C to optimize specific parts of an app. This flexibility makes it suitable for building complex applications with performance needs. React Native has a strong developer community and an extensive ecosystem of libraries and tools, making rapid development and iteration possible. Despite minor performance limitations compared to fully native apps, it performs efficiently for most use cases. These characteristics have made React Native a leading choice in cross-platform mobile development today.(Presa Käld et al. 2021)

Figure A.8: React-Native logo

Babel Babel is a popular open-source JavaScript compiler that allows developers to write modern JavaScript using the latest language features and syntax while maintaining compatibility with older environments. It works by converting ECMAScript 2015+ code into backward compatible JavaScript that can run in browsers or platforms that do not support new features natively. Babel supports a wide range of plugins and presets, enabling customization of the code transformation process, including support for JSX used in React and TypeScript. Its extensible architecture allows developers to create custom plugins tailored to their project's needs. By bridging the gap between new JavaScript advancements and diverse runtime environments, Babel enables developers to use modern syntax and improve productivity without sacrificing browser compatibility. Although newer transpilers like SWC offer faster speeds in some scenarios, Babel remains a widely adopted and reliable tool in JavaScript development.(Prasetya et al. 2025)

Figure A.9: Babel logo

Firebase Firebase is a powerful, cloud based platform developed by Google that offers a wide range of backend services to help developers build, manage, and scale web and mobile applications efficiently. It provides real-time databases like Firebase Realtime Database and Cloud Firestore, which store data as JSON objects and support synchronized, scalable data access—ideal for applications requiring real-time updates, such as tourist tracking apps. Firebase also offers authentication services supporting multiple providers, analytics to monitor user behavior, and tools like Cloud Messaging for push notifications and Crashlytics for crash reporting. Unlike traditional relational databases like MySQL, Firebase's NoSQL structure enables faster data retrieval and simpler handling of applications with less complex relationships. Its seamless integration with Google Cloud services and support for multiple platforms make Firebase a comprehensive backend solution that accelerates development and improves app performance.(Sudiartha et al. 2020)

Figure A.10: Firebase logo

A.3 Implementation

A.3.1 Implementation steps

A.3.1.1 Importing libraries

First, we import all the necessary libraries required to build our recommendation system, which leverages artificial intelligence to analyze soil data collected from sensors. Figure A.11 shows a snippet of code that imports essential libraries such as NumPy and Pandas, in addition to model handling packages.

```python
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.svm import SVC
from sklearn.naive_bayes import GaussianNB
from sklearn.tree import DecisionTreeClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score
from sklearn.preprocessing import StandardScaler, LabelEncoder
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, LSTM, GRU, Dropout
from tensorflow.keras.utils import to_categorical
```

Figure A.11: importing libraries

A.3.1.2 Data splitting

We divided the dataset into three subsets: one for training, one for validation, and one for testing (train/val/test). This division was performed using the code presented in Figure A.12. The LSTM and GRU divisions are shown in Figure 2 A.16

```python
X_train, X_temp, y_train, y_temp = train_test_split(X, y, test_size=0.4, random_state=42)
X_val, X_test, y_val, y_test = train_test_split(X_temp, y_temp, test_size=0.5, random_state=42)
```

Figure A.12: Splitting datase

```python
X_train = X_train.reshape((X_train.shape[0], 1, X_train.shape[1]))
X_val = X_val.reshape((X_val.shape[0], 1, X_val.shape[1]))
X_test = X_test.reshape((X_test.shape[0], 1, X_test.shape[1]))
```

Figure A.13: Splitting the Dataset for LSTM and GRU

A.3.1.3 Training model

The code below handles the training process of our different models.

Training the Naive Bayes (NB) classifier was achieved using the code shown in the figure A.14.

```
param_grid_nb = {'var_smoothing': np.logspace(-9.5, -8.5, 20)}
grid_nb = GridSearchCV(GaussianNB(), param_grid_nb, cv=5, scoring='accuracy', verbose=1)
```

Figure A.14: Naive Bayes Mode

We trained our Decision Tree (DT) model using the following snippet, as illustrated in the figure A.15.

```
param_grid_dt = {'criterion': ['entropy'],'max_depth': [15, 25, 35],'min_samples_split': [5, 12],'min_samples_leaf': [1,5]}
grid_dt = GridSearchCV(DecisionTreeClassifier(random_state=42),param_grid_dt,cv=5,scoring='accuracy', verbose=1,n_jobs=-1)
```

Figure A.15: Decision Tree Mode

The Logistic Regression (LR) model was trained using the script highlighted in the figure below.

```
param_grid = {'C': [0.01, 0.1, 1, 10, 100],'kernel': ['linear', 'rbf', 'poly', 'sigmoid'],'gamma': ['scale', 'auto', 0.01, 0.1, 1]}
grid_search = GridSearchCV(SVC(), param_grid, cv=3, scoring='accuracy', n_jobs=-1)
```

Figure A.16: Logistic Regression Mode

To train the Support Vector Machine (SVM), we used the code shown below, as illustrated in the figure A.17.

```
lr_classifier = LogisticRegression(max_iter=200, solver='liblinear', C=0.5, penalty='l1')
```

Figure A.17: Support Vector Machine Mode

The LSTM-based model was trained using the script provided here, which is demonstrated in the figure A.18.

```
lstm_model.fit(X_train, y_train_cat, epochs=50, batch_size=32, verbose=1, validation_data=(X_val, y_val_cat))
```

Figure A.18: LSTM Mode

As illustrated in the figure A.19, the GRU model's training process is handled by the code block below.

```
gru_model.fit(X_train, y_train_cat, epochs=50, batch_size=16, verbose=1, validation_data=(X_val, y_val_cat))
```

Figure A.19: GRU Mode

A.3.1.4 Test

Following the training of the various models, their performance was assessed on the test dataset using several important statistical metrics, as shown in the figure below .

```
def evaluate_after(name, y_true, y_pred):
    acc = accuracy_score(y_true, y_pred)
    prec = precision_score(y_true, y_pred, average='weighted', zero_division=1)
    rec = recall_score(y_true, y_pred, average='weighted')
    f1 = f1_score(y_true, y_pred, average='weighted')
```

Figure A.20: Test step

A.3.2 Interfaces

In the following, we present screenshots of the main graphical interfaces developed for our application.

A.3.2.1 Get started page

This image shows the initial screen that appears when the application is launched for the first time. It welcomes the user and gives a brief introduction to the app's features as a "Smart Soil Analyzer and Crop Guidance System". It serves as a starting point to guide the user through the next steps using the "Get Started" button.

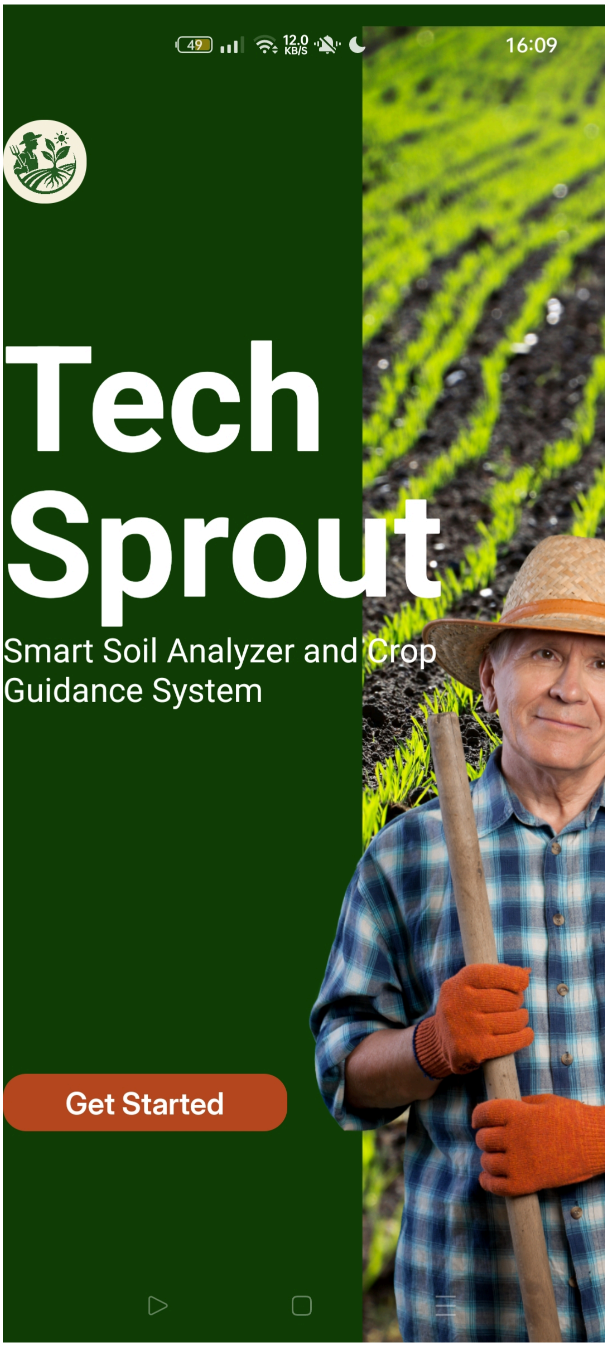


Figure A.21: Get started interface

A.3.2.2 Authentication pages

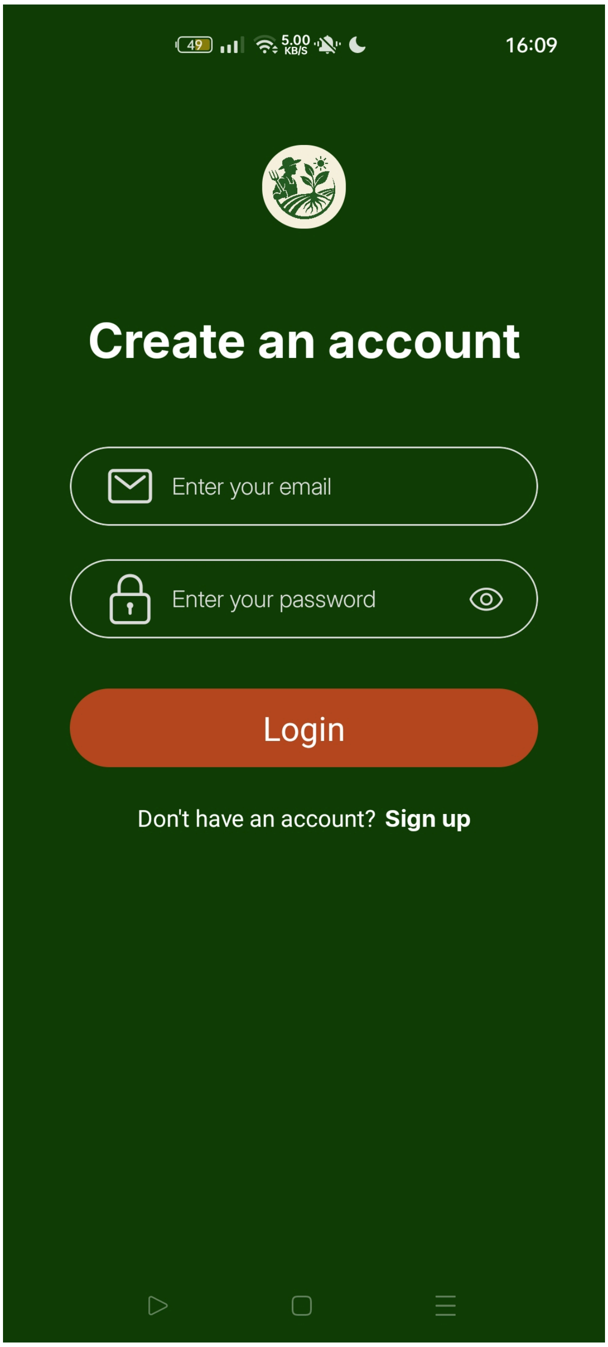


Figure A.22: Login Interface

This figure displays the login screen, allowing existing users to access their accounts by entering their email and password. It is designed for straightforward access to the application's personalized features.

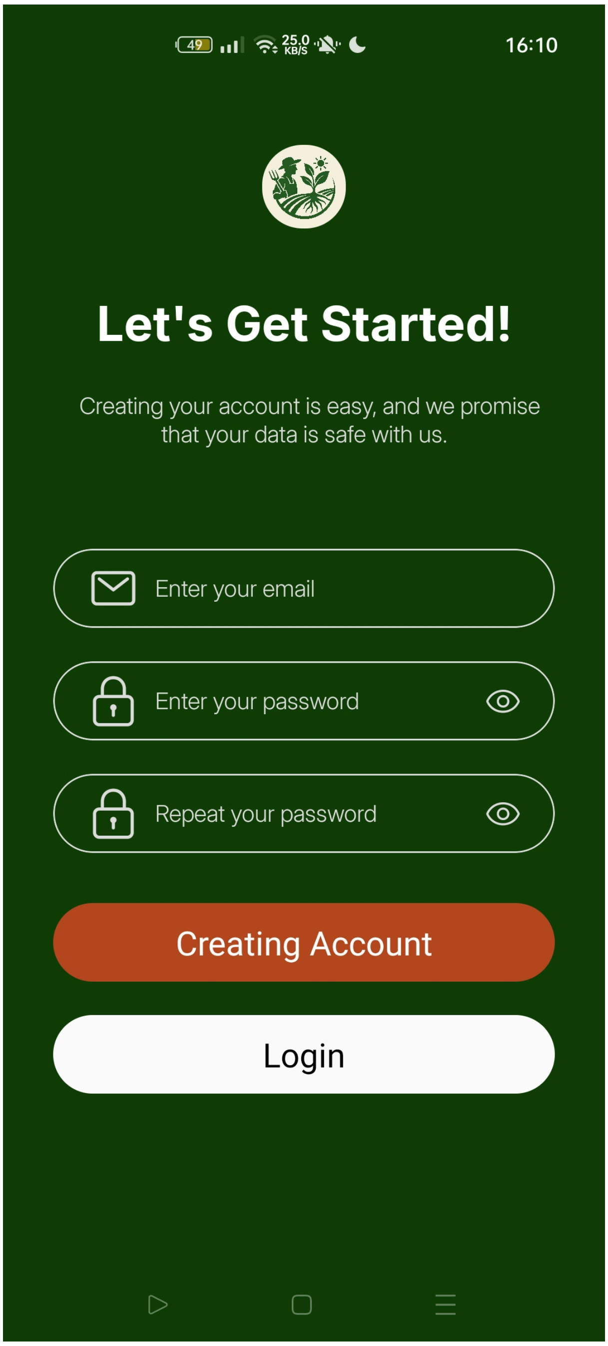


Figure A.23: Sign Up Interface

This image presents the sign-up interface, where new users can easily create an account. It requires entering an email, setting a password, and confirming it, ensuring a simple and secure registration process.

A.3.2.3 Input parameters interface

This figure shows the main input screen where users can enter soil and environmental parameters. This interface is essential for collecting the data required by the crop recommendation engine.

