

Project Report

**ML-Integrated App-Interfaced Ukulele Chord Detector**

**TEAM MEMBERS:**

**J. Venkatesh (23BEC1231),**

**S. Anirudh (23BEC1001),**

**S S V Mridula (23BEC1030), Aniruddhan. S (23BLC1159),**

**Mirunalini (23BLC1224)**

**Semester: Winter**

**Academic Year: 2024-2025**

ML-Integrated App-Interfaced Ukelele Chord Detector

**Abstract**

This project presents an innovative app-based machine learning (ML) integrated ukulele chord detector, combining hardware sensing with intelligent audio analysis. The system features pressure sensors interfaced with an Arduino microcontroller to detect basic chord finger positions(C and E), offering real-time visual feedback. Simultaneously, the mobile application records audio input from the ukulele using a built-in microphone, converts the signal to a .wav format, and processes it using a trained ML model to classify the played chord with high accuracy. The app not only displays the detected chord but also includes integrated playback of the correct chord sound for auditory verification and learning support. This dual-sensing approach—merging tactile and acoustic analysis—enhances chord recognition robustness and provides an interactive learning experience for beginners and hobbyists alike.

**Keywords:** Acoustic Analysis, Arduino, Audio Classification, Chord Detection, Interactive Music Education, Machine Learning, Mobile App, Musical Instrument Learning, Pressure Sensors, Real-time Feedback, Signal Processing, Sound Playback, Ukulele, WAV Conversion.

**1. Introduction**

The integration of artificial intelligence (AI) with interactive music technology has yielded promising tools for enhancing musical education and performance. One significant innovation is the development of systems that facilitate real-time chord detection and feedback, combining audio signal processing and tactile sensing to deliver a robust learning interface. This literature review explores the state of the art in chord recognition, tactile sensor integration, and intelligent educational applications, supporting the development of a dual-sensing ukulele chord detection system.

Chord recognition through machine learning has progressed significantly due to the application of deep learning models. Nakayama and Arai (2018) proposed a hybrid DNN-LSTM-CRF model for audio chord recognition, combining feature learning with sequential classification for improved accuracy (DNN-LSTM-CRF Model for Automatic Audio Chord Recognition). Similarly, Ito and Arai (2021) introduced a CNN-LSTM model using harmonic representations of audio data, reducing model size while maintaining performance, a critical factor for mobile deployment (Harmonic Representation for CNN-LSTM Automatic Chord Recognition). Hori et al. (2017) expanded on this by using bidirectional LSTMs in a chord recognition framework to enhance sequence modelling (Music chord recognition from audio data using bidirectional encoder-decoder LSTMs).

In parallel, the tactile dimension of instrument interaction has received growing attention. Lin et al. (2023) developed a smart hand exoskeleton that uses pressure sensors and ML for piano learning, highlighting the potential of pressure-based interaction for novice musicians (Feeling the beat: a smart hand exoskeleton for learning to play musical instruments). Wibawa and Putra (2022) demonstrated the integration of BME280 pressure sensors with Arduino to measure tactile inputs accurately, which supports the viability of similar sensor integration in the ukulele chord detector (Design of air pressure and height measuring equipment based on Arduino nano using BME280 sensor).

In terms of interactive and intelligent music interfaces, Deja et al. (2023) and Kosch et al. (2024) discussed systems that augment musical instruments to aid learning and self-expression through adaptive assistance, laying the conceptual foundation for tools like the proposed ukulele app (Intelligent Music Interfaces; Intelligent Music Interfaces). These systems represent a convergence of design thinking, pedagogy, and embedded technology.

The significance of ML-driven music education applications is further evidenced in Özbaltan’s (2024) work on a real-time chord identification app aimed at adult learners, employing CNNs and mel spectrograms for user-centric learning (Real-time chord identification application). Wang (2023) extended ML applications into classical music education, evaluating instructional quality using ML algorithms (Application of Machine Learning Technology in Classical Music Education).

The technical infrastructure for real-time audio analysis is also advancing. Shu and Anderson (2024) presented Audiosockets, a Python framework for low-latency audio stream processing, essential for mobile-based ML systems like the ukulele app (Audiosockets).

In sum, the convergence of deep learning for audio classification, microcontroller-enabled pressure sensing, and user-focused app development creates a fertile ground for systems that enhance learning and performance in music. The proposed dual-sensing ukulele chord detector builds upon this interdisciplinary body of work, aiming to offer a reliable, interactive, and pedagogically sound solution for music education, especially helpful for visually impaired people.

