

A Comparative Evaluation of Machine Learning Algorithms for Smart Irrigation Decision Support

Rohit Nagwanshi¹, Dr. Ruchika Pachori²

Department of Information Technology, Mahakal Institute of Technology, Ujjain, Madhya Pradesh, India

ABSTRACT

Sustainable farming depends on careful water management, particularly where irrigation draws on limited supplies. Machine learning (ML) can help plan irrigation by finding patterns in soil and weather readings, but it is hard to know which algorithm works best, because published studies rarely share their datasets, validation schemes or performance metrics. This paper compares four classical algorithms on an irrigation on/off prediction task: Decision Tree (DT), Random Forest (RF), Support Vector Machine (SVM) and K-Nearest Neighbors (KNN).

The data come from a public IoT-based agricultural dataset, Dataset for Predicting Watering the Plants. Only 2,200 of its 100,000 sensor records had every field filled in, so after dropping records with missing values those 2,200 became our working set. We judged each model on test accuracy, stratified five-fold cross-validation, precision, recall, F1-score, confusion matrices, ROC curves with AUC, feature importance and running time.

We also ran GridSearchCV to see how far hyperparameter tuning changes the picture, and used McNemar's test to check whether the differences between the models' predictions were larger than chance alone would explain.

With default settings, Decision Tree had the best test accuracy (91.36%) and scored $91.18\% \pm 1.52\%$ in cross-validation; tuning raised its test accuracy to 92.27%. Random Forest had the best baseline AUC (0.954). Decision Tree had the lowest prediction latency (0.0107 s) and trained in 0.09 s. The tree-based models differed significantly from SVM and KNN, while the baseline gap between Decision Tree and Random Forest was not significant. Together, these results offer a reference point for choosing models in smart irrigation decision support, where prediction quality and computing cost both matter.

Keywords: *Smart irrigation, Machine learning, Comparative evaluation, Decision Tree, Random Forest, Computational efficiency, Hyperparameter optimization, Statistical significance, Internet of Things, Precision agriculture.*

I. INTRODUCTION

Agriculture is one of the biggest users of freshwater, and when that water is scarce, crop output suffers and natural resources come under strain [1]. Farmers and policymakers therefore need ways to irrigate more efficiently. Most irrigation today still runs on a fixed schedule or a preset soil-moisture cutoff. Such rules are simple to apply, but they ignore changes in weather, soil condition and crop demand, so water may be applied when it is not needed, which wastes supply and can sometimes hurt yield [2].

Low-cost IoT sensors now make it practical to record soil moisture, temperature, humidity and rainfall continuously, giving a running picture of field conditions. ML models can learn how these readings relate to irrigation need and support decisions that adapt better than fixed rules [3]. Even so, accuracy is only part of a model's value. The computation it needs, and whether it suits the place it will be deployed, matter too.

Decision Tree (DT), Random Forest (RF), Support Vector Machine (SVM) and K-Nearest Neighbors (KNN) are classical algorithms that have been applied to agricultural prediction and soil-moisture problems [3]–[5]. They differ in how they learn, what they cost to run and how well they predict. Results from separate papers are hard to line up, though, because each one uses its own data, preprocessing, metrics and settings. Accuracy also tends to get most of the attention, while stability, computing cost and the practical consequences of choosing one model over another get treated more lightly.

This work grows out of the gaps we identified in our earlier systematic literature review [24]. That review called for more consistent algorithm comparisons, evaluation that goes beyond accuracy, and closer attention to computational efficiency. We respond by testing four classical classifiers on one public IoT agricultural dataset under a single experimental setup. The evaluation combines predictive metrics with cross-validation, feature analysis, timing measurements, hyperparameter tuning and significance testing.

This paper makes the following contributions:

1. **Comparative assessment of classifiers:** DT, RF, SVM and KNN are compared on the same data and under the same experimental conditions.
2. **Multi-metric performance evaluation:** Accuracy, precision, recall, F1-score, stratified five-fold cross-validation, confusion matrices and ROC-AUC are reported together, giving a fuller view of each model.
3. **Computational efficiency analysis:** Training time and prediction latency are measured alongside predictive results, to show what each algorithm costs to run.