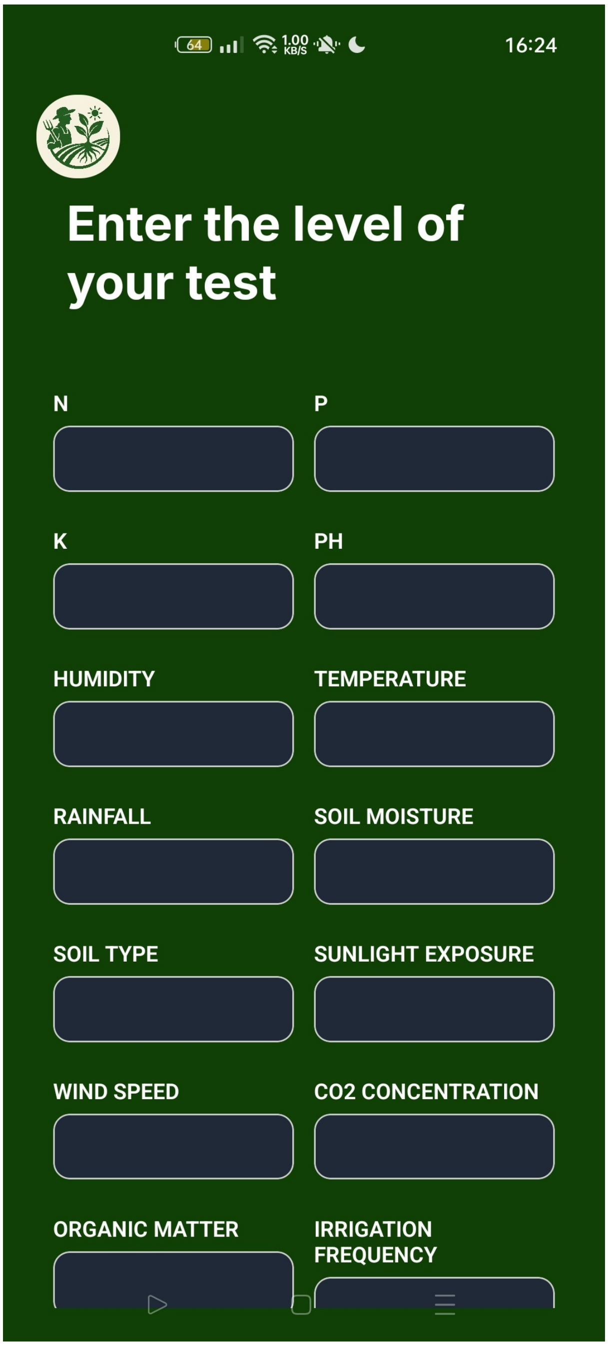


Figure A.24: Input interface for entering soil and environmental parameters.

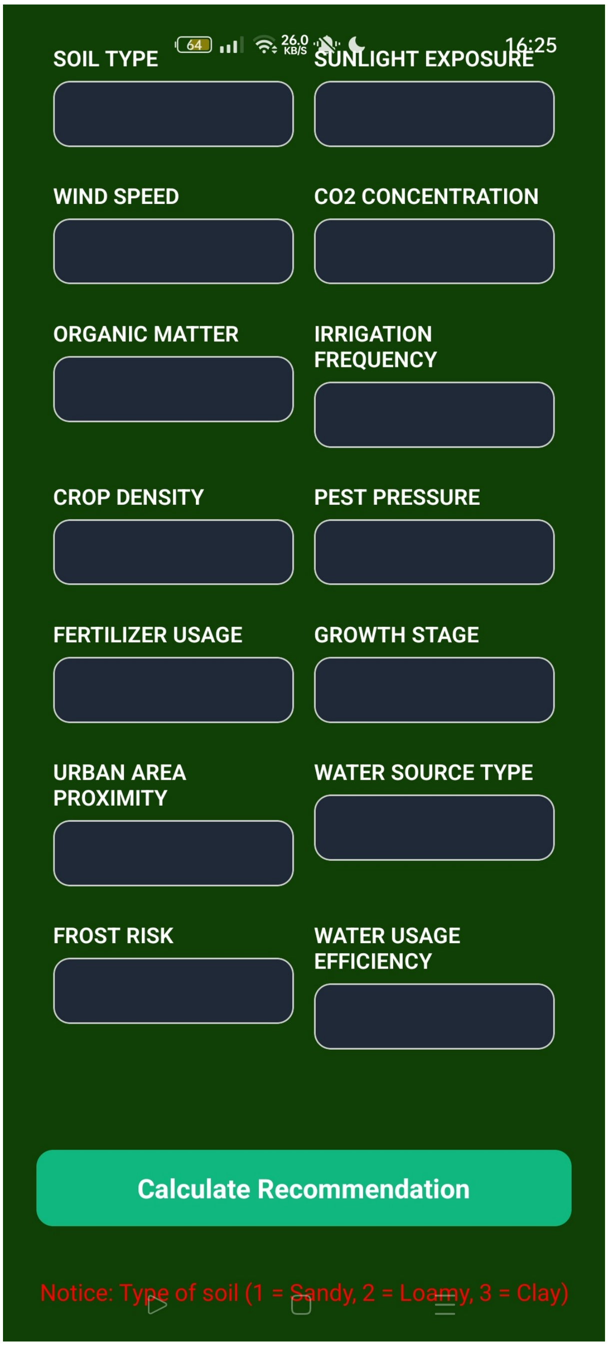


Figure A.25: Screen for users to input soil test levels and environmental data.

A.3.2.4 Recommendation output page

This figure displays the final result provided by the application. After the user inputs the parameters and clicks "Calculate Recommendation", this screen shows the suitable crop based on the analyzed data. An example result shown is "Suitable Crop: chickpeas", confirming the recommended plant for the given conditions.

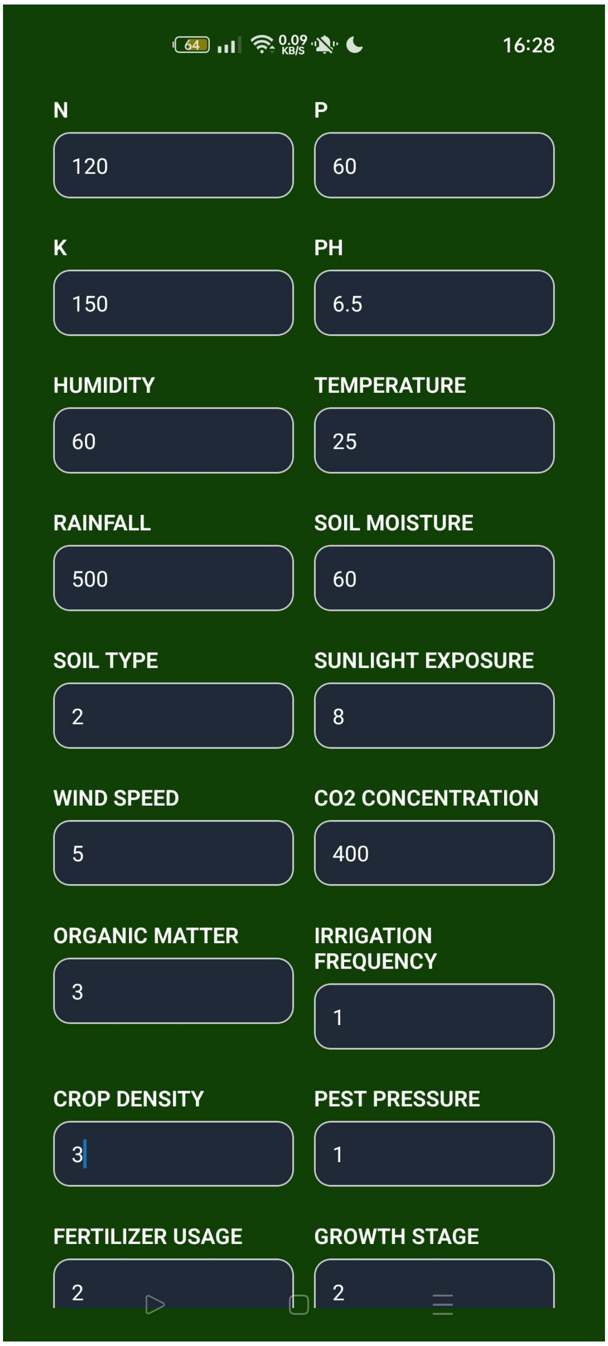


Figure A.26: An example of input parameters entered into the application.

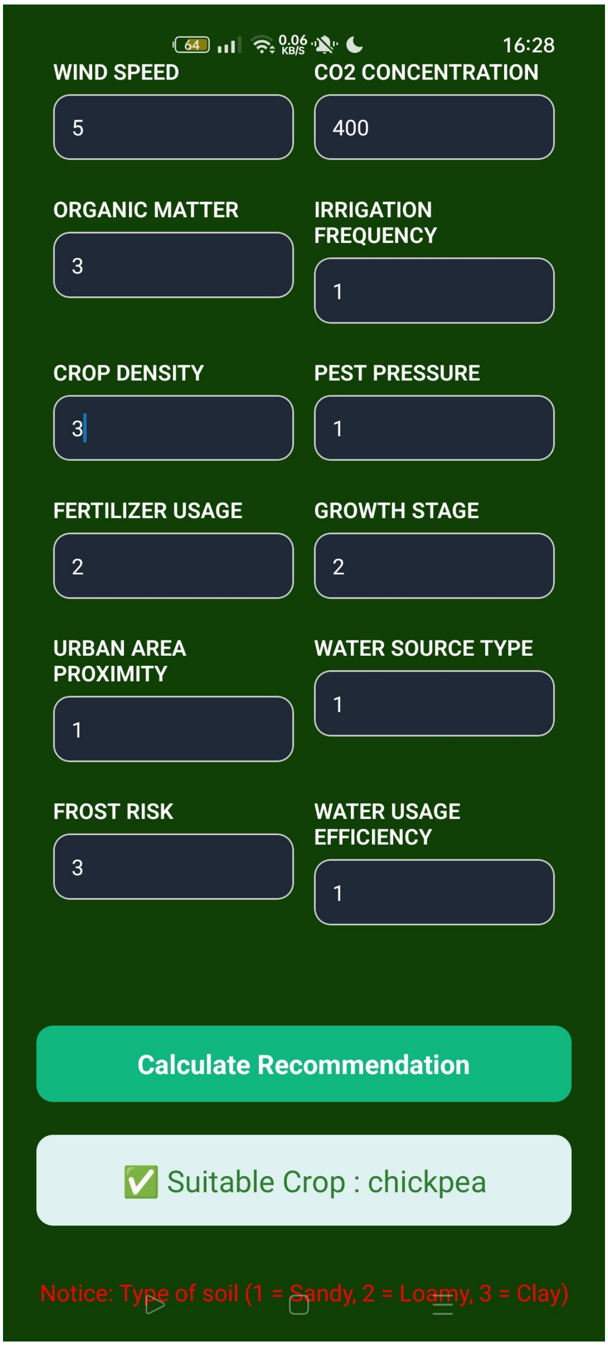


Figure A.27: Output interface showing the recommended crop ("chickpeas") based on input data.

A.4 Conclusion

In this annex, we addressed various aspects of the development of our AI based soil analysis system. We provided a detailed description of the tools used and the implementation steps of the system, starting from data preparation and library importation to model construction, training, and evaluation. Additionally, we included code snippets from our Python implementation to illustrate key components in the development of the crop recommendation system, which relies on soil characteristics and leverages both traditional machine learning and deep learning techniques.

جامعة 20 أوث سكيكدة

كلية العلوم

قسم الإعلام الآلي

مشروع لنيل شهادة الماستر وشهادة مشروع ناشئ

ضمن إطار القرار الوزاري رقم 1275

نظام تحليل التربة الذكي وتوجيه المحاصيل

التخصص: الذكاء الاصطناعي

من إعداد: مروة بوخزنة

تحت إشراف: الدكتورة سميرة حزمون **قسم الإعلام الآلي**

إشراف مشترك: الدكتورة سهيمة بالة **قسم العلوم الفلاحية**

السنة الجامعية: 2024 / 2025

المحتويات

قائمة الأشكال

قائمة الجداول

إهداء

أُهدي ثمرة جهدي المتواضعة إلى من كانوا سببًا في نجاحي:

إلى أرواح والديَّ العزيزين، رحمهم الله وأسكنهم فسيح جناته، فقد كانت محبتهم ودعواتهم ومبادئهم النبيلة زادي ونور طريقي رغم غيابهم.

إلى أختي وعماتي، عرفانًا لكل واحدة منهن على دعمها، وإلى أزواجهن، وخاصة عمي قدور، الذي كان لي بمثابة الأب طوال هذه السنوات.

إلى عائلتي الكبيرة من أعمام وعمات وأبناء وبنات العم وكل الأقارب، شكرًا من القلب على دعواتكم وتشجيعكم وحضوركم الدائم في حياتي، فقد كنتم حقًا سندًا وفخرًا لي.

إلى عائلتي ككل، على الرعاية والدافع والانتماء الذي منحتموني إياه، سأظل ممتنًا لكم دائمًا.

إلى كل أساتذتي الذين رافقوني في رحلتي التعليمية، من المرحلة الابتدائية إلى الجامعة، شكرًا على كل حرف علمتموني إياه وكل قيمة غرستموها فيّ.

إلى زملائي الذين شاركوني هذا المشوار الأكاديمي، كانت صحبتكم ذات معنى وإلهام.

إلى كل من ساعدني بكلمة أو فعل ووقف إلى جانبي لإتمام هذا العمل، أسأل الله أن يجزيكم عني كل خير.

إلى كل من يسعى للعلم ويؤمن برسالته النبيلة.

وإلى كل من يحمل في قلبه لي حبًا أو احترامًا أو تقديرًا... هذا العمل أُهديه لكم.

شكر وتقدير

الحمد لله أولًا وآخرًا، المنعم الكريم، الرحمن الرحيم، الذي منحني القوة والصبر والمثابرة لإتمام هذا العمل، وهداني في مسيرتي الأكاديمية كلها.

أتقدم بأسمى آيات الشكر والامتنان إلى مشرفتَيّ العزيزتَين: الدكتورة سميرة حزمون والدكتورة سهيمة بالة، على توجيهاتهما القيمة، ودعمهما المستمر، ونصائحهما الثمينة طيلة فترة إنجاز هذا البحث. لقد كانت تشجيعاتهما وإخلاصهما عاملًا حاسمًا في توجيه هذا العمل وصقل جودته.

وبمشاعر يغمرها الحنين، أهدي دعاءً خاصًا إلى أرواح والديَّ وإخوتي رحمهم الله، وأسأل الله أن يتغمدهم برحمته ويسكنهم فسيح جناته. لقد كانت محبتهم ودعواتهم وثقتهم بي مصدر قوة وإلهام لا يزال يرافقني حتى بعد رحيلهم.

كما أتوجه بجزيل الشكر إلى عائلتي الكبيرة من أعمام وعمات وأبناء وبنات العم وكل الأقارب، على دعمهم المعنوي وتشجيعهم الذي كان له بالغ الأثر في رحلتي.

ولا يفوتني أن أشكر جميع الأساتذة الذين ساهموا في تكويني العلمي، وكل زملائي وأصدقائي الذين ساندوني ووقفوا إلى جانبي خلال هذا المشوار الأكاديمي.

مع خالص الامتنان والتقدير للجميع.

مقدمة عامة

في عصر تتزايد فيه التحديات المرتبطة بالأمن الغذائي والتغيرات المناخية، أصبحت الزراعة الحديثة مطالبة أكثر من أي وقت مضى بتبني تقنيات ذكية تُمكّن من تحسين استغلال الموارد وتحقيق إنتاجية أعلى بجودة أفضل. ومن أبرز هذه التحديات التي يواجهها المزارعون اليوم هي اختيار المحصول الزراعي الأمثل الذي يتوافق مع خصائص التربة والمناخ المحلي، وذلك لضمان مردود اقتصادي وبيئي فعّال.

غالبًا ما يتم اتخاذ قرارات الزراعة بناءً على الخبرة أو التقليد، دون الاعتماد على تحليل علمي دقيق لمكونات التربة أو الظروف البيئية المحيطة. هذا ما يؤدي في كثير من الحالات إلى زراعة محاصيل غير ملائمة، فتكون النتيجة خسائر مادية وضعف في الإنتاجية، إضافة إلى استنزاف للتربة والموارد المائية.

وفي هذا السياق، تسعى الجزائر إلى تحديث القطاع الزراعي عبر تبني حلول ذكية ومستدامة، انسجامًا مع استراتيجياتها الوطنية لتحقيق الأمن الغذائي وتقليل التبعية للاستيراد. ويُعد التحول الرقمي في الزراعة أحد أبرز ركائز هذا التوجه، حيث تهدف الدولة إلى تشجيع استخدام الذكاء الاصطناعي وإنترنت الأشياء في تطوير أساليب الزراعة والري، وتحسين إدارة التربة والمياه، بما يواكب التحولات البيئية والاقتصادية التي يشهدها العالم.