**2. Design andArchitecture of the proposed system**

**Hardware Circuitry:**

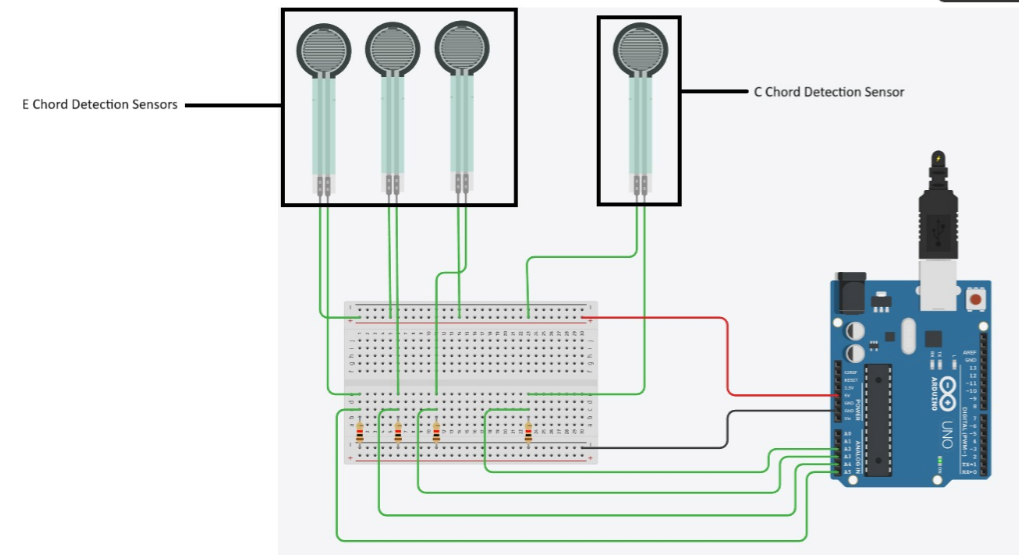


Figure 1. Hardware Integration of the proposed System

The hardware design for the ukulele chord detection system utilizes Force Sensitive Resistors (FSRs) to recognize specific chord patterns based on finger placement on the fretboard in the ukelele. This setup is engineered to detect two chords—C major and E major—using a total of four FSRs. Three of these sensors are allocated for detecting the E major chord, while one sensor is used for identifying the C major chord. The FSRs are physically attached to the ukulele frets at positions where fingers must be placed to form these chords. When a user presses a string onto a fret, the corresponding FSR detects the applied pressure. FSRs are variable resistors whose resistance decreases with increasing pressure, thereby allowing changes in the electrical signal that can be captured by the Arduino UNO's analog input pins. Each FSR is paired with a fixed 10kΩ resistor in a voltage divider configuration. The junction point of each voltage divider—where the FSR and the resistor meet—is connected to an analog input on the Arduino. Specifically, the three FSRs assigned to detect the E chord are connected to analog pins A0, A1, and A2, while the FSR for the C chord connects to analog pin A3. This analog voltage represents the magnitude of pressure applied, and the Arduino continuously reads these values to determine if a chord is being formed correctly.

Power for the entire circuit is supplied by the Arduino through its 5V and GND pins, which are connected to the positive and negative rails of a breadboard. One terminal of each FSR is connected to the 5V rail, ensuring a constant voltage input, while the other terminal connects to the analog input via the resistor to ground, thus completing the voltage divider. The resistors are all placed in the lower section of the breadboard, with each one tied between an FSR output and the ground rail to form independent channels for sensing. Jumper wires carry the resulting voltage signals to the appropriate analog pins on the Arduino. This modular layout ensures minimal interference and allows each sensor to be read independently and accurately. The FSRs are precisely mounted at the frets involved in forming C and E chords, enabling reliable detection of finger placement and pressure levels. This setup translates the physical act of playing a chord into quantifiable electrical signals, which the Arduino can interpret in real time and could be displayed appropriately on the serial monitor. The resulting detection mechanism is ideal for educational tools, interactive musical interfaces, or digital accompaniment systems, making the ukulele more accessible for learners and performers alike.