4. **Model optimization and statistical testing:** GridSearchCV tests how much hyperparameter choice matters, and McNemar's test checks whether differences between classifiers are statistically meaningful.

5. **Implications for irrigation decision support:** The results inform model choice and sensor prioritization for smart irrigation, weighing predictive strength against computational demand.

In this paper, decision support means guidance drawn from comparing ML models and their input features. We did not build an irrigation controller, a farmer-facing app or automatic ON/OFF actuation; implementing and field-testing such a system is left for future work.

II. RELATED WORK

Several recent studies have applied machine learning to smart irrigation. Goap et al. [3] built an open-source IoT irrigation management system that predicts irrigation needs from soil moisture, temperature and weather forecasts, showing that ML-driven automation is feasible. They did not, however, compare several algorithms or examine computational efficiency.

Tace et al. [4] applied classifiers to multi-sensor data (soil humidity, temperature and rainfall) for irrigation decisions in an IoT-based system. Their work shows the approach is practical, but it does not look at how training time or latency differs between algorithms. Setiawan et al. [5] compared Linear Regression, KNN and Decision Tree for calibrating low-cost soil-moisture sensors. KNN gave the lowest RMSE, followed by Decision Tree, which suggests both suit soil-moisture prediction. Their focus was regression error, however, so classification metrics, ROC-AUC and computational cost were not examined.

Liakos et al. [6] reviewed ML in agriculture broadly. They noted that classical methods such as Decision Tree, Random Forest and SVM remain common for prediction because they are interpretable and cheaper to run than deep learning, and they argued for standardized evaluation protocols across studies.

Kaur et al. [12] designed a hybrid IoT irrigation system for Rajasthan, India, and compared KNN, Naive Bayes, Random Forest, SVM and Logistic Regression for automated motor control; they did not report training-time or latency comparisons. Hossain et al. [13] systematically reviewed real-time IoT-ML irrigation management and found that several ML approaches improved irrigation prediction or management over fixed-threshold strategies, with some reaching R-squared values above 0.89. That review synthesizes separately reported studies rather than running one controlled head-to-head comparison, so we use it as supporting evidence for adopting ML, not as proof.

A comparative study on ensemble learning for IoT-based smart irrigation [14] tested SVM, Decision Tree, Random Forest, Boosting and hybrid ensembles for water-usage prediction. It found that ensemble combinations can beat individual classical models, but it also lacked a unified framework covering both accuracy and efficiency.

Deep learning has been tried for irrigation and soil-moisture prediction as well. Yan et al. [20] proposed a BiLSTM-CNN-Attention model for irrigation volume, Park et al. [21] used an RNN-LSTM for soil moisture in soybean fields, and Basir et al. [23] forecast subsurface soil moisture with an LSTM and weather data. These models often reach high accuracy, but they need far more computation and training time than tree-based or distance-based classifiers. That is why this study concentrates on lightweight, low-latency models for resource-constrained IoT settings. Nguyen et al. [22] likewise preferred low-cost multi-sensor ML over complex deep architectures for field-deployable soil-moisture monitoring.

Across these studies, none evaluates cross-validation, ROC-AUC, confusion matrices, feature importance and computational efficiency together under identical conditions. This paper aims to fill that gap.

III. Methodology

A. Dataset Description and Preprocessing

We used the publicly available IoT-based agricultural dataset "Dataset for Predicting Watering the Plants" from Kaggle [11]. The original dataset has 100,000 instances and 15 columns: 14 input attributes and one target variable. The inputs are soil moisture, temperature, soil humidity, time of day, air temperature, wind speed, air humidity, wind gust, atmospheric pressure, soil pH, rainfall, and the macronutrients nitrogen (N), phosphorus (P) and potassium (K). The target variable, Status, is the binary irrigation status (ON/OFF). Table I summarizes the preprocessing stages, and Table II gives the feature-wise missing-value percentages that motivated our use of listwise deletion.