هنا يأتي دور مشروعنا الذي يُعتمد فيه على تقنيات الذكاء الاصطناعي وتحليل البيانات من أجل تحديد أفضل منتج زراعي يمكن أن تنتجه أرض معينة بناءً على خصائصها الفريدة. حيث يقوم النظام الذي تم تطويره بجمع وتحليل بيانات التربة مثل نسبة النيتروجين، الفسفور، البوتاسيوم، درجة الحموضة، الرطوبة، نوع التربة، التعرض للشمس، ونوعية مصدر المياه، إضافة إلى المعطيات المناخية مثل درجة الحرارة والرطوبة ومعدلات الأمطار والرياح وغيرها.

وباستخدام خوارزميات تعلم الآلة، يُمكن للنموذج اقتراح المحصول الأنسب للزراعة في تلك الأرض، مما يوفر على الفلاح الجهد والتجربة، ويعزز من فرص النجاح الزراعي.

يهدف مشروعنا إلى توفير أداة رقمية ذكية في متناول المزارعين، تسهل عليهم اتخاذ قرارات مبنية على العلم، وتساعدهم في تحسين دخلهم وجودة محاصيلهم. كما يسعى المشروع إلى المساهمة في التنمية الزراعية المستدامة من خلال ترشيد استخدام الموارد ورفع فعالية الإنتاج.

يتناول المحور الأول من هذا العمل تقديم المشروع وأهميته في دعم القرارات الزراعية اعتمادًا على الذكاء الاصطناعي. ويتناول المحور الثاني تفاصيل الابتكار والمزايا المقترحة التي يقدمها النظام مقارنة بالطرق التقليدية. بينما يُخصص المحور الثالث للتحليل الاستراتيجي للسوق، وتشخيص البيئة الزراعية الحالية واحتياجات الفلاحين. المحور الرابع يتناول خطة الإنتاج والتطوير التقني للنموذج، بدءًا من جمع البيانات إلى بناء النموذج التجريبي. أما المحور الخامس فيركز على الخطة المالية

التي تضم دراسة التكاليف، الإيرادات المتوقعة، ونموذج العمل التجاري. وختامًا، يُقدم المحور السادس النموذج التجريبي الأولي للنظام، في حين يتناول المحور السابع نموذج العمل التجاري في صيغته الشاملة.

الفصل 1

تقديم المشروع

1.1 تمهيد

عندما فكرت في تقديم فكرة مشروع مبتكر، أدركت أنه لا بد من العناية والتفكير الدقيق. كنت أبحث عن فكرة تختلف عن الموجود، فكرة تضيف شيئًا جديدًا فعلاً وتقدم قيمة حقيقية للناس. كنت حريصة على أن تكون الفكرة فريدة من نوعها، تحل مشكلة حقيقية ومستمرة في المجتمع، وتلبي حاجة واضحة يشعر بها الكثيرون. تأكدت أيضًا من أن الفكرة قابلة للتنفيذ، وأنها يمكن أن تستمر مع مرور الوقت، خاصة إذا توفرت الموارد المالية والبشرية والتقنية المناسبة. ومن الجانب الربحي، كانت من المهم بالنسبة لي أن تحقق الفكرة عائدًا ماليًا ملموسًا. لذلك ركزت في هذا المشروع على بلورة الفكرة بشكل واضح، وتحديد الفريق سيعمل معي، ووضع أهداف محددة، إلى جانب خطة زمنية تساعدنا على تنفيذ المشروع خطوة بخطوة.

2.1 فكرة المشروع

فكرتي هي تطوير تطبيق ذكي متصل بمُجسَّم آلي يُوضع في التربة. هذا التطبيق يتيح للمستخدمين، خاصة المزارعين، الحصول على بيانات دقيقة لحالة التربة في الوقت الفعلي. من خلال مستشعرات كهربائية ومستشعرات لقياس رطوبة التربة، يقوم الجهاز بتحليل العناصر الغذائية الموجودة، وتحديد المواد الناقصة التي تحتاجها النباتات. يعرض التطبيق هذه المعلومات بطريقة مبسطة، ويوفر توصيات ذكية حول أفضل أنواع المحاصيل التي يمكن زراعتها في المنطقة بناءً على خصائص التربة. يعتمد التطبيق على تقنيات تحليل البيانات الفورية وخوارزميات ذكية لضمان تقديم توصيات فعالة تدعم الزراعة الذكية وتقلل من الهدر في الموارد.

1.2.1 كيف بدأت الفكرة

الإنتاج الزراعي يُعدّ ركيزة أساسية لتحقيق التكامل الاقتصادي، حيث يسهم في تأمين الأمن الغذائي، وتوفير فرص العمل، وتحسين مستويات المعيشة، خاصة في المناطق الريفية. كما يُعزز من قدرة الدول على تحقيق الاكتفاء الذاتي وتقليل الاعتماد على الاستيراد، مما يساهم في استقرار الاقتصاد الوطني. ومع التحديات التي تواجه الزراعة التقليدية، برزت أهمية إدماج التكنولوجيا الحديثة في هذا القطاع. فالتقنيات الذكية، مثل تطبيقات تحليل التربة، تُسهم في تحسين كفاءة الإنتاج الزراعي وتقليل الهدر في الموارد. من هنا، جاءت فكرة تطوير تطبيق ذكي يُغذى بنتائج تحليل التربة، حيث لاحظت أن العديد من المزارعين يواجهون صعوبة في معرفة احتياجات التربة، مما يؤدي إلى زراعة عشوائية وضعف في الإنتاج. يهدف هذا التطبيق إلى تقديم توصيات دقيقة للمزارعين حول العناصر الغذائية الناقصة في التربة، ومستوى الرطوبة، وأفضل المحاصيل التي يمكن زراعتها، مما يساعدهم على اتخاذ قرارات زراعية مدروسة، ويُسهم في تعزيز الإنتاجية وتحقيق التنمية الزراعية المستدامة.

2.2.1 ما الذي سوف نقوم به

1. تصميم وتطوير جهاز الاستشعار وتطبيق الهاتف المحمول

- تصميم واجهة مستخدم بسيطة وسهلة الاستخدام.
- جمع عناصر الجهاز والبدء في تصنيعه.
- ربط التطبيق بجهاز استشعار لجمع بيانات التربة (الرطوبة، درجة الحرارة، pH، العناصر الغذائية).
- عرض حالة التربة في الوقت الفعلي من خلال واجهات رسومية واضحة.
- تطوير خوارزميات لتحليل بيانات التربة وتوليد توصيات فورية.

2. جمع البيانات ودمجها

- استخدام أجهزة استشعار ذكية للحصول على قراءات دقيقة لحالة التربة.
- تحديث البيانات بشكل دوري لضمان دقة التوصيات.
- تخزين البيانات في قاعدة بيانات لمتابعة حالة الأرض على المدى الطويل.

3. تطوير خوارزميات الذكاء الاصطناعي

- تحليل بيانات التربة باستخدام خوارزميات تعلم الآلة لتحديد المحاصيل المثلى.
- تقديم توصيات ذكية بأنسب المحاصيل حسب الموسم وخصائص التربة.

- تخصيص التوصيات بناءً على سلوك المستخدم ونوع أرضه.
- تحسين الخوارزميات تدريجيًا من خلال ملاحظات المستخدمين والتجارب السابقة.

4. اختبار التطبيق وإطلاقه

- اختبار التطبيق من حيث دقة البيانات وسهولة الاستخدام.
- جمع ملاحظات من المزارعين والمستخدمين لتحسين الأداء.
- إجراء التعديلات اللازمة بناء على التغذية الراجعة.

3.2.1 القيم المقترحة

يهدف المشروع إلى تطوير جهاز مدعوم بتطبيق ذكي يُمكّن المزارعين من فهم تربة أراضيهم واستغلالها بأفضل الطرق، بالاعتماد على تحليل البيانات التي توفرها تقنيات الاستشعار الحديثة. ويساعد التطبيق في دعم اتخاذ القرار الزراعي بدقة وفعالية، وذلك من خلال القيم التالية:

سهولة الاستخدام: تصميم واجهة مبسطة وسهلة الفهم، تتيح للمزارعين استخدام التطبيق دون الحاجة إلى خلفية تقنية متقدمة.

اقتراحات مخصصة: تقديم توصيات زراعية دقيقة بناءً على تحاليل التربة وخصائص البيئة الزراعية لكل مستخدم، مثل العناصر الغذائية الناقصة، وأنواع المحاصيل المناسبة.

الدقة والموثوقية: اعتماد التطبيق على بيانات واقعية وتحليل علمي يضمن معلومات موثوقة تساعد في اتخاذ قرارات زراعية فعّالة.

الكفاءة: تحسين الإنتاج الزراعي من خلال تقليل الهدر في المياه والموارد، وزيادة العائد بفضل توصيات دقيقة وموجهة.

الهدف النهائي: تقديم تجربة زراعية ذكية تسهم في تحسين الإنتاجية، تقليل التكاليف، وتحقيق الاستدامة الزراعية بطريقة علمية ومبسطة.

3.1 فريق العمل

يتكون فريق المشروع من ثلاثة أعضاء، كل منهم يتحمل مسؤوليات متعددة لضمان نجاح المشروع، حيث تم توزيع المهام على النحو التالي:

المهام الخاصة بالعضو الأول

(مدير المشروع، مهندس برمجيات، مهندس بيانات، مهندس ذكاء اصطناعي)

- التخطيط العام للمشروع وإدارة الموارد لضمان تنفيذ الجدول الزمني.
- تصميم وبرمجة تطبيق الهاتف المحمول لأنظمة Android وiOS.
- تطوير واجهة المستخدم وربط التطبيق بأجهزة الاستشعار لجمع البيانات.
- جمع، تخزين، وتنظيم بيانات التربة من مصادر متعددة.
- تطوير وتحسين خوارزميات تعلم الآلة لتحليل بيانات التربة وتقديم توصيات ذكية.
- تحسين نماذج الذكاء الاصطناعي بناءً على البيانات وسلوك المستخدم.

المهام الخاصة بالعضو الثاني

(مسؤول اختبار وضمان الجودة، متخصص تسويق وتواصل)

- وضع خطة تسويقية لإطلاق التطبيق والترويج له بين الفلاحين والمزارعين.
- إجراء اختبارات شاملة للتطبيق لضمان دقته وسهولة استخدامه.
- متابعة التقارير وتحليل ملاحظات المستخدمين وتقديم التحسينات اللازمة.
- التواصل مع المستخدمين لجمع التغذية الراجعة واحتياجات السوق.

المهام الخاصة بالعضو الثالث

(مهندس أجهزة واستشعار (Engineer IoT))

- تصميم واختيار أجهزة الاستشعار المناسبة لجمع بيانات التربة، مثل مستشعرات الرطوبة ودرجة الحرارة، وأجهزة قياس الحموضة ومستويات العناصر الغذائية.
- اختبار الأجهزة في البيئات الزراعية لضمان دقة وموثوقية القياسات.
- ضمان تكامل أجهزة الاستشعار مع التطبيق من خلال حلول اتصال فعالة (بلوتوث، واي فاي).
- العمل على تقليل استهلاك الطاقة في الأجهزة لضمان استمرار عملها في الحقول لفترات طويلة دون الحاجة إلى صيانة متكررة.
- توفير الدعم الفني لحل أي مشاكل تقنية تتعلق بالأجهزة أو الربط مع التطبيق.

4.1 أهداف المشروع

تمكين المزارعين من اتخاذ قرارات زراعية ذكية: من خلال توفير تحليل دقيق لحالة التربة وتقديم توصيات واضحة حول ما يمكن زراعته والعناصر الغذائية التي تحتاجها التربة.

زيادة الإنتاجية الزراعية: مساعدة المزارعين على استغلال أراضيهم بشكل أفضل من خلال الزراعة المبنية على بيانات دقيقة بدلاً من التخمين أو العشوائية.

تقليل الهدر في الموارد: من خلال تحديد الاحتياجات الفعلية للتربة وتقديم مواعيد ري دقيقة، مما يقلل من استهلاك الماء والأسمدة.

تحقيق الاستدامة الزراعية: دعم ممارسات زراعية أكثر استدامة وصديقة للبيئة عبر الاستخدام الأمثل للتربة والموارد.

تعزيز الاكتفاء الذاتي: المساهمة في تقليل الاعتماد على الاستيراد الغذائي من خلال تحسين جودة وكمية الإنتاج المحلي.

دعم الاقتصاد المحلي: من خلال تحسين الإنتاج الزراعي وخلق فرص عمل جديدة في مجالات التكنولوجيا الزراعية والدعم الفني.

الوصول للمزارعين في مختلف المناطق: توسيع نطاق استخدام التطبيق ليشمل مختلف المناطق الزراعية، وخاصة المناطق التي تفتقر إلى الإرشاد الزراعي.

رفع الوعي الزراعي: نشر ثقافة الزراعة الذكية وتحفيز المزارعين على استخدام التكنولوجيا في تحليل التربة واتخاذ قرارات مدروسة.

تحسين جودة المحاصيل: من خلال الزراعة المناسبة لكل نوع تربة، بما ينعكس إيجابًا على جودة المنتجات الزراعية وصحتها.

5.1 الجدول الزمني

1.5.1 التخطيط والتحضير

- **تحديد المتطلبات والأهداف:** تحليل السوق الزراعي وتحديد المتطلبات الوظيفية والتقنية لتطبيق اختيار المحاصيل الأفضل.
- **جمع البيانات:** جمع بيانات التربة، المناخ، أنواع المحاصيل، خصائص البيئة الزراعية، وكل المعلومات اللازمة لتطوير التطبيق.

- **البحث عن أجهزة الاستشعار المناسبة:** دراسة وتحديد أنواع أجهزة الاستشعار الأنسب لقياس خصائص التربة مثل الرطوبة، درجة الحرارة، الحموضة، والمغذيات، بما يتناسب مع بيئات الزراعة المستهدفة.

2.5.1 التصميم و التطوير

- **تصميم قاعدة بيانات:** تصميم قاعدة بيانات متكاملة لتنظيم بيانات التربة، المحاصيل، وبيانات أجهزة الاستشعار بطريقة تُمكن من الوصول والتحليل بكفاءة.
- **تطوير النموذج الأولي:** إنشاء نموذج أولي للتطبيق لاختبار واجهة المستخدم وتجربة تدفقات الاستخدام لدى المزارعين.
- **تطوير البرمجيات:** برمجة التطبيق الأساسي على منصات الهواتف الذكية وربطه بقاعدة البيانات وبأجهزة الاستشعار لجمع البيانات في الوقت الحقيقي.
- **تصميم واجهة المستخدم:** تصميم واجهة مستخدم بديهية وسهلة الاستخدام، تلائم احتياجات المزارعين وتُظهر التوصيات بشكل مبسط وواضح.
- **تطوير خوارزميات الذكاء الاصطناعي:** تدريب خوارزميات تعلم الآلة باستخدام بيانات التربة والمناخ لتحديد المحاصيل المثلى لكل منطقة أو مزرعة.
- **اختبار التطبيق:** إجراء اختبارات شاملة لضمان جودة الأداء، دقة النتائج، وسهولة استخدام التطبيق في ظروف العمل الزراعي.
- **أجهزة الاستشعار المدمجة:**سيتم استخدام جهاز واحد مدمج يحتوي على مجموعة من أجهزة الاستشعار المتخصصة لجمع بيانات التربة بدقة عالية، بما في ذلك مستشعرات رطوبة التربة، ومستشعرات درجة الحرارة، وأجهزة قياس الحموضة (pH)، وأجهزة قياس مستويات العناصر الغذائية، بحيث يتيح هذا الجهاز توفير معلومات شاملة وموثوقة تساعد في تحليل حالة التربة واتخاذ القرارات الزراعية المناسبة.