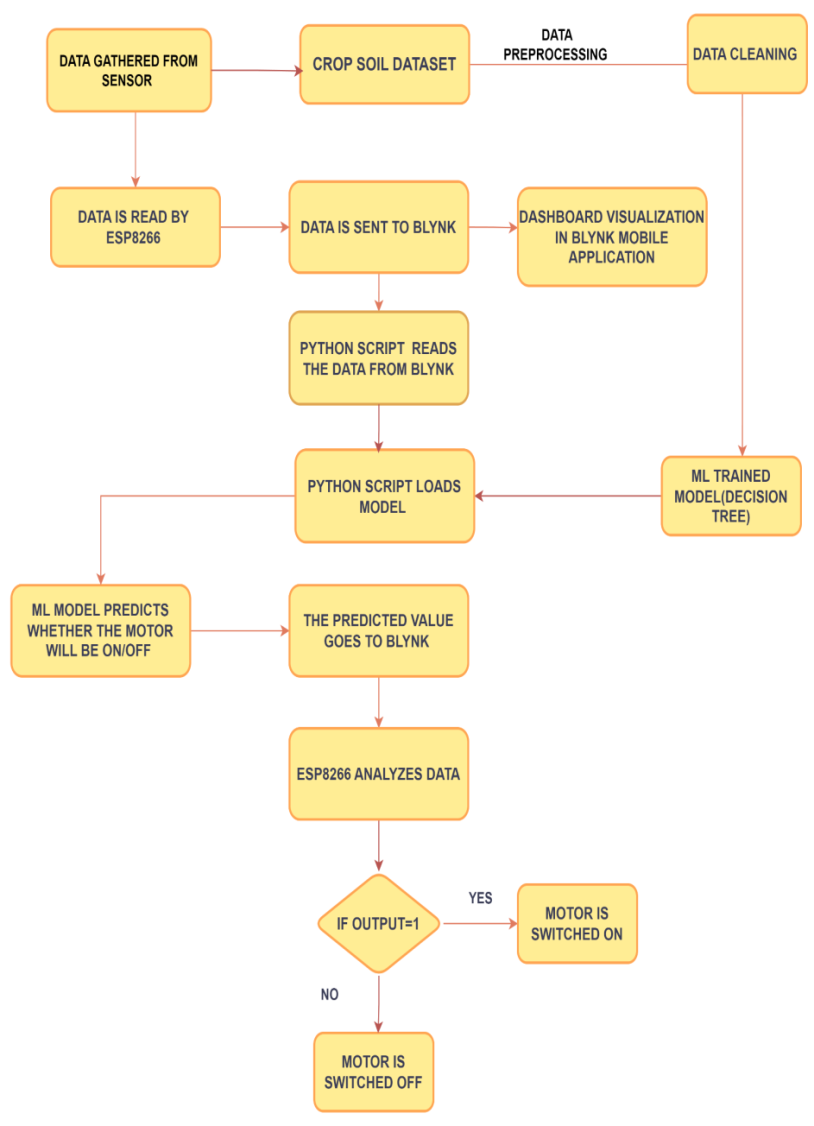
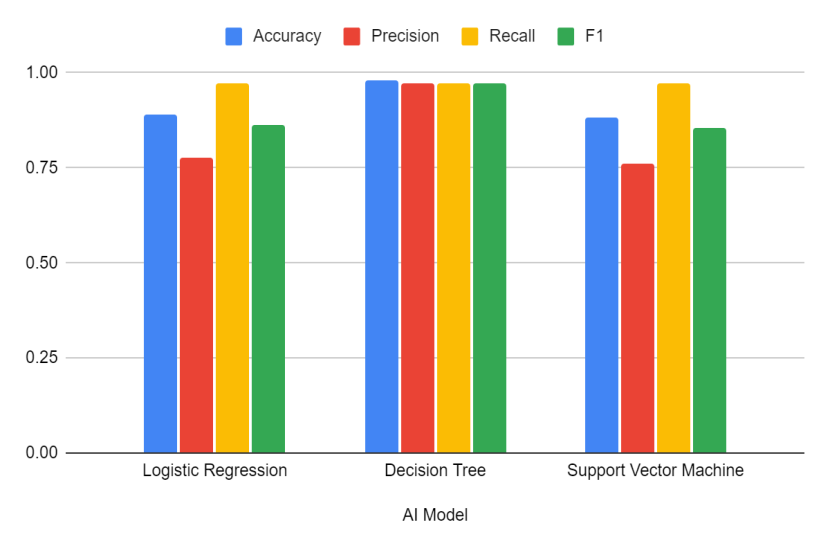