3.5.1 الإطلاق والتسويق

- **إعداد استراتيجية التسويق:** إعداد وتنفيذ خطة تسويقية موجهة للمزارعين، تركز على أهمية التطبيق في تحسين الإنتاج الزراعي من خلال تحليل بيانات التربة. سيتم إبراز سهولة استخدام النظام ودقة توصياته الذكية بناءً على الذكاء الاصطناعي.
- **التوعية بمميزات وحدة الاستشعار:** سيتم التركيز على الترويج لوحدة الاستشعار المدمجة بشكل منفصل، باعتبارها جزءاً أساسياً من الحل التقني. تحتوي الوحدة على مستشعرات متعددة مثل رطوبة التربة، درجة الحرارة، الحموضة (pH)، والعناصر الغذائية، وتمتاز بسهولة الاستخدام والتركيب، وصيانتها منخفضة التكاليف، مما يجعلها مناسبة للبيئة الزراعية.

- **إطلاق النسخة التجريبية:** سيتم توفير نسخة تجريبية (Beta) من التطبيق بالتوازي مع تجربة ميدانية لوحدة الاستشعار، لجمع آراء المستخدمين حول كل من التطبيق والوحدة ومدى تكامل الأداء بينهما.

- **إطلاق النسخة النهائية:** نشر التطبيق على متاجر التطبيقات مع تضمين الإرشادات الفنية لتثبيت وحدة الاستشعار واستخدامها بالشكل الأمثل، بالإضافة إلى توفير قنوات دعم فني وتحديثات دورية لضمان تجربة مستخدم متكاملة وسلسة.

4.5.1 الدعم والتحسين المستمر

- **دعم العملاء:** توفير دعم فني شامل للمستخدمين لتسهيل استخدام التطبيق ووحدة الاستشعار، ويشمل ذلك الرد على الاستفسارات، تقديم المساعدة التقنية، ومعالجة المشكلات التي قد تظهر أثناء التشغيل.

- **التحسينات والتحديثات:** تحديث التطبيق بشكل دوري بناءً على ملاحظات المستخدمين وتحليل البيانات المجمعة، بهدف تحسين الأداء وزيادة كفاءة الخوارزميات والوظائف.

- **تقديم صيانة للجهاز:** سيتم تقديم خدمات صيانة دورية لوحدة الاستشعار لضمان استمرار عملها بكفاءة عالية في البيئات الزراعية. تشمل الصيانة التحقق من سلامة المستشعرات، تنظيفها، واستبدال الأجزاء التالفة عند الحاجة، مما يضمن دقة البيانات واستمرارية استخدام النظام بدون انقطاع.

يوفر هذا الجدول الزمني خريطة طريق واضحة لتنفيذ مشروع الزراعة الذكية، حيث يضمن أن كل مرحلة من مراحل المشروع تحصل على الوقت والموارد اللازمة لتحقيق الأهداف بكفاءة وفعالية.

المرحلة	المدة الزمنية (بالأسابيع)
التخطيط والتحضير	12
التصميم والتطوير	25
الإطلاق والتسويق	8
الدعم والتحسين المستمر	مستمر
إجمالي المدة الزمنية	**45 أسبوعًا**

جدول 1.1: الجدول الزمني لتنفيذ تطبيق الزراعة الذكي

الفصل 2

الجوانب الإبتكارية

1.2 طبيعة الإبتكارات

يعرف الابتكار التكنولوجي بأنه عملية إنشاء وتطبيق تقنيات وأدوات وأنظمة وعمليات جديدة أو محسنة تُحدث تقدمًا كبيرًا في مختلف المجالات. يتضمن هذا الابتكار تسخير المعرفة والخبرة والموارد لتطوير حلول مبتكرة تُعالج المشكلات، وتحسن الكفاءة، وتدفع بالتقدم، وتُضيف قيمة. استنادًا إلى هذا التعريف، تمكنا من تصنيف مشروعنا ضمن الابتكارات التكنولوجية.

2.2 مجالات الإبتكار

تعتبر مراقبة وتحليل بيانات التربة عاملاً حاسماً لتحسين الإنتاج الزراعي، لكن غالباً ما يواجه المزارعون صعوبة في الحصول على المعلومات الدقيقة والموثوقة باستخدام الأدوات التقليدية. من هذا المنطلق، يسعى مشروعنا لتحقيق تحول في تجربة المزارعين من خلال تقديم نظام متكامل يجمع بين تطبيق ذكي وأجهزة استشعار دقيقة، مما يمكّن المستخدمين من مراقبة حالة التربة واتخاذ قرارات زراعية مبنية على بيانات دقيقة وسهلة الوصول.

1.2.2 تطبيق تكنولوجيا متقدمة

يعتمد مشروعنا على مجموعة من التقنيات التكنولوجية المتقدمة لتحسين تجربة المزارعين في مراقبة وتحليل بيانات التربة. إليك بعض التفاصيل:

- **تطبيق الهاتف المحمول:** يتم تطوير تطبيق ذكي يعمل على نظامي Android وiOS، يتيح للمزارعين الوصول إلى بيانات التربة والتوصيات الزراعية بسهولة وفي الوقت الحقيقي.

- **أجهزة الاستشعار المتكاملة:** يستخدم المشروع أجهزة استشعار متخصصة لجمع بيانات دقيقة حول رطوبة التربة، درجة الحرارة، الحموضة، ومستويات العناصر الغذائية، مما يوفر معلومات شاملة تدعم اتخاذ القرار الزراعي.

- **تقنيات الذكاء الاصطناعي :(AI)** تُستخدم خوارزميات الذكاء الاصطناعي لتحليل بيانات التربة، وتقديم توصيات ذكية حول المحاصيل المثلى وأساليب الري والتسميد.

- **تقنيات تعلم الآلة (Machine Learning):** تعتمد تقنيات تعلم الآلة على تحليل بيانات المستخدم وسلوك المزرعة لتحسين دقة التنبؤات وتخصيص التوصيات وفقاً لاحتياجات كل مزرعة.

تجمع هذه التقنيات بين الابتكار والدقة لتوفير نظام متكامل يدعم المزارعين في زيادة إنتاجيتهم وتحسين جودة المحاصيل بطريقة مستدامة وفعالة.

2.2.2 نموذج الأعمال المبتكر

- **الاشتراك السنوي:** يُتيح النموذج للمزارعين الاشتراك في نظام مراقبة وتحليل بيانات التربة عبر التطبيق، مع خيارات مرنة للاشتراك السنوي حسب حجم المزرعة واحتياجاتها.

- **العروض والخصومات:** يتم تقديم خصومات خاصة للمزارعين الذين يستخدمون النظام بانتظام، مما يعزز استمرارية الاستخدام ويشجع على التحول نحو الزراعة الذكية.

- **خدمات الدعم والتدريب:**يتضمن النموذج تقديم جلسات تدريبية دورية ومواد تعليمية مبسطة (مثل فيديوهات أو كتيبات) لمساعدة المزارعين على فهم استخدام التطبيق، قراءة البيانات الصادرة من أجهزة الاستشعار، وتطبيق التوصيات الزراعية بشكل عملي في الحقل.

يهدف هذا النموذج إلى تزويد المزارعين بمنصة تقنية متكاملة تساعدهم على اتخاذ قرارات مدروسة، وتحسين جودة التربة والمحاصيل بطريقة مستدامة وسهلة الوصول.

3.2.2 العروض والخصومات

- **عرض خاص للمشترين الأوائل للجهاز:** يحصل أول المشترين للجهاز على اشتراك مجانية في التطبيق لمدة سنة كاملة، لتشجيع تبني التقنية والاستفادة القصوى من الخدمات.

- **عروض الموسم:** تقديم عروض خاصة خلال المواسم الزراعية أو المناسبات، مثل تخفيضات على الاشتراكات أو خدمات إضافية مجانية.

- **عروض الشراكات:** إمكانية التعاون مع متاجر وموردي المنتجات الزراعية لتقديم خصومات أو عروض مشتركة للمستخدمين.

4.2.2 التحليل البياني

- **تحليل المناطق التي تم العمل فيها أو الوصول إليها:** يتم جمع بيانات جغرافية دقيقة حول المناطق التي يستخدم فيها المزارعون التطبيق والجهاز، وذلك لفهم التوزيع الجغرافي للمزارع ونوعية التربة في كل منطقة. يساعد هذا التحليل في تحديد المناطق ذات الإنتاجية العالية أو التي تحتاج إلى دعم إضافي. كما يُمكن استخدام النتائج لتخصيص توصيات زراعية موجهة حسب خصائص كل منطقة، مما يُعزز من فعالية التطبيق ويساهم في تحسين النتائج الزراعية بشكل مخصص وفعّال.

- **تحليل بيانات الاستخدام:**يتم جمع وتحليل بيانات استخدام التطبيق والخدمة، مثل عدد المستخدمين النشطين، ومدة الاستخدام، والميزات الأكثر استخدامًا، وذلك بهدف تحسين تجربة المستخدم وتطوير الخدمات المقدمة.

- **تحليل ردود الفعل:** يتم جمع ملاحظات المستخدمين وتحليلها لفهم ما يعجبهم وما لا يعجبهم في التطبيق، ويمكن استخدام هذه الملاحظات لتحسين التطبيق وتلبية احتياجات المستخدمين بشكل أفضل.

- **تحليل تجربة المستخدم:** يتم تحليل كيفية تفاعل المستخدمين مع التطبيق، ويمكن تحليل عناصر واجهة المستخدم لتحسين تجربة المستخدم وجعلها أكثر سهولة وفعالية.

- **تحليل الأداء:** يتم مراقبة أداء التطبيق والخدمة بشكل عام، مثل سرعة التحميل واستجابة التطبيق، ويمكن اتخاذ التحسينات اللازمة لتحسين أداء التطبيق.

5.2.2 التكامل الشامل

يشير التكامل الشامل في هذا السياق إلى توفير تجربة زراعية متكاملة للمزارعين من خلال دمج مختلف عناصر النظام من التطبيق إلى الجهاز بطريقة سلسة ومنسقة. وفيما يلي أبرز الجوانب التي تُسهم في تحقيق هذا التكامل:

- **تكامل البيانات الزراعية:** ربط بيانات الرطوبة، ودرجة الحرارة، ودرجة الحموضة، والعناصر الغذائية المجمعة من الجهاز بشكل مباشر مع قاعدة بيانات التطبيق لتوليد تقارير وتوصيات فورية.

- **تكامل واجهة الاستخدام:** تصميم واجهة تطبيق موحدة تسهّل على المزارع الوصول إلى المعلومات، تتبع حالة التربة، استلام التنبيهات، وتطبيق الإرشادات الزراعية بسهولة دون الحاجة للتنقل بين منصات مختلفة.

- **تكامل التوصيات الذكية:** اعتماد خوارزميات تعلم آلي لتحليل البيانات المجمعة (مثل نوع التربة، درجة الحموضة، الرطوبة، ودرجة الحرارة) لتوليد توصيات زراعية مخصصة لكل مزارع، تشمل تحديد **أفضل المحاصيل المزروعة** التي تناسب ظروف الأرض والمناخ، مما يساعد في زيادة الإنتاجية وتحسين جودة المحاصيل

- **تكامل التخزين السحابي:** تخزين البيانات تلقائيًا في السحابة لضمان عدم فقدانها، وتمكين الوصول إليها من أي جهاز في أي وقت.

من خلال هذا التكامل الشامل، يصبح النظام الزراعي أكثر كفاءة وتطورًا، ويساعد المزارعين على اتخاذ قرارات مدروسة قائمة على بيانات دقيقة في الوقت الفعلي، مع تقديم توصيات مخصصة تضمن اختيار المحاصيل الأنسب لكل منطقة.

الفصل 3

التحليل الاستراتيجي للسوق

1.3 عرض القطاع السوقي

1.1.3 السوق المحتمل

يتمثل السوق المحتمل لهذا المشروع في المناطق الريفية والزراعية التي تعتمد بشكل كبير على الزراعة كمصدر دخل رئيسي. ويشمل هذا السوق:

- **المزارع النموذجية:** يمكن أن تستفيد هذه المزارع من أحدث التقنيات الزراعية لزيادة الإنتاج وتحسين الجودة.
- **القرى والمناطق الزراعية:** حيث يشكل المزارعون النسبة الأكبر من السكان ويعتمدون على تقنيات تقليدية، مما يجعلهم بحاجة ماسة إلى حلول ذكية لتحسين الإنتاجية.
- **البيوت المحمية والمزارع المتطورة:** التي تبحث عن تقنيات حديثة لتحسين جودة الزراعة وكفاءة استهلاك الموارد مثل المياه والأسمدة.
- **المناطق التي تعاني من تدهور في جودة التربة:** والتي يمكن أن تستفيد من التحليلات الدقيقة والتوصيات الذكية التي يقدمها النظام لتحسين خصوبة التربة وزيادة الإنتاج.
- **المدارس والجامعات الزراعية:** حيث يمكن استخدام التطبيق والجهاز لأغراض تعليمية وتدريبية في مجالات الزراعة الذكية والاستشعار البيئي.
- **الجمعيات التعاونية والمؤسسات الزراعية:** التي تسعى إلى دعم أعضائها من المزارعين عبر حلول تقنية متكاملة تسهّل عمليات الزراعة والتخطيط والإدارة.

2.1.3 أعداد السوق المحتملين

يُعد القطاع الزراعي في الجزائر أحد ركائز الاقتصاد الوطني، وقد شهد نموًا ملحوظًا في عام 2024. ووفقًا لتصريحات الرئيس عبد المجيد تبون بتاريخ 27 نوفمبر 2024، ساهم هذا القطاع بنسبة 15% من الناتج المحلي الإجمالي، أي ما يعادل نحو 37 مليار دولار أمريكي أي ما يعادل نحو 4.995 تريليون دينار جزائري، مما يعكس حيوية هذا السوق ودوره المركزي في التنمية الاقتصادية. كما سجّل القطاع نموًا سنويًا بنسبة 5.1 %، ونموًا بنسبة 5.2 % في الربع الرابع من العام، بحسب بيانات الديوان الوطني للإحصائيات.

وفيما يلي أبرز المؤشرات التي توضح حجم السوق المحتمل لمشروع الزراعة الذكية في الجزائر:

- **العمالة الزراعية:** يعمل في القطاع الزراعي حوالي 6.2 مليون شخص، وهو ما يمثل أكثر من ربع اليد العاملة الوطنية، بحسب تصريحات رسمية لوزارة الفلاحة والتنمية الريفية.
- **المساحات الزراعية:** تُقدّر المساحة الزراعية المستغلة بنحو 5.8 مليون هكتار، مع توسع في المساحات المروية نتيجة للاستثمارات الحديثة في البنية التحتية الزراعية، وفقًا لتقارير رسمية نُشرت في 2024.
- **المزارع النشطة:** تشير التقديرات إلى وجود أكثر من 2.1 مليون مزرعة موزعة عبر التراب الوطني، مما يوفّر أرضية واسعة لتطبيق الحلول التكنولوجية الزراعية الحديثة.
- **مشاريع استراتيجية:** شهد عام 2024 إطلاق مشاريع استثمارية كبرى، أبرزها المشروع المتكامل في ولاية تيميمون على مساحة 36 ألف هكتار لإنتاج الحبوب والبقوليات، بالإضافة إلى إنشاء وحدات تحويلية، وفقًا لتصريحات وزير الفلاحة محمد عبد الحفيظ هني خلال جلسة للبرلمان في ديسمبر 2024.
- **الأمن الغذائي:** يُغطي القطاع الزراعي نحو 75% من احتياجات الجزائر الغذائية، وهو ما أكده الرئيس تبون خلال خطابه بمناسبة الأسبوع الوطني للفلاحة في نوفمبر 2024، ما يعزز من أهمية تطوير هذا القطاع في إطار الأمن الغذائي الوطني.
- **التوجه التكنولوجي:** تعرف الفلاحة الجزائرية توجهًا متزايدًا نحو رقمنة القطاع وتشجيع استخدام التقنيات الذكية، مثل أجهزة الاستشعار، تحليل التربة، والأنظمة الزراعية المتصلة، لرفع الإنتاجية وضمان الاستدامة.

بناءً على هذه المؤشرات الرسمية والموثوقة، يُمثل السوق الزراعي الجزائري فرصة واعدة لتطبيق مشروع الزراعة الذكية القائم على تحليل البيانات والتوصيات المخصصة، بما يضمن دعم الفلاحين، وتحقيق مردودية أفضل، والمساهمة في تعزيز الأمن الغذائي بطريقة فعالة ومستدامة.

المؤشر	القيمة / الرقم	المصدر
مساهمة القطاع الزراعي في الناتج المحلي الإجمالي	15% (حوالي 37 مليار دولار)	خطاب الرئيس تبون – نوفمبر 2024
النمو السنوي للقطاع الزراعي	5.1 %	الديوان الوطني للإحصائيات
النمو في الربع الرابع	5.2 %	الديوان الوطني للإحصائيات
عدد العمال الزراعيين	6.2 مليون شخص (أكثر من ربع اليد العاملة)	وزارة الفلاحة والتنمية الريفية
عدد المزارع النشطة	أكثر من 2.1 مليون مزرعة	تقديرات قطاع الفلاحة
المساحة الزراعية المستغلة	حوالي 5.8 مليون هكتار	وزارة الفلاحة
تغطية الأمن الغذائي	75% من الاحتياجات الغذائية الوطنية	خطاب الرئيس تبون – نوفمبر 2024
مشروع استثماري بارز	36 ألف هكتار في تيميمون (حبوب وبقوليات)	وزير الفلاحة محمد عبد الحفيظ هني – ديسمبر 2024
التوجه الرقمي	اعتماد متزايد على الزراعة الذكية والتقنيات الحديثة	سياسات الوزارة 2024

جدول 3.1: مؤشرات توضح السوق المحتمل لمشروع الزراعة الذكية في الجزائر

3.1.3 السوق المستهدف

يستهدف مشروعنا الزراعي الذكي فئة المزارعين الذين يسعون لتحسين إنتاجيتهم من خلال استخدام تقنيات حديثة في الزراعة. ويشمل ذلك:

- **المزارع النموذجية:** التي تسعى إلى تطبيق أحدث التقنيات والممارسات الزراعية لتعظيم الإنتاجية والجودة، وتمثل مراكز للابتكار والتعلم في القطاع الزراعي.
- **المزارعون التقليديون:** الذين يرغبون في الانتقال إلى الزراعة الذكية لمواكبة التغيرات المناخية وتحسين جودة الإنتاج.
- **الشركات الفلاحية والمتعاونات الزراعية:** التي تسعى لرفع كفاءة العمل الزراعي وتحسين العوائد الاقتصادية.

- **المستثمرون في الزراعة الصحراوية:** والمشاريع الكبرى التي تهتم بتحقيق الأمن الغذائي على نطاق واسع.
- **الفلاحون الشباب ورواد الأعمال الزراعيون:** الذين يهتمون بالتقنيات الحديثة ويرغبون في حلول رقمية متكاملة لإدارة مزرعتهم.
- **الجهات الحكومية والمؤسسات البحثية:** المهتمة بجمع بيانات دقيقة عن التربة والإنتاج لتحسين الخطط والسياسات الزراعية.

يتميز هذا السوق بوعي متزايد بأهمية **التحول الرقمي** في الزراعة، ويبحث المستخدمون عن أدوات تُمكّنهم من اتخاذ قرارات زراعية مستنيرة بناءً على **تحليلات فورية للبيانات** دون الحاجة إلى الاعتماد الكلي على الخبرة التقليدية أو التقدير الشخصي.

2.3 التحليل الإستراتيجي للسوق

1.2.3 من سيستخدم تطبيقنا الزراعي

- **الفلاحون الأفراد:** ممن يزرعون في مساحات صغيرة أو متوسطة، ويبحثون عن أدوات لتحسين الإنتاجية ومراقبة حالة التربة.
- **الشركات الفلاحية الكبرى:** التي تعمل على نطاق واسع وتحتاج إلى حلول رقمية لإدارة عمليات الزراعة وتحسين الكفاءة.
- **المستثمرون الزراعيون الجدد:** وخاصةً في المشاريع الصحراوية أو الزراعات الاستراتيجية مثل الحبوب والبقوليات.
- **المهندسون الزراعيون ومراكز الإرشاد:** الذين يستخدمون بيانات دقيقة لتحليل التربة وتقديم التوصيات للمزارعين.
- **الطلاب والباحثون في المجال الزراعي:** الباحثين عن أدوات تعليمية وبيانية حديثة لاستخدامها في التدريب والأبحاث.

2.2.3 دوافع الاستخدام

- **الدقة والموثوقية:** يقدم التطبيق بيانات حية حول رطوبة التربة، ودرجة الحموضة، واحتياج النبات للعناصر الغذائية.
- **التخصيص:** يولّد توصيات زراعية مخصصة بناءً على نوع التربة، المحاصيل المزروعة، والموقع الجغرافي.
- **التوفير في التكاليف:** من خلال تجنّب الإفراط في استخدام المياه أو الأسمدة، وتحسين توقيت الزراعة والري.
- **تحسين جودة الإنتاج:** باستخدام بيانات دقيقة لاتخاذ قرارات زراعية أفضل تؤدي إلى محصول أغنى وجودة أعلى.

3.2.3 مبررات اختيار هذا السوق

- **أهمية القطاع الزراعي:** يساهم بنسبة 15% من الناتج المحلي الإجمالي، ويوظف أكثر من 6.2 مليون شخص.
- **دعم الدولة للاستثمار في الزراعة:** مثل مشروع تيميمون على مساحة 36 ألف هكتار واستهداف زرع 2.3 مليون هكتار من الحبوب.
- **التوجه نحو التقنيات الحديثة:** تشجيع الدولة والمزارعين على استخدام الزراعة الذكية لزيادة الإنتاج والاستدامة.
- **الحاجة لتحسين الأمن الغذائي:** حيث يوفر القطاع الزراعي 75% من الاحتياجات الغذائية الوطنية.
- **التحديات المناخية:** ارتفاع الحرارة ونقص المياه يتطلب حلولاً ذكية للتكيّف وتحسين كفاءة استخدام الموارد.

4.2.3 إمكانية إبرام عمليات شراء

- **شراكات مع التعاونيات الزراعية:** لتوفير الأجهزة والخدمات على نطاق جماعي بأسعار تفضيلية.
- **اتفاقيات مع مؤسسات الدعم والتمويل:** لتقديم التطبيق ضمن مشاريع الدعم الريفي أو برامج التمويل الفلاحي.
- **خطة تسويق موجهة:** تقديم نسخة مجانية محدودة من التطبيق، مع إصدار مدفوع يحتوي على ميزات متقدمة مثل التوصيات التنبؤية والتحليلات الذكية.
- **فرص تجارية إضافية:** مثل بيع البيانات التحليلية للمؤسسات البحثية أو شركات إنتاج الأسمدة، والترويج لمنتجات زراعية محلية داخل التطبيق.

3.3 قياس شدة المنافسة في السوق الجزائرية

1.3.3 المنافسين المباشرين

تطبيق AgriDZ : هو تطبيق محلي يقدم معلومات إرشادية للمزارعين حول مواعيد الزرع والطقس وبعض أنواع المحاصيل. يتميز التطبيق بواجهة استخدام بسيطة ودعم لغوي محلي، مما يجعله ملائمًا للفلاحين الجزائريين. أما نقاط الضعف فهي افتقاره إلى التحليل المتقدم أو التخصيص الدقيق بناءً على خصائص الأرض أو نوع المحصول، كما أن المحتوى غالبًا ما يكون ثابتًا ولا يعتمد على بيانات حية أو تقنيات الاستشعار الحديثة. ومن حيث الحصة السوقية، يُعتقد أن التطبيق يمتلك انتشارًا محدودًا نتيجة هذه القيود التقنية.

المبادرات الحكومية: هناك بعض المبادرات أو المنصات التي تشرف عليها وزارة الفلاحة والتنمية الريفية، وتوفر عبر بواباتها الرسمية معلومات موسمية وإرشادات زراعية. تتميز هذه المبادرات بثقة المستخدمين كونها مصدر رسمي، ولكنها تفتقر في العادة إلى التفاعلية والتقنيات الذكية أو التحليل البياني المتقدم، مما يجعلها أقل جاذبية للفلاحين الذين يبحثون عن حلول مبتكرة وسريعة.

2.3.3 المنافسين الغير مباشرين

تطبيقات دولية مثل CropIn وAgriApp: هذه التطبيقات تقدم أدوات تحليل متقدمة تعتمد على الذكاء الاصطناعي وبيانات الأقمار الصناعية، وتتمتع بميزات تقنية متطورة جدًا. إلا أن هذه التطبيقات لا تدعم اللغة العربية أو السياق المحلي الجزائري، كما أن أسعار الاشتراك فيها مرتفعة نسبيًا، مما يحد من انتشارها بين الفلاحين في الجزائر، خصوصًا في المناطق الداخلية أو الصحراوية. لذلك، بالرغم من قوتها التقنية، فإن تأثيرها في السوق المحلي محدود.

يمكن القول إن السوق الزراعي الجزائري يعاني من نقص في التطبيقات الذكية الشاملة التي تجمع بين البساطة، والدقة، واستخدام التقنيات الحديثة كالذكاء الاصطناعي والاستشعار عن بعد، مما يفتح فرصة كبيرة لتطوير تطبيق مبتكر يلبي حاجات الفلاحين بشكل مخصص وبكلفة مناسبة، ويعزز من فرص نجاحه وانتشاره في السوق المحلي.

4.3 الإستراتيجية التسويقية

1.4.3 دراسة السوق المستهدفة

الفئة المستهدفة الأساسية في هذا المشروع تشمل الفلاحين في الجزائر، خصوصًا في المناطق الداخلية والصحراوية، بالإضافة إلى المهندسين الزراعيين والمستثمرين الصغار في القطاع الفلاحي. احتياجاتهم تتمثل في الحصول على توصيات دقيقة حول أفضل نوع يمكن إنتاجه حسب خصائص الأرض والموسم، مع واجهة سهلة الاستخدام.

2.4.3 تحديد الميزة التنافسية

يستخدم المشروع تقنيات الذكاء الاصطناعي لتحليل بيانات الأراضي والمواسم الزراعية بهدف تقديم توصيات دقيقة ومعلومات مخصصة تتناسب مع الظروف المحلية. كما يتميز بتقديم إرشادات شخصية للفلاحين تعتمد على بيانات أرضهم ونوع المحصول، مما يساهم في تحسين الإنتاجية وجودة المحاصيل بشكل فعّال.

3.4.3 استخدام قنوات التسويق الفعالة

1. **التسويق الرقمي**

- الإعلانات على وسائل التواصل الاجتماعي: الاستفادة من فيسبوك، إنستغرام، وتويتر لاستهداف الفلاحين والمزارعين في المناطق المختلفة.
- تحسين ظهور التطبيق عبر محركات البحث باستخدام إعلانات جوجل عند البحث عن حلول زراعية.
- التسويق عبر البريد الإلكتروني: إرسال محتوى تعليمي وتوعوي مخصص للفلاحين والمشتركين.

2. **التسويق عبر الهواتف المحمولة**

- إرسال إشعارات فورية للمستخدمين حول نصائح زراعية موسمية، تحديثات الطقس، والعروض الخاصة بالتطبيق.
- التعاون مع تطبيقات محلية أخرى لعرض إعلانات داخل تطبيقاتهم بهدف توسيع قاعدة المستخدمين.

3. **التسويق التقليدي**

- نشر إعلانات في المراكز الزراعية، الأسواق الفلاحية، والندوات الزراعية لتعريف الفلاحين بالتطبيق.
- تقديم عروض تعريفية مثل فترة تجريبية مجانية أو خصومات على الاشتراكات لجذب الفلاحين لتجربة التطبيق.

4.4.3 تحسن تجربة المستخدم

- تطوير واجهة بسيطة وواضحة تتيح للفلاحين الوصول السريع إلى المعلومات والخدمات.
- تحسين تجربة المستخدم باستمرار من خلال جمع الملاحظات والاستجابة لاحتياجاتهم الفعلية.

5.4.3 الشراكات الاستراتيجية

1. **التعاون مع الشركات والمؤسسات الزراعية المحلية**

- إقامة شراكات مع شركات توزيع المبيدات والأسمدة، ومزودي المعدات الزراعية لتعزيز توفير خدمات متكاملة للفلاحين.
- التفاوض على تقديم عروض وخصومات حصرية للمزارعين الذين يستخدمون التطبيق.

2. **التعاون مع المؤسسات التعليمية ومراكز البحث الزراعي**

- عقد اتفاقيات مع الجامعات ومراكز البحث الزراعي لتوفير محتوى علمي وإرشادات مبنية على أحدث الدراسات.
- تقديم برامج تدريبية وورش عمل للفلاحين عبر التطبيق بالتعاون مع الخبراء والمختصين.

الفصل 4

خطة الإنتاج والتنظيم

1.4 خطة الإنتاج

1.1.4 التخطيط والتحضير

1. **تحديد المتطلبات والأهداف**

- **اجتماعات تعريفية:** عقد اجتماعات مع الأطراف المعنية (التطوير، التسويق، الإنتاج، الإدارة) لتحديد أهداف المشروع وتوقعات المستخدمين.
- **تحليل الاحتياجات:** دراسة خصائص الأراضي وأنواع المحاصيل الأكثر شيوعًا، والتحديات التي تواجه الفلاح في اتخاذ القرار.
- **تحديد الخصائص:** تحديد الميزات الأساسية للنظام مثل جمع البيانات، تحليلها، وتقديم توصيات باستخدام الذكاء الاصطناعي.

2. **جمع البيانات**

- **تحديد المصادر:** تحديد مصادر البيانات التكميلية مثل مراكز الأبحاث الزراعية، قواعد بيانات الإنتاج النباتي، والخرائط الجغرافية.
- **تنظيم البيانات:** تنظيم ترتيب النتائج داخل بنية بيانات منظمة لتسهيل معالجتها لاحقًا.

3. **رؤية الجهاز**

- **المتطلبات التقنية للجهاز:** تحديد المستشعرات، حجم الجهاز وشكله التقريبي، مصدر الطاقة (بطارية، طاقة شمسية)، طريقة الاتصال مع التطبيق (بلوتوث، واي فاي).

2.1.4 التصميم والتطوير

1. **تصميم قاعدة البيانات**

- **هيكلة قاعدة البيانات:** تصميم هيكل قاعدة البيانات بما يتناسب مع نوعية بيانات التربة والنتائج المقترحة.
- **إنشاء قاعدة البيانات:** استخدام أدوات إدارة قواعد البيانات لإنشاء قاعدة بيانات مرنة وآمنة.

2. **تطوير النموذج الأولي**

- **تصميم الواجهات الأولية:** تصميم واجهات المستخدم الأولية وتدفقات العمل التي تعرض نتائج التحليل.
- **اختبار النموذج الأولي:** اختبار النموذج الأولي داخليًا لجمع الملاحظات وتحسين التصميم.

3. **تطوير البرمجيات**

- **برمجة التطبيق:** البدء ببرمجة التطبيق باستخدام لغات البرمجة المناسبة (مثل Java، Python، React Native).
- **دمج قاعدة البيانات:** ربط التطبيق بقاعدة البيانات لضمان تدفق البيانات بشكل صحيح.
- **اختبارات الوحدات:** إجراء اختبارات أولية على وحدات التطبيق للتأكد من عملها بشكل صحيح.

4. **تصميم واجهة المستخدم**

- **تطوير واجهة المستخدم:** تطوير واجهة مستخدم جذابة وسهلة الاستخدام، مناسبة للفلاحين.
- **تجربة المستخدم:** تحسين تجربة المستخدم من خلال إجراء اختبارات.

5. **تطوير خوارزميات الذكاء الاصطناعي**

- **تطوير الخوارزميات:** تصميم خوارزميات لتحليل بيانات التربة وتقديم التوصيات المناسبة.
- **تدريب النماذج:** تدريب نماذج الذكاء الاصطناعي باستخدام البيانات المجمعة.
- **اختبار النماذج:** اختبار النماذج لضمان دقتها وفعاليتها.