Figure 3. Flow chart of the proposed model

For example, if the soil moisture falls below a certain level, indicating that the plants need watering, the system triggers the irrigation mechanism. To ensure energy efficiency, the system may employ techniques such as duty cycling and sleep modes to minimize power consumption when the system is idle or when environmental conditions do not require immediate action. Users can set preferences, thresholds, and schedules through the Blynk application. They can also receive notifications or alerts when certain conditions are met or when manual intervention is required. The ML model takes in all the sensor data as input and gives a prediction about the state of the farm. The system operates autonomously based on the predefined parameters and user settings, reducing the need for manual intervention and ensuring efficient use of resources. If the moisture is below the threshold level, it activates the pump connected to the motor driver. Arduino Uno is used to power the motor driver as the motor requires a 5V input voltage and the ESP8266 supplies a maximum voltage of 3V. The input pins of the motor driver (L293D) are connected to the ESP8266, and the VCC and GND pins are connected to the Arduino Uno. Figure 2 shows the practical implementation of the irrigation system using the specified hardware components. At the top is an application layer with functionality like plant monitoring, irrigation scheduling, and

other recommendation services. The operation of the sensors and the steps involved in data collecting and storage at the data server for additional analytics are described in Figure 2. The pin configuration, Wi-Fi and Blynk connection, and a timer were set to periodically read sensor data. The beacon signal continuously check the connection to the Blynk platform and the timer. When the timer runs out, the sendSensor(() function is called. The sendSensor function attempts to read sensor data (FC-28 and DHT) and send it to designated virtual pins in your Blynk app if successful. Debugging messages are printed to the serial monitor to help you monitor sensor readings.

**2.1 Testing with Different ML Models**

The experimentation was carried out with various machine learning models like DT, SVM, and LR to automate irrigation based on real-time data obtained from the field and the crop information. The comparison of the effectiveness of all three tested ML models is depicted in Table 1.

Table 1. Comparison between all tested AI models.

**AI Model**

**Logistic Regression**

**Decision Tree**

**Support Vector Machine**

**Accuracy**

0.881

0.92

0.861

**Precision** **Recall** **F1**

0.75 1.0 0.857

0.85 0.94 0.89

0.72 1.0 0.837

Figure 4. Visualization of the model efficiency parameters

The Decision Tree (DT) model used in the proposed system. It takes decisions based on the readings taken from sensors placed in the field. The sensors are connected to an ESP8266, which reads data such as soil moisture levels, surrounding temperature, and humidity. The users will have to specify the crop that they are cultivating on their farm. All this data is analyzed and used to design the irrigation schedule for the particular crop that is being cultivated on the farm. The efficiency of the DT model that we used completely depends on the dataset used to train it. The dataset includes previous records of data readings from the farm, such as soil moisture, temperature, humidity, days required for crops to fully cultivate, water requirement information for different crop types, and successful irrigation practices leading to best crop growth. The model analyzes this data and maps relationships between the data, crop type, and successful irrigation events. It is a useful tool to see whether the data is good enough to makegooddecisions about irrigation.TheDTmodel is visually clear and easy to understand, so farmers can understand how the model decides when to water the crops. The model is also very flexible, as it easily allows for the accommodation of new crops. It can also adjust to different environmental conditions. The model is also computationally efficient, which allows it to make real-time decisions on low-end devices. From Table 1, it is concluded that the DT performed the best on the input data compared to the other two ML models. All four evaluation parameters, which are accuracy, precision, recall, and F1 score, of the tested ML models have been visualized in Figure 4.

Table 2. 5-fold validation for different models.

**Fold No. 0 1 2 3**

**4**

**5-fold validation for DT** 0.712871 0.860000 0.940000 0.910000

0.960000

**5-fold validation for SVM** 0.712871 0.870000 0.860000 0.830000

0.900000

**5-fold validation for LR** 0.712871 0.840000 0.800000 0.830000

0.860000

Each of the three models, DT, SVM, and LR, is validated on the dataset. The dataset is divided into five equal folds, and then each model is trained on the fourth fold and tested on the last fold. This process is repeated five times for each model. The above process helps to check the scalability and flexibility of the model for different scenarios and gives us a good insight into its performance. It helps to check if the model picks up the trends and patterns in the data. The validation results for each model are depicted below in Table 2. From the results, we can see that DT provides the most reliable results compared to all models.