6. **اختبار التطبيق**

- **اختبار النظام بالكامل:** إجراء اختبارات شاملة للتطبيق للتحقق من أدائه واستقراره.
- **اختبار القبول:** إجراء اختبار قبول المستخدم للتأكد من أن التطبيق يلبي المتطلبات والاحتياجات.

7. **تصميم وتطوير الجهاز**

- **المكونات المادية للجها:** يتكون الجهاز من مجموعة متكاملة من المستشعرات الدقيقة، مثل مستشعر 7 in 1 NPK بالإضافة إلى ذلك، يحتوي الجهاز على وحدة معالجة مركزية صغيرة لمعالجة البيانات الأولية، ووحدة اتصال لاسلكي لتبادل البيانات مع التطبيق.
- **آلية عمل الجهاز:** تقوم المستشعرات في الجهاز بجمع البيانات من المحيط، مثل درجة الحرارة والرطوبة وجودة الهواء. ثم، تعالج وحدة المعالجة المركزية هذه البيانات الأولية وتحولها إلى معلومات رقمية. وأخيراً، ترسل وحدة الاتصال اللاسلكي هذه المعلومات إلى التطبيق على هاتفك.
- **تكامل الجهاز مع التطبيق:**يستقبل التطبيق البيانات المرسلة من الجهاز لاسلكياً، ثم يقوم بمعالجتها وتحليلها باستخدام خوارزميات ذكية. بعد ذلك، يعرض التطبيق النتائج على شكل رسوم بيانية وجداول تفاعلية، أو ربما تنبيهات ذكية إذا لزم الأمر.

3.1.4 الإطلاق والتسويق

1. **إعداد استراتيجية التسويق**

- **وضع خطة تسويقية:** إعداد خطة تسويقية تتضمن الإعلان عبر الإنترنت، وسائل التواصل الاجتماعي، والتسويق التقليدي.
- **تصميم المواد التسويقية:** تصميم المواد الترويجية مثل الإعلانات، الفيديوهات التوضيحية، والملصقات.

2. **إطلاق النسخة التجريبية**

- **إطلاق بيتا:** إطلاق نسخة تجريبية من التطبيق لجمع التغذية الراجعة من المستخدمين.
- **جمع الملاحظات:** جمع وتحليل ملاحظات المستخدمين لتحسين التطبيق.

3. **إطلاق النسخة النهائية**

- **تحسين التطبيق:** تنفيذ التحسينات النهائية بناءً على ملاحظات النسخة التجريبية.
- **النشر على المتاجر:** نشر التطبيق على متاجر التطبيقات مثل Google Play وApp Store.
- **إطلاق حملة تسويقية:** تنفيذ حملة تسويقية شاملة للترويج للتطبيق وجذب المستخدمين.

4.1.4 الدعم والتحسين المستمر

1. **دعم العملاء**

- **تشكيل فريق دعم فني متخصص:** سيتم تأسيس فريق دعم متكامل يختص بالرد على استفسارات المستخدمين، وحل المشاكل التقنية التي قد تواجههم أثناء استخدام التطبيق أو وحدة الاستشعار.

- **توفير قنوات دعم متعددة:** دعم فني عبر الهاتف، البريد الإلكتروني، والدردشة المباشرة لضمان سرعة الاستجابة وجودة الخدمة.

2. **التحسينات والتحديثات**

- **جمع الملاحظات بشكل دوري:** الاستمرار في استقبال ملاحظات المستخدمين عبر استبيانات ومراجعات لضمان تلبية احتياجاتهم وتطوير تجربة الاستخدام.
- **إطلاق تحديثات منتظمة:** إصدار تحديثات تضيف ميزات جديدة وتحسن أداء التطبيق وخوارزميات الذكاء الاصطناعي بما يتناسب مع تطورات المجال ومتطلبات المستخدمين.
- **رصد وتحليل الأداء:** متابعة مؤشرات أداء التطبيق ووحدة الاستشعار باستمرار لاكتشاف النقاط التي تحتاج إلى تحسين وتطوير.

3. **صيانة وحدة الاستشعار**

- **خدمات الصيانة الدورية:** تقديم صيانة منتظمة لوحدة الاستشعار لضمان استمرار عملها بكفاءة في البيئات الزراعية المختلفة.
- **فحص وتنظيف المستشعرات:** التأكد من سلامة جميع المستشعرات وتنظيفها واستبدال الأجزاء التالفة عند الحاجة، مما يضمن دقة البيانات واستمرارية النظام دون انقطاع.

5.1.4 التقييم والاستدامة

1. **تقييم الأداء**

- **مراجعة دورية للأداء:** سيتم إجراء تقييم دوري لأداء النظام من خلال مؤشرات دقيقة مثل دقة التوصيات التي يقدمها الذكاء الاصطناعي، مدى رضا المزارعين، وعدد مرات استخدام التطبيق فعليًا في الميدان.
- **تحليل الأثر والعوائد:** تحليل العوائد الناتجة من تبني التوصيات، ومدى تحسن الإنتاجية الزراعية لدى المستخدمين، لقياس فعالية الجهاز والتطبيق في تقديم قرارات زراعية ذكية بناءً على بيانات التربة.

2. **استراتيجيات الاستدامة**

- **تطوير الميزات المستقبلية:** سيتم التركيز على إضافة مزايا متقدمة مثل دعم أنواع محاصيل جديدة، تقديم توصيات تسميد وري مخصصة، وتكامل أوسع مع منصات الطقس أو البيانات المناخية.
- **توسيع الشراكات:** بناء شراكات مع مراكز أبحاث زراعية، مؤسسات تمويل زراعي، وهيئات حكومية لدعم اعتماد التكنولوجيا وتحقيق انتشار أوسع في المناطق الفلاحية المستهدفة.

6.1.4 الجدول الزمني

المرحلة	النشاط	المدة الزمنية (الأسابيع)
التخطيط والتحضير	تحديد المتطلبات، جمع البيانات، وضع خطة المشروع	12
التصميم والتطوير	تصميم قاعدة البيانات، تطوير النموذج الأولي، برمجة التطبيق، تصميم واجهة المستخدم، تطوير خوارزميات الذكاء الاصطناعي، اختبار التطبيق	25
الإطلاق والتسويق	إعداد استراتيجية التسويق، إطلاق النسخة التجريبية، إطلاق النسخة النهائية، تنفيذ حملة تسويقية	8
الدعم والتحسين المستمر	دعم العملاء، جمع الملاحظات، التحديثات الدورية	مستمر
إجمالي المدة الزمنية	//	**45 أسبوعًا**

جدول 4.1: الجدول الزمني التفصيلي لإطلاق التطبيق

الفصل 5

الخطة التسويقية

1.5 الخطة المالية للمشروع ونموذج العمل التجاري

1.1.5 التكاليف والرسوم

سيدعم المشروع عدة تكاليف ورسوم على النحو التالي:

1. تكاليف لوجيستية

- يُقدَّر إيجار مقر المشروع بـ .40,000.00 دج/شهريًا، أي ما يعادل 480,000.00 دج/سنة.
- تُقدّر رواتب الموظفين بـ 70,000.00 دج/شهريًا، أي ما يعادل 840,000.00 دج/سنة.
- تطوير البرمجيات والتصميم يشمل هذا البند تكلفة تطوير النظام البرمجي الذي يقوم بأخذ البيانات وتحليلها، بالإضافة إلى تصميم واجهة المستخدم الخاصة بالتطبيق. تُقدّر التكلفة الإجمالية بـ 600,000.00 دج.
- تصميم وتصنيع جهاز ذكي يشمل هذا البند تكلفة شراء المكونات والمواد اللازمة لتصميم وتجميع الجهاز الذكي. تُقدّر التكلفة الإجمالية بـ 00.1,000,000 دج، مع إمكانية اختلافها حسب نوع وجودة المكونات المستخدمة.

2. تكاليف التسويق والدعاية

- حملات التسويق الرقمي: تكلفة الإعلانات عبر الإنترنت ومواقع التواصل الاجتماعي والترويج للتطبيق. يتوقف تقدير هذه التكاليف على نطاق وحجم الحملة ومدى التغطية المطلوبة، وتُقدّر بـ .200,000.00 دج.
- إعداد المواد الترويجية: تكلفة إنشاء المواد الترويجية مثل الفيديوهات التعريفية والمطبوعات الدعائية، وتُقدّر بـ .20,000.00 دج.

- العلاقات العامة: تكلفة الترويج والتسويق من خلال التعاون مع وسائل الإعلام والنشرات الصحفية والفعاليات العامة، وتُقدّر بـ .100,000.00 دج، مع التزامنا بالمشاركة في جميع الفعاليات المتعلقة بالمجال لتعزيز انتشار المشروع.

3. تكاليف التشغيل و الصيانة

- **شراء وصيانة الأجهزة والمعدات:** تشمل تكلفة شراء الأجهزة اللازمة لتشغيل التطبيق وأجهزة الكمبيوتر، بالإضافة إلى تكاليف الصيانة الدورية. تُقدّر التكلفة الإجمالية لهذا البند بـ .300,000.00 دج.

4. تكاليف الإدارة

- تجهيزات مكتبية: تُقدّر بـ 400,000.00 دج.

- التشغيل اليومي للمقر: تشمل تكاليف الكهرباء والمياه لمكتب الشركة، بالإضافة إلى تكاليف الإنترنت والهواتف واللوازم الأخرى. تُقدّر تكلفة هذا البند بـ 450,000.00 دج سنويًا.

5. تكاليف قانونية

- الاستشارات القانونية: تكلفة الحصول على استشارات قانونية للمساعدة في إعداد الاتفاقيات ومعالجة القضايا القانونية المحتملة. تُقدّر بـ 50,000.00 دج سنويًا.

2.5 تمويل المشروع

سيتم تمويل المشروع من قبل صندوق الشركات الناشئة الجزائري سداد الاعتمادات والقروض سيتم تنفيدها على المدى الطويل والذي سيتم التفاوض عليه مع المنظمة المعنية.

جدول 5.1: جدول التكاليف والاستثمار (التوقعات)

الشرط	التكلفة
تطوير البرمجيات والتصميم	600,000 دج
تصميم وتصنيع الجهاز الذكي	1,000,000 دج
حملات التسويق الرقمي	200,000 دج
إعداد المواد الترويجية	20,000 دج
العلاقات العامة والترويج	100,000 دج
أجهزة التشغيل والصيانة	300,000 دج
تجهيزات مكتبية	400,000 دج
التشغيل اليومي (كهرباء، ماء، إنترنت، هواتف...)	450,000 دج
عباء صندوق التأمين CNAS	75,600 دج/سنة
الاستشارات القانونية	50,000 دج
إيجار مقر العمل 40,000 دج/شهر	480,000 دج/سنة
رواتب الموظفين 70,000 دج/شهر	840,000 دج/سنة
المجموع التقديري	**4,515,600 دج**

3.5 رقم الأعمال

1.3.5 النظرة التفاؤلية

وضح الجدول التالي تطور رقم أعمال الشركة عبر عدة سنوات، حيث يعتمد نموذج الإيرادات على بيع جهاز ذكي في السنة الأولى لكل مشترك جديد، ثم تحصيل اشتراك سنوي ثابت في السنوات اللاحقة. يُظهر الجدول نموًا تدريجيًا في عدد المشتركين، مما أدى إلى زيادة في مبيعات الأجهزة خلال السنوات الأولى، تلتها زيادة ملحوظة في إيرادات الاشتراكات السنوية. يعكس هذا النمو نجاح استراتيجية الشركة في استقطاب عملاء جدد مع الحفاظ على المشتركين الحاليين، مما يعزز من استدامة الإيرادات واستقرار الأداء المالي على المدى الطويل.

جدول 5.2: تفاصيل الاشتراكات والإيرادات (النظرة المتفائلة)

العام	عدد الاشتراكات	عدد الأجهزة الجديدة (زيادة الاشتراكات)	عدد الاشتراكات السنوية	سعر الجهاز (دج)	سعر الاشتراك السنوي (دج)	الإيرادات الإجمالية (دج)
N	**50**	**50**	0	80,000	0	$50 \times 80,000 =$ **4,800,000**
N+1	**120**	$120 - 50 =$ **70**	50	80,000	9,000	$(70 \times 80{,}000) + (50 \times 9{,}000) =$ **6,950,000**
N+2	**220**	$220 - 120 =$ **100**	120	80,000	9,000	$(100 \times 80{,}000) + (120 \times 9{,}000) =$ **9,080,000**
N+3	**350**	$350 - 220 =$ **130**	220	80,000	9,000	$(130 \times 80{,}000) + (220 \times 9{,}000) =$ **12,380,000**

2.3.5 النظرة التشاؤمية

جدول 5.3: تفاصيل الاشتراكات والإيرادات (النظرة التشاؤمية)

العام	عدد الاشتراكات	عدد الأجهزة الجديدة (زيادة الاشتراكات)	عدد الاشتراكات السنوية	سعر الجهاز (دج)	سعر الاشتراك السنوي (دج)	الإيرادات الإجمالية (دج)
N	**10**	**10**	0	80,000	0	$10 \times 80,000 =$ **800,000**
N+1	**25**	$25 - 10 =$ **15**	10	80,000	9,000	$(15 \times 80,000) + (10 \times 9,000) =$ **1,290,000**
N+2	**60**	$60 - 25 =$ **35**	25	80,000	9,000	$(20 \times 80,000) + (40 \times 9,000) =$ **3,025,000**
N+3	**100**	$100 - 60 =$ **40**	60	80,000	9,000	$(40 \times 80,000) + (60 \times 9,000) =$ **3,740,000**

4.5 جداول حسابات النتائج

جدول 5.4: المصاريف الشهرية خلال السنة الأولى (الجزء الأول)

المصاريف / الأشهر	1	2	3	4	5	6
تطوير التطبيق	30,000	30,000	30,000	30,000	30,000	30,000
تصنيع الجهاز	50,000	50,000	50,000	50,000	50,000	50,000
تكاليف التسويق والدعاية	20,000	20,000	20,000	20,000	20,000	20,000
معدات المكتبة	400,000	/	/	/	/	/
صندوق أعباء التأمين CNAS	6,300	6,300	6,300	6,300	6,300	6,300
تأمين المعدات	8,000	8,000	8,000	8,000	8,000	8,000
التكاليف الإدارية	4,166	4,166	4,166	4,166	4,166	4,166
رواتب العمال	70,000	70,000	70,000	70,000	70,000	70,000
تكلفة الإيجار	40,000	40,000	40,000	40,000	40,000	40,000
الرسوم الخارجية	37,500	37,500	37,500	37,500	37,500	37,500
المحاسبون القانونيون	6,830	6,830	6,830	6,830	6,830	6,830
مجموع النفقات	672,796	272,796	272,796	272,796	272,796	272,796

جدول 5.5: المصاريف الشهرية خلال السنة الأولى (الجزء الثاني)

المصاريف	7	8	9	10	11	12	المجموع
تطوير التطبيق	30,000	30,000	30,000	30,000	30,000	30,000	360,000
تصنيع الجهاز	50,000	50,000	50,000	50,000	50,000	50,000	600,000
تكاليف التسويق والدعاية	20,000	20,000	20,000	20,000	20,000	20,000	240,000
معدات المكتبة	/	/	/	/	/	/	400,000
صندوق أعباء التأمين CNAS	6,300	6,300	6,300	6,300	6,300	6,300	75,600
تأمين المعدات	8,000	8,000	8,000	8,000	8,000	8,000	96,000
التكاليف الإدارية	4,166	4,166	4,166	4,166	4,166	4,166	49,992
رواتب العمال	70,000	70,000	70,000	70,000	70,000	70,000	840,000
تكلفة الإيجار	40,000	40,000	40,000	40,000	40,000	40,000	480,000
الرسوم الخارجية	37,500	37,500	37,500	37,500	37,500	37,500	450,000
المحاسبون القانونيون	6,830	6,830	6,830	6,830	6,830	6,830	81,960
مجموع النفقات	272,796	272,796	272,796	272,796	272,796	272,796	3,233,552