**3. Result and Discussion**

The proposed system established stable real-time data collection from the ESP8266 using DHT11 (for temperature and humidity) and a soil moisture sensor (for moisture content in the soil) as shown in Fig.5. Using the Blynk IoT mobile application, the data was neatly visualized in a user-friendly manner, leaving the selection of crop type to the user. The pre-trained DT model was giving accurate results as intended, taking the sensor data as input parameters along with crop type and days, and giving the output as one or zero to control the irrigation system. The initial miniature working prototype of the system demonstrated the real-time application of the automated irrigation system's working capability of adjusting the flow of water through the pump based on the sensor reading taken while simultaneously displaying the values, helping to monitor the plant's condition and help it grow properly.

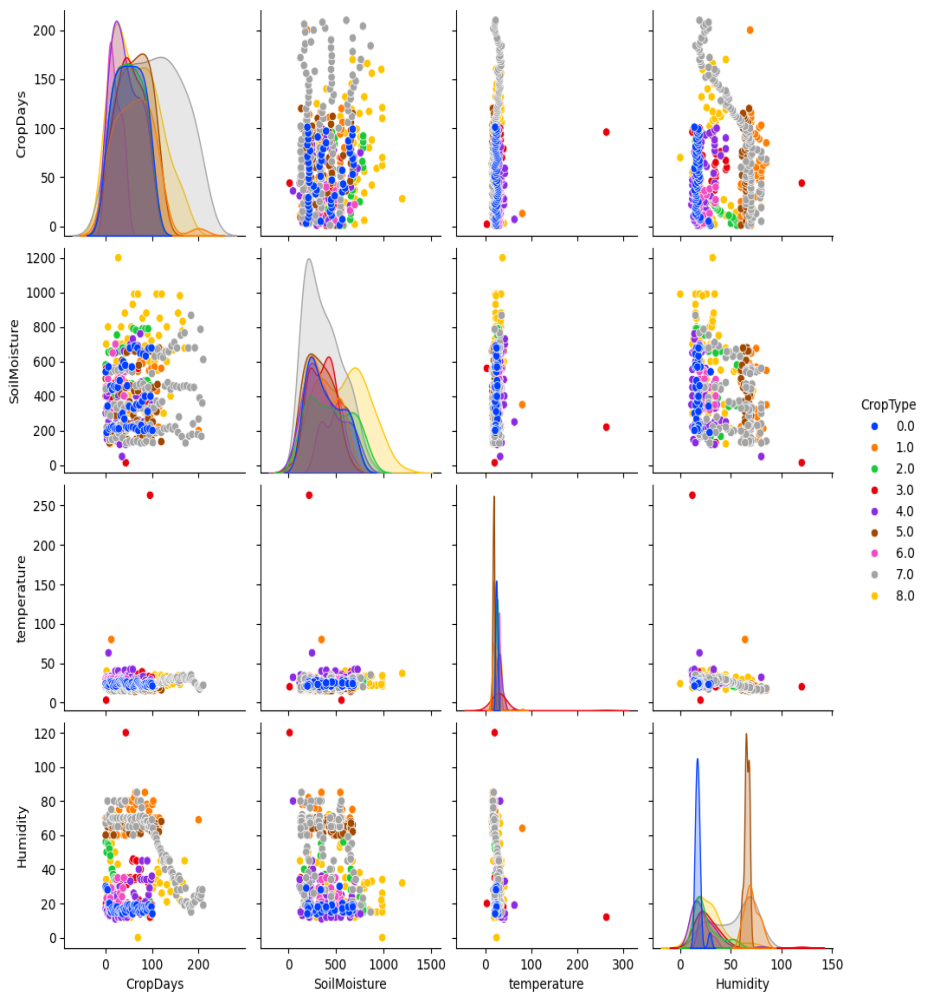
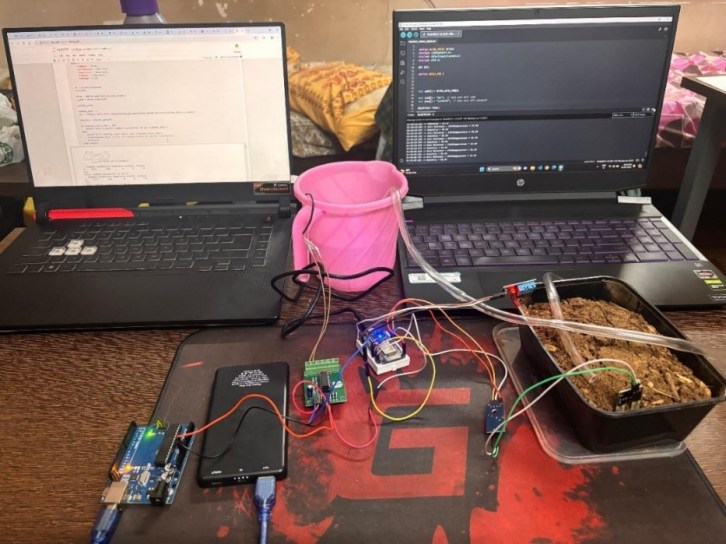