5.5 خطة الخزينة

خطة الخزينة التي سينتهجها مشروعنا هي كالتالي في السنوات المقبلة موضحة في الجداول الموالية:

جدول 5.6: الإيرادات والنفقات المتوقعة خلال السنة الأولى حسب كل شهر لنشاطنا (الجزء الأول)

الأشهر	1	2	3	4	5	6
البيع	309,800	309,800	309,800	309,800	309,800	309,800
المبيعات صافي	309,800	309,800	309,800	309,800	309,800	309,800
المصاريف	272,796	272,796	272,796	272,796	272,796	272,796
هامش الربح	37,004	37,004	37,004	37,004	37,004	37,004
الربح التشغيلي	37,004	37,004	37,004	37,004	37,004	37,004
إيرادات فوائد	0	0	0	0	0	0
الربح قبل الضريبة	37,004	37,004	37,004	37,004	37,004	37,004
رسوم ضريبية	0	0	0	0	0	0
صافي الإيرادات	37,004	37,004	37,004	37,004	37,004	37,004

جدول 5.7: الإيرادات والنفقات المتوقعة خلال السنة الأولى حسب كل شهر لنشاطنا (الجزء الثاني)

الأشهر	7	8	9	10	11	12	المجموع
البيع	309,800	309,800	309,800	309,800	309,800	309,800	3,717,600
المبيعات صافي	309,800	309,800	309,800	309,800	309,800	309,800	3,717,600
المصاريف	272,796	272,796	272,796	272,796	272,796	272,796	3,233,552
هامش الربح	37,004	37,004	37,004	37,004	37,004	37,004	484,048
الربح التشغيلي	37,004	37,004	37,004	37,004	37,004	37,004	484,048
إيرادات فوائد	0	0	0	0	0	0	0
الربح قبل الضريبة	37,004	37,004	37,004	37,004	37,004	37,004	484,048
رسوم ضريبية	0	0	0	0	0	0	0
صافي الإيرادات	37,004	37,004	37,004	37,004	37,004	37,004	484,048

جدول 5.8: الإيرادات والنفقات المتوقعة خلال السنة الخامسة حسب كل شهر لنشاطنا (الجزء الأول)

الأشهر	1	2	3	4	5	6
البيع	1,930,983	1,930,983	1,930,983	1,930,983	1,930,983	1,930,983
صافي المبيعات	1,930,983	1,930,983	1,930,983	1,930,983	1,930,983	1,930,983
المصاريف	1,766,289	1,766,289	1,766,289	1,766,289	1,766,289	1,766,289
هامش الربح	164,694	164,694	164,694	164,694	164,694	164,694
الربح التشغيلي	164,694	164,694	164,694	164,694	164,694	164,694
إيرادات فوائد	0	0	0	0	0	0
الربح قبل الضريبة	164,694	164,694	164,694	164,694	164,694	164,694
الضرائب	0	0	0	0	0	0
صافي الإيرادات	164,694	164,694	164,694	164,694	164,694	164,694

جدول 5.9: الإيرادات والنفقات المتوقعة خلال السنة الخامسة حسب كل شهر لنشاطنا (الجزء الثاني)

الأشهر	7	8	9	10	11	12	المجموع
البيع	1,930,983	1,930,983	1,930,983	1,930,983	1,930,983	1,930,983	23,171,796
صافي المبيعات	1,930,983	1,930,983	1,930,983	1,930,983	1,930,983	1,930,983	23,171,796
المصاريف	1,766,289	1,766,289	1,766,289	1,766,289	1,766,289	1,766,289	21,195,468
هامش الربح	164,694	164,694	164,694	164,694	164,694	164,694	1,976,328
الربح التشغيلي	164,694	164,694	164,694	164,694	164,694	164,694	1,976,328
إيرادات فوائد	0	0	0	0	0	0	0
الربح قبل الضريبة	164,694	164,694	164,694	164,694	164,694	164,694	1,976,328
الضرائب	0	0	0	0	0	0	0
صافي الإيرادات	164,694	164,694	164,694	164,694	164,694	164,694	1,976,328

الفصل 6

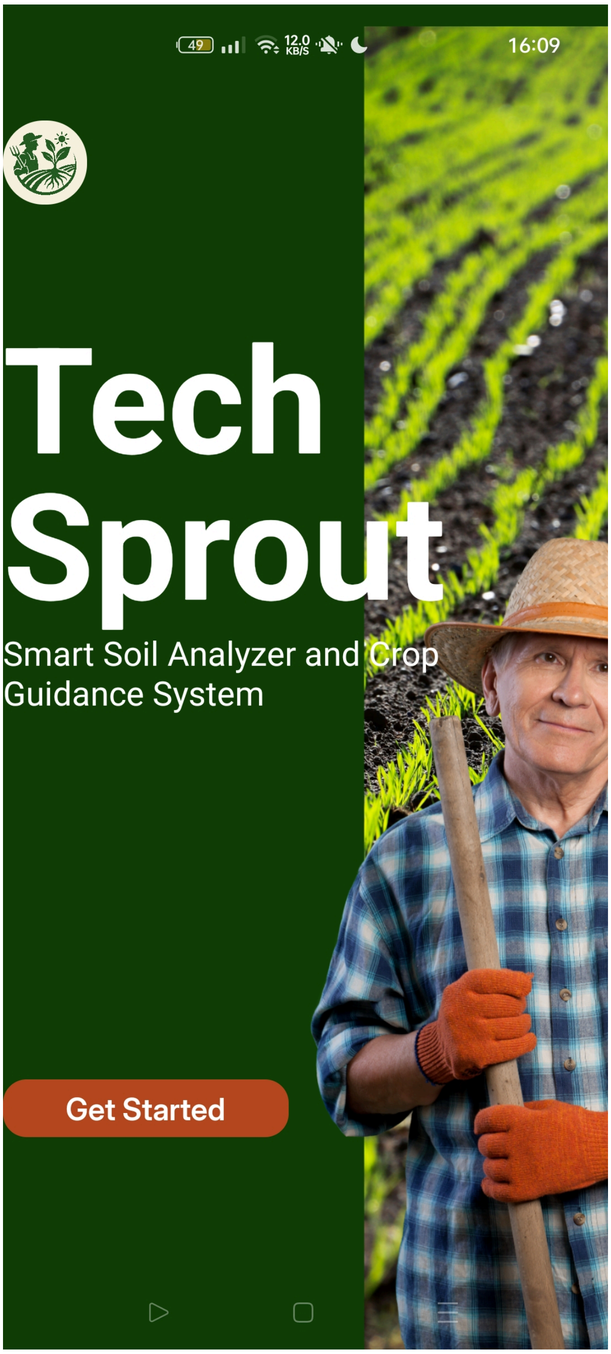
االنموذج التجريبي الأولي

1.6 واجهة المستخدم

في ما يلي، سنعرض بعض لقطات الشاشة لواجهات المستخدم التي تم تطويرها لتطبيقنا.

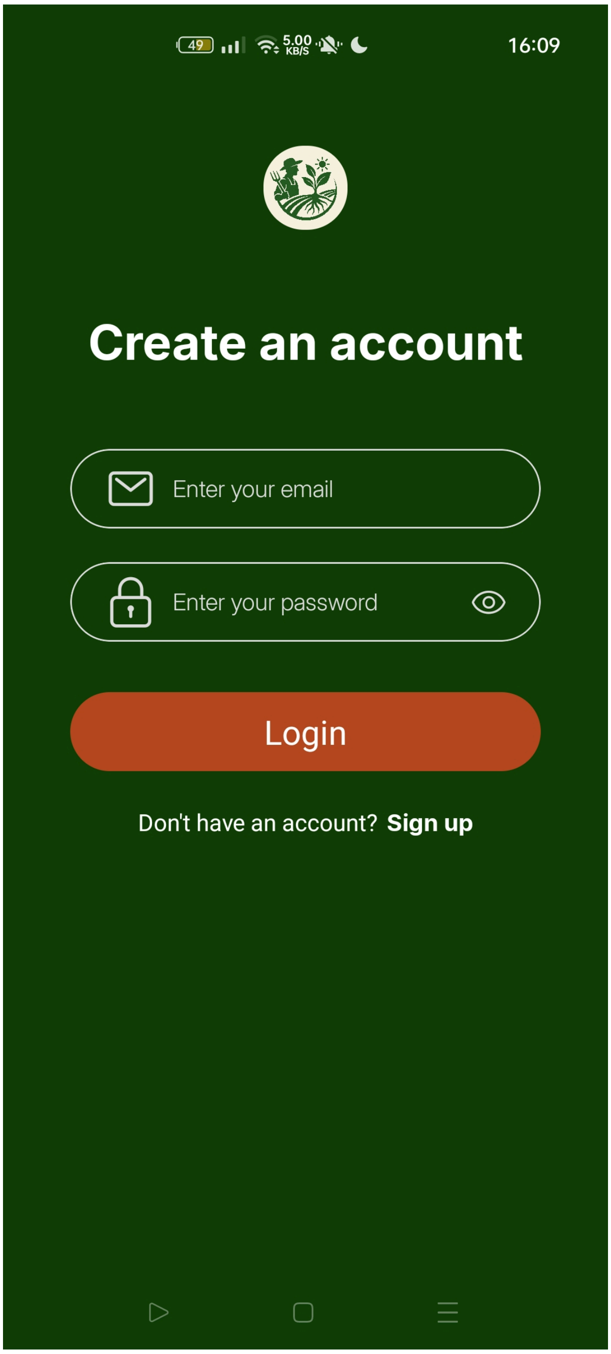
2.6 صفحة البدء

تعرض هذه الصورة الشاشة الأولية التي تظهر عند فتح التطبيق لأول مرة، حيث يتم الترحيب بالمستخدم وتقديم مقدمة سريعة حول خصائص التطبيق باعتباره "محلل التربة الذكي ونظام توجيه المحاصيل"، وتتضمن زر "البدء" للانتقال إلى الخطوة التالية.



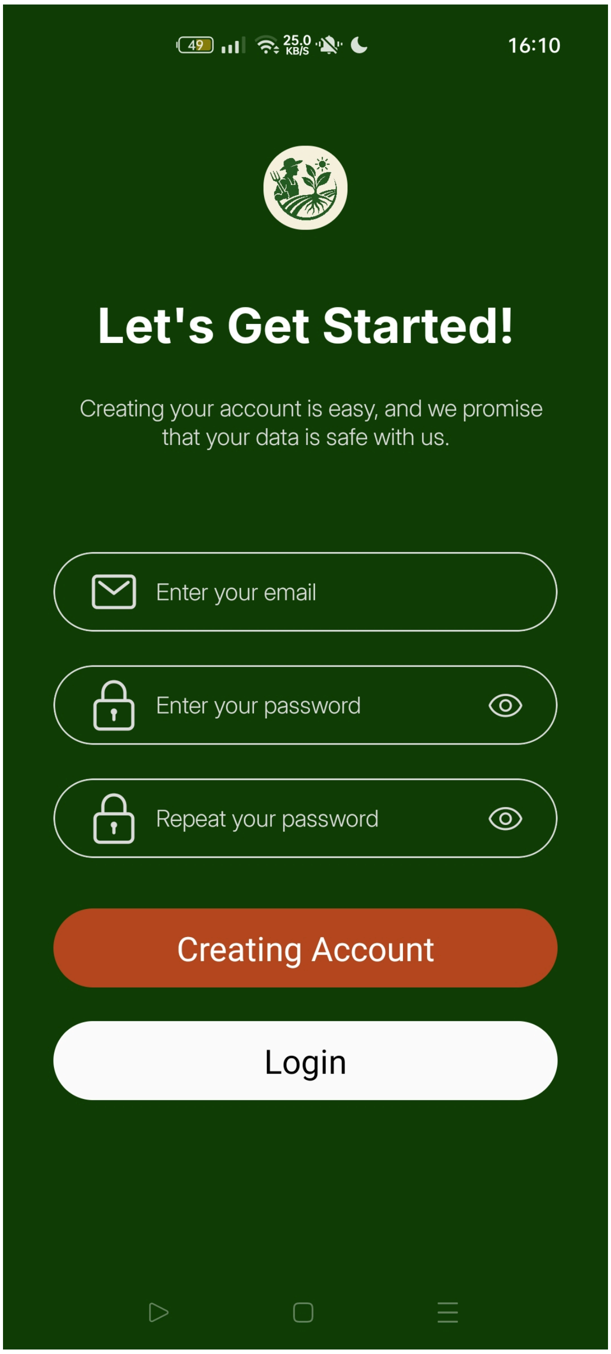
شكل 6.1: واجهة البدء

3.6 صفحة المصادقة



شكل 6.2: واجهة تسجيل الدخول

تعرض هذه الصورة واجهة تسجيل الدخول، التي تسمح للمستخدمين الحاليين بالوصول إلى حساباتهم عبر إدخال البريد الإلكتروني وكلمة المرور، بطريقة بسيطة وسلسة.

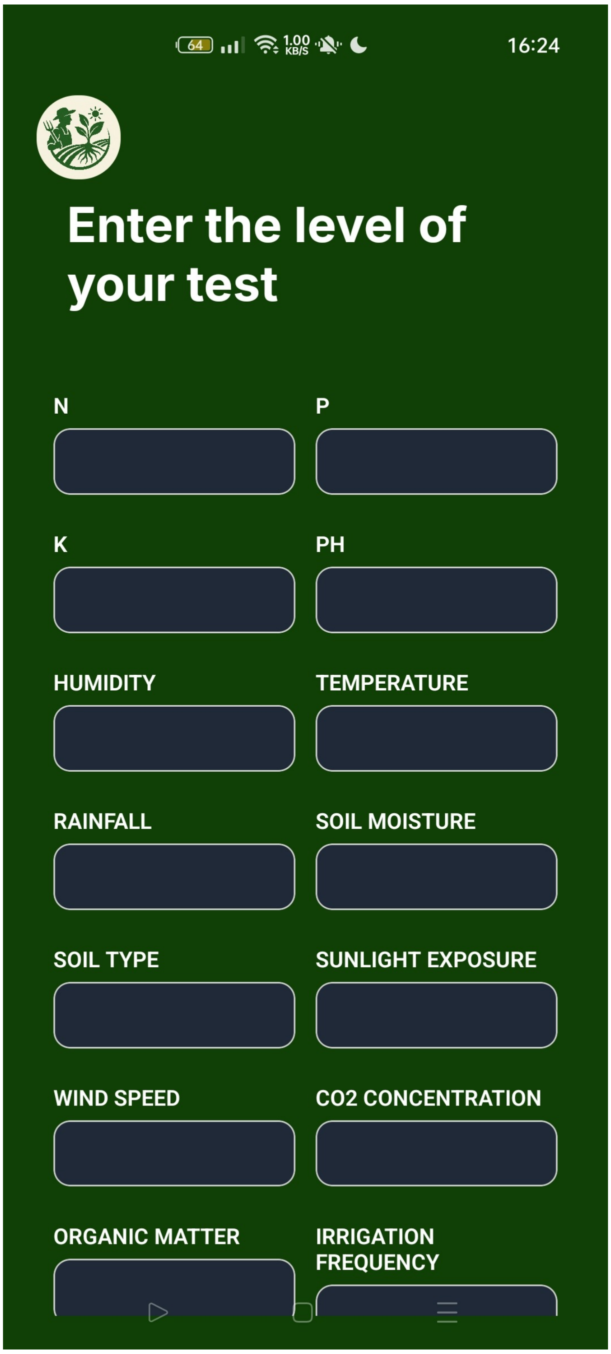


شكل 6.3: واجهة إنشاء الحساب

تصور هذه الصورة واجهة التسجيل للمستخدمين الجدد، حيث تتضمن خانات لإدخال البريد الإلكتروني وكلمة المرور وتأكيدها، مما يضمن عملية تسجيل سهلة وآمنة.