Figure 5. A Snapshot of the proposed Irrigation system

Figure 6. Visualization of the parameters collected from the external sensors

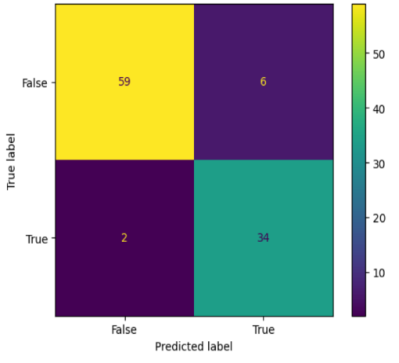
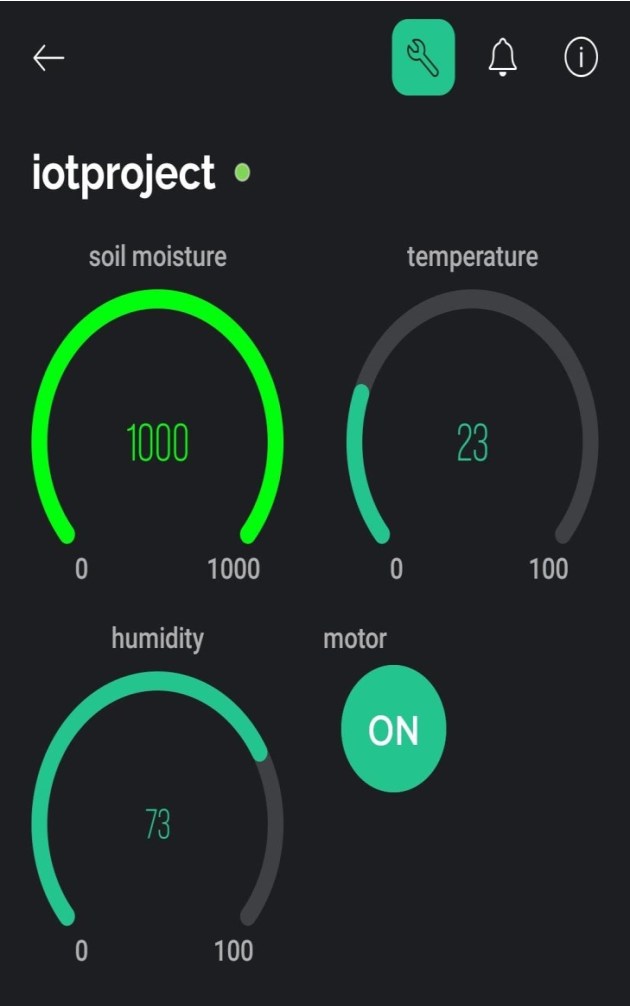
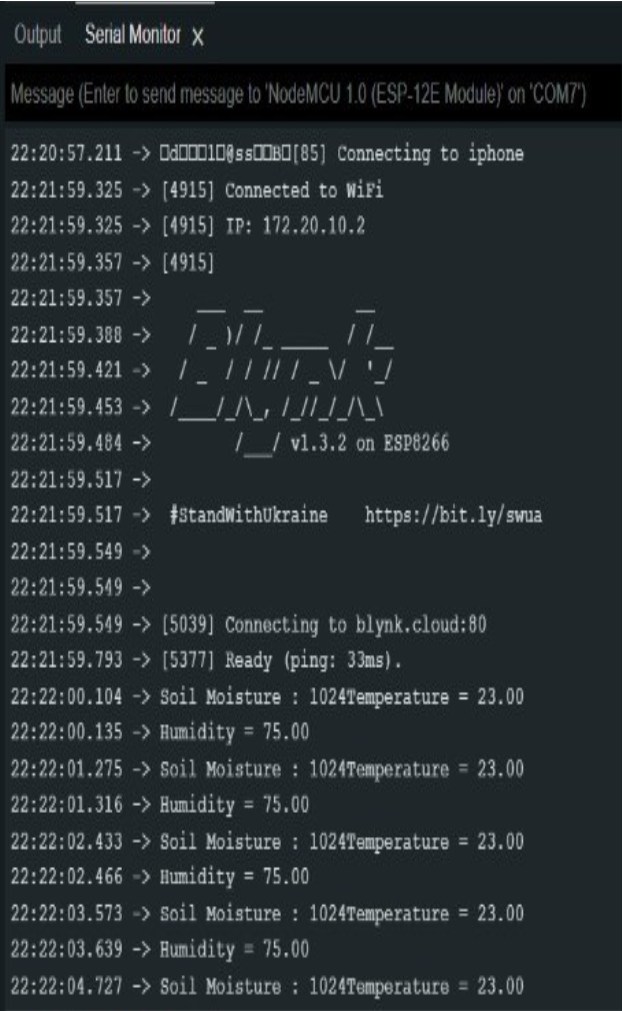
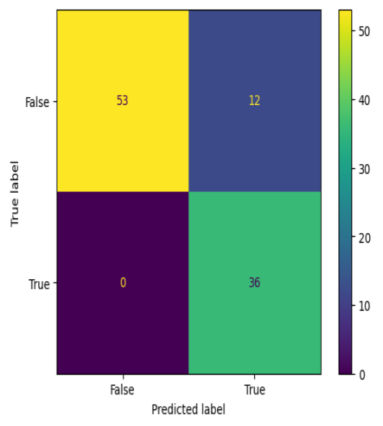
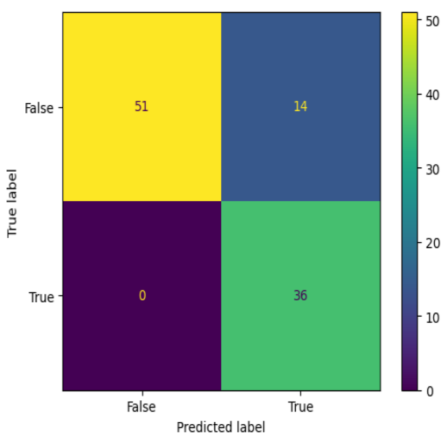


Figure 7.1 Confusion Matrix for

DT Model

Figure 7.2 Confusion Matrix for SVM

Model

Figure 7.3 Confusion Matrix for

LR Model

Figure 8 Output of the ESP8266 on the serial monitor

To test the performance of the proposed irrigation system, we evaluated the accuracy of DT, SVM, and LR for the same dataset. The models were able to accurately predict the output based on real-time sensor data, with an accuracy of over 92%, 86%, and 88%, respectively. The confusion matrix, which depicts precision, recall, F1-score, and accuracy for the three models, is shown in Figures 7.1, 7.2, and 7.3. It showcases the effectiveness of different models in automating the irrigation system and improving water conservation. Along with accuracy, this also evaluated the efficiency of the system. The system was able to process sensor data and make irrigation decisions in real-time without any significant delay. This ensures that water is delivered to crops as needed, without any waste.

Meticulously fitting five folds for each of the 15,552 parameters for a total of 77,760 fits for eight different types of parameters, including max\_features, splitter, ccp\_alpha, max\_depth, criterion, min\_samples\_leaf, random\_state, and max\_features. This exhaustive search helped us find the optimal parameters for the DT model, which helped improve the accuracy of the predictions. Despite the process being computationally intensive, it was a one-time process to fine-tune the model to give the best predictions while also avoiding overfitting to help give a more generalized model to work with real-time data. The high accuracy achieved is a product of the above method. The use of Grid Search CV along with data scaling resulted in an increase in accuracy from 88% to 92% as compared to the base paper, showcasing the effectiveness of the process. Moreover, the process of different parameter fitting for model optimization increases the reproducibility of the above process, which helps in further advancements in the respective field.

The ESP8266 module fetches the real-time sensor data and crop information and send them via virtual pins to the Blynk cloud, which then displays the sensor data on the mobile application. Based on the prevailing conditions and crop needs, the algorithm determines the optimal watering duration and frequency. On the mobile application, the data received is displayed on a dashboard which offers real-time insights into system performance and environmental parameters such as temperature, soil moisture, humidity, and pump state as depicted in Figure 8. The Python script running on the cloud servers fetches the data from the Blynk platforms, formats and processes it, and then passes it through the pre-trained DT model, which gives the output and sends it to the Blynk cloud platform. This targeted approach ensures that water is delivered precisely when and where it's most needed, promoting significant water conservation efforts. Various models used in different reference papers have been evaluated and compared with the proposed model. The proposed ML models in this paper outperform most of the other existing models in terms of accuracy and effectiveness as depicted in Table 3.