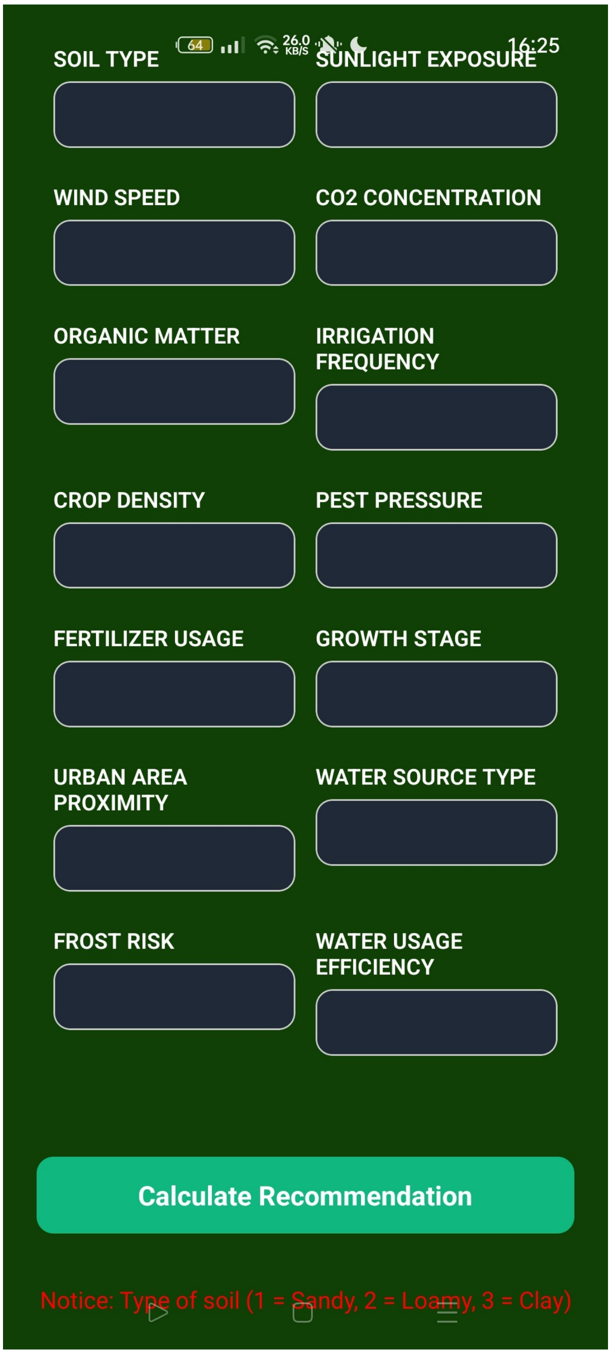
4.6 الصفحة الرئيسية

1.4.6 إدخال البيانات



شكل 6.4: واجهة إدخال معلمات التربة (الجزء 1)

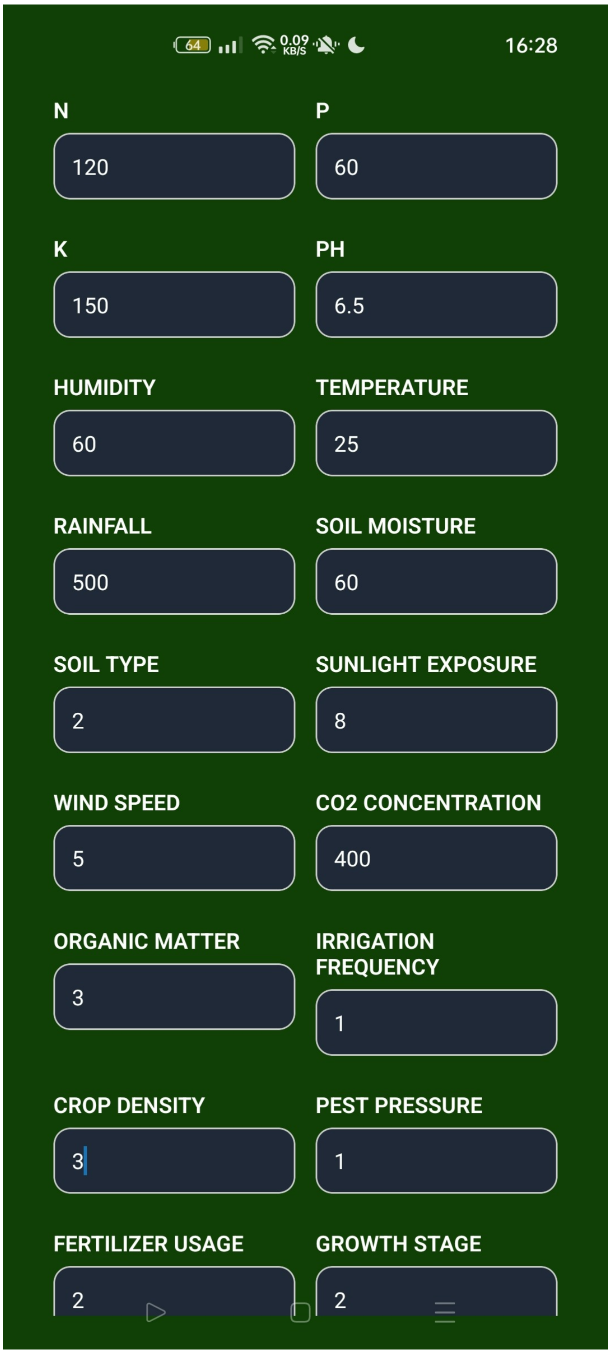
تعرض هذه الصورة الجزء الأول من واجهة إدخال البيانات، حيث يملأ المستخدم خصائص التربة مثل النيتروجين (N)، الفوسفور (P)، البوتاسيوم (K)، درجة الحرارة، الرطوبة، ودرجة الحموضة.



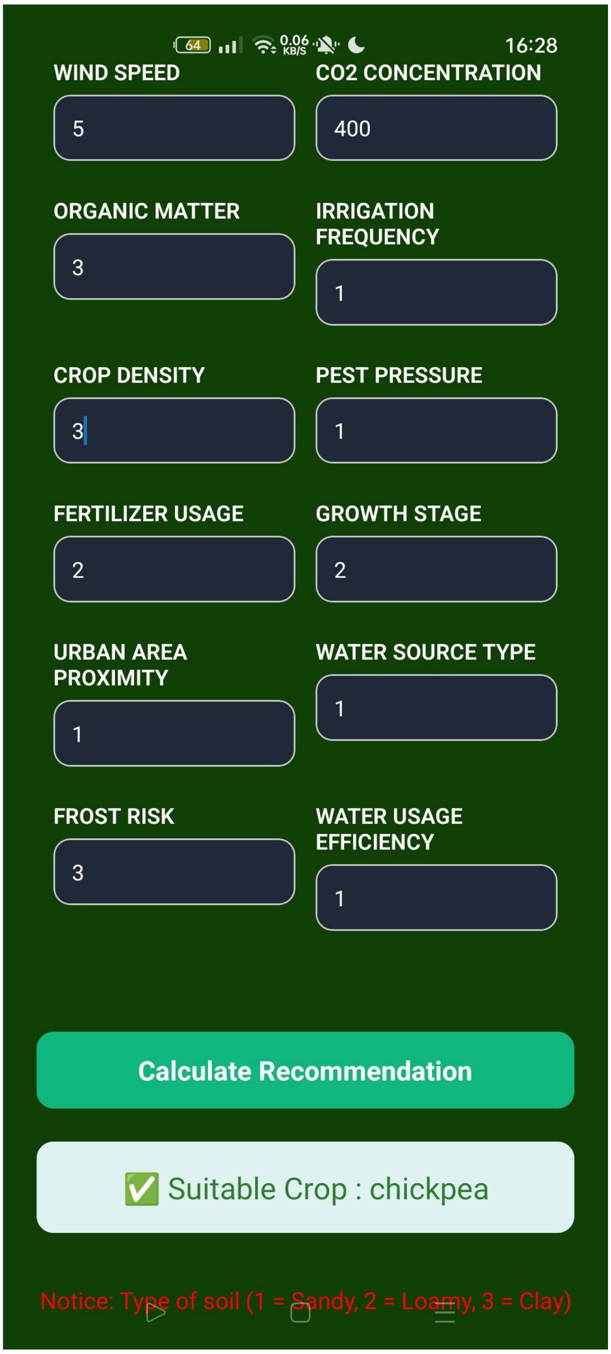
شكل 6.5: واجهة إدخال معلمات التربة (الجزء 2)

تعرض هذه الصورة الجزء الثاني من واجهة إدخال البيانات، والتي تشمل معايير إضافية مثل نوع التربة، خطر الصقيع، واستخدام المياه، مع زر "احسب التوصية" لإرسال البيانات.

2.4.6 عرض النتائج



شكل 6.6: مثال على معلمات مدخلة في التطبيق

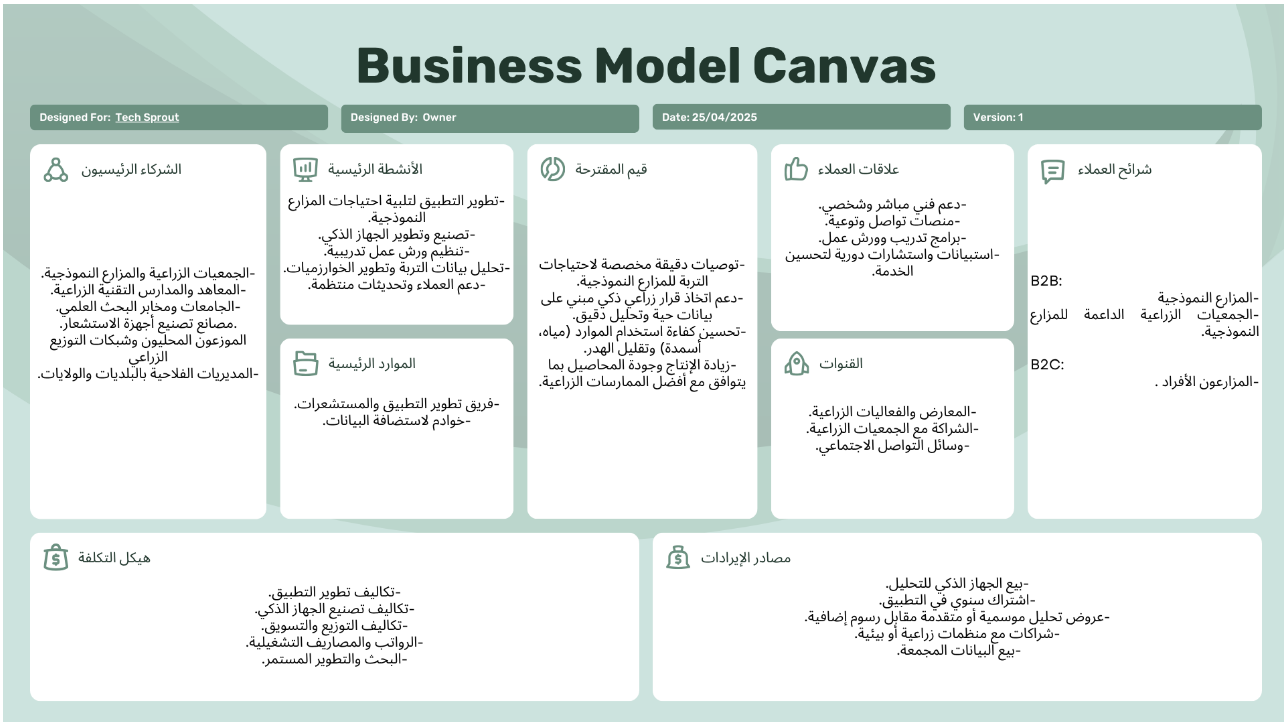


شكل 6.7: عرض التوصية

تعرض هذه الصورة شاشة النتيجة النهائية، حيث يقترح التطبيق المحصول المناسب بناءً على البيانات التي أدخلها المستخدم. في المثال المعروض، التوصية هي "الحمص" كمحصول مناسب.

الفصل 7

نموذج العمل التجاري



شكل 7.1: نموذج العمل التجاري

خلاصة عامة

في الختام، يُعد هذا المشروع خطوة أولى نحو تحقيق زراعة ذكية ومستدامة تعتمد على الذكاء الاصطناعي في اتخاذ القرار الزراعي، من خلال تحليل خصائص التربة والعوامل المناخية لتحديد أنسب محصول يمكن زراعته في أرض معينة. وقد هدفنا من خلال هذا العمل إلى تقديم أداة رقمية تساعد المزارع على اتخاذ قرارات مبنية على تحليل علمي دقيق، بعيدًا عن الاجتهادات التقليدية التي كثيرًا ما تؤدي إلى نتائج غير فعالة أو استنزاف للموارد.

ومن أبرز التحديات التي حالت دون تنفيذ بعض الجوانب العملية في هذا المشروع، هو عدم توفر المكونات الأساسية الضرورية لتصنيع الجهاز الذكي داخل السوق الجزائرية، لا سيما الحساسات الدقيقة (Sensors) والشرائح الإلكترونية المتقدمة التي تُعد عنصرًا محوريًا في النظام. هذا النقص في التجهيزات التقنية حال دون إمكانية تجسيد الجانب المادي من المشروع في هذه المرحلة، رغم الأهمية الكبرى التي يمثلها هذا الجهاز في تسهيل عملية تحليل التربة ميدانيًا. إذ كان من المخطط أن يتيح هذا الجهاز للمزارع إمكانية قياس الخصائص الحيوية للتربة بشكل مباشر، ثم إرسال تلك البيانات بشكل لحظي إلى التطبيق الذكي للحصول على توصيات فورية بشأن أنسب محصول يمكن زراعته. غير أن هذه الخطوة قد تم تأجيلها مؤقتًا، في انتظار توفر الشروط التقنية واللوجستية اللازمة لتنفيذها على أرض الواقع في مرحلة لاحقة من تطوير المشروع.

ورغم التحديات التقنية واللوجستية، فإننا نطمح مستقبلاً إلى تطوير نموذج أولي متكامل للجهاز الذكي بالتعاون مع مخابر إلكترونية مختصة، والعمل على توسيع قاعدة البيانات الزراعية من خلال جمع معلومات ميدانية من مناطق فلاحية متنوعة داخل الجزائر. كما نسعى إلى دمج خاصية التوصيات الزمنية لتحديد الفترات المثلى لزراعة كل محصول حسب المناخ المحلي، وتطوير واجهة استخدام متعددة اللغات تشمل العربية والأمازيغية والفرنسية لتوسيع دائرة الاستفادة.

ومن بين التوسعات المستقبلية المهمة التي نخطط لها، نعتزم دمج نظام ري ذكي داخل الجهاز، يعمل بناءً على تحليلات دقيقة للرطوبة ودرجة الحرارة ونوع التربة، لضبط كميات المياه المستعملة وترشيد استهلاكها بما يحقق أفضل مردودية ممكنة. كما نهدف إلى إضافة نظام للكشف المبكر عن الحرائق، يعتمد على حساسات الحرارة والدخان، لتنبيه المزارع بشكل فوري عند حدوث أي مؤشر خطر، مما يُساهم في حماية المحاصيل والمزرعة من الكوارث الفجائية وتقليل الخسائر المحتملة.

بالإضافة إلى ذلك، نخطط لربط النظام بالكامل بمنصات إنذار مبكر للمخاطر المناخية مثل الصقيع أو الجفاف، إلى جانب بناء شراكات فعالة مع مؤسسات فلاحية، مخابر بحث، وحاضنات أعمال من أجل تعزيز الجانب التجاري والتطبيقي لهذا الابتكار.

إن التزامنا بتطوير هذا الحل الذكي، رغم محدودية الموارد والتحديات، يعكس إيماننا العميق بأهمية تسخير التكنولوجيا الحديثة في خدمة الفلاحة الجزائرية. ونأمل أن يكون هذا المشروع مساهمة فعلية في بناء نموذج إنتاج زراعي أكثر ذكاءً

واستدامة، يدعم المزارع المحلي، ويُعزز من الأمن الغذائي الوطني على أسس علمية وتقنية راسخة.