Table 3. Comparison of the proposed model with other existing models

**Reference**

**[25]**

**[30]**

**Supervised Model**

Linear Regression

KNN, SVM, Naive Bayes

**Features**

The paper aims to forecast the amount of water required using data collected through several detection sensors

The paper presents a threshold-based classification using sensor data and is implemented at ThinkSpeak IOT cloud.

**Accuracies**

-

KNN – 71%

Naive Bayes – 76%

SVM – 87.5 %

**[31]**

**[32]**

**[33]**

**Proposed Model**

KNN and SVM

Linear Regression

Decision Tree

SVM

Artificial Neural Network (ANN)

Fuzzy Inference System (FIS)

Adaptive Neuro-Fuzzy System (ANFIS)

Decision Tree

The paper is about detection of infection in plant samples.

The paper talks about ML model-based prediction on sensor data to maximize crop yield and minimize water wastage.

The paper talks about a IoT based smart irrigation system which uses a digital twin concept for efficient water usage. The system monitors soil condition in real time and sends data for analysis.

The paper proposes an IoT-based smart irrigation system that uses a decision tree model to automate irrigation decisions and a mobile app for remote monitoring, achieving significant water conservation.

SVM – 96%

LR – 85%

DT – 89.6%

SVM – 84%

ANN – 91%

FIS – 89.68%

ANFIS - 91.77%

DT - 92.079%

SVM – 86.138%

LR – 88.118%

From Table 3, it is inferred from the above results that the proposed system is a sustainable and energy efficient practicethat can help farmers reducetheirwaterand energy usageand improve crop yield.

**4. Conclusion**

**CRediT authorship contribution statement**

**Raju Patel, Manigandan M, Rohith G**: Writing – review & editing, Validation, Supervision, Methodology, Investigation, Formal analysis, Conceptualization. **Devansh Tewari**: Writing – review & editing, Methodology, Investigation, Formal analysis. **Adarsh Kumar Upadhyay, Parth Dedhia**: Writing – original draft, Software, Methodology, Formal analysis, Data curation. **Aneesh Raskar, Sahil Khodke**: Writing – original draft, Software, Formal analysis.

**S S V Mridula, S Anirudh, J Venkatesh:** Arduino coding and circuitry, Validation and Testing.

**R Advaith:** ML Training(Backend). **S Aniruddhan:** Frontend-Backend Integration, Testing.

**A Mirunalini**: Ukelele Chord Detector App(Frontend)

**Declaration of competing interest**

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

**References**

[1] Deja, J. A., Eska, B., Shrestha, S., Hoppe, M., Karolus, J., Kosch, T., Matviienko, A., Weiß, A., & Marky, K. (2023). Intelligent Music Interfaces: When Interactive Assistance and Augmentation Meet Musical Instruments. *Proceedings of the Augmented Humans International Conference.*

[2] Hori, T., Nakamura, K., & Sagayama, S. (2017). Music chord recognition from audio data using bidirectional encoder-decoder LSTMs. *APSIPA ASC.*

[3] Ito, T., & Arai, S. (2021). Harmonic Representation for CNN-LSTM Automatic Chord Recognition. *ICORIS 2021.*

[4] Kosch, T., Weiß, A., Deja, J. A., Shrestha, S., Hoppe, M., Matviienko, A., & Marky, K. (2024). Intelligent Music Interfaces: When Interactive Assistance and Adaptive Augmentation Meet Musical Instruments. *Augmented Humans Conference 2024.*

[5] Lin, M., Paul, R., Abd, M. A., Jones, J., Dieujuste, D., Chim, H., & Engeberg, E. (2023). Feeling the beat: a smart hand exoskeleton for learning to play musical instruments. *Frontiers in Robotics and AI, 10.*

[6] Nakayama, S., & Arai, S. (2018). DNN-LSTM-CRF Model for Automatic Audio Chord Recognition. *PRAI 2018.*

[7] Özbaltan, N. (2024). Real-time chord identification application. *Online Journal of Music Sciences.*

[8] Shu, N., & Anderson, D. V. (2024). Audiosockets: A Python socket package for Real-Time Audio Processing. *ArXiv.*

[9] Wang, D. (2023). Application of Machine Learning Technology in Classical Music Education. *International Journal of Web-Based Learning and Teaching Technologies.*

[10] Wibawa, I. M. S., & Putra, I. K. (2022). Design of air pressure and height measuring equipment based on Arduino nano using BME280 sensor. *International research journal of engineering, IT & scientific research.*