Table I: Dataset Preprocessing Statistics

Preprocessing Stage		Record Count	Remarks
Original	Dataset	100,000	Raw IoT sensor data (Kaggle)

Incomplete Rows Removed	97,800	Listwise deletion applied (rows with ≥ 1 missing value)
Final Clean Dataset	2,200	Used for model training & evaluation
Training Set (80%)	1,760	Model fitting
Test Set (20%)	440	Performance evaluation

Table II: Per-Feature Missing Value Distribution (Original Dataset)

Feature Group	Missing (%)	Likely Cause
Soil Moisture, Soil Humidity, Time, Temperature	0.0%	Logged continuously by the on-board IoT sensor node
Air Temperature, Wind Speed, Air Humidity, Wind Gust, Pressure	76.0%	Logged only when the auxiliary weather-station feed was active
N, P, K, pH, Rainfall	97.8%	Periodic, non-continuous lab-based soil sampling, rarely time-aligned with sensor reads

Table II shows that the missing values are not scattered at random; they fall into three tiers. Soil Moisture, Soil Humidity, Time and Temperature, which come from the on-board sensor node, are complete, and so is the target Status. Five weather

variables (Air Temperature, Wind Speed, Air Humidity, Wind Gust and Pressure), apparently from an auxiliary weather-station feed, are missing in 76.0% of records. Five soil-chemistry variables (N, P, K, pH and Rainfall), apparently from occasional lab sampling, are missing in 97.8%.

A record is complete only when all three tiers are present at once, which explains the drop from 100,000 to 2,200 complete records. This block pattern suggests the dataset was put together by merging sources with different logging frequencies, rather than being hit by random sensor faults. The missingness looks structured and probably non-random [18], although the documentation is too thin to say whether it is truly Missing Not At Random (MNAR) or Missing At Random given logging-source information we do not have.

We chose listwise deletion so that only genuine sensor readings were used and no estimated values entered the data. Mean, median and k-NN imputation were considered but left for future work (Section V), so that imputation artifacts would not be mistaken for real differences between models in this first comparison. The uncertainty about the missingness mechanism is an added limitation, discussed in Section IV.I.

The target was label-encoded (ON=1, OFF=0) and the data were split 80:20 with random state 42, giving 1,760 training and 440 test instances. We used a random split rather than a time-based or grouped one because it suits the benchmark-style framing of this first comparison.

One question remains open. The dataset documentation does not say whether the Status label came from a fixed rule or threshold on soil moisture and time, or whether the 2,200 retained records include repeated readings from the same node or period. If either is true, part of the reported accuracy might reflect a model rediscovering the rule or memorizing near-duplicate records, instead of learning a pattern that generalizes. Checking how the label was generated, and moving to a time-based or grouped split where possible, are both left for future work (Section V).

B. Machine Learning Models

We implemented four classical supervised classifiers with the Scikit-learn library (version 1.6.1) in Python 3.13.15 [15]: Decision Tree (DT), Random Forest (RF), Support Vector Machine (SVM) and K-Nearest Neighbors (KNN). They were chosen to compare different learning approaches for binary irrigation-status prediction under a common experimental setup.

The Decision Tree used the Gini impurity criterion and a fixed random state of 42. The Random Forest used 100 decision trees with a random state of 42. The Support Vector Machine used a radial basis function (RBF) kernel with probability estimation switched on to support ROC analysis, and the K-Nearest Neighbors classifier used five neighbors with Euclidean distance.

All classifiers were trained and evaluated on the same 80:20 training-test split. The training subset was used to fit the models, and the held-out test subset to measure predictive performance. We compared the models on accuracy, precision, recall, F1-score, confusion matrices, ROC-AUC and computational time.

No feature scaling was applied in the reported baseline experiments. SVM and KNN are sensitive to differences in feature scales, so their performance may be affected by this choice. Evaluating a scaled pipeline is therefore a direction for further investigation.

The experiments ran on Google Colab with Python 3.13.15 and scikit-learn 1.6.1, on a 64-bit Linux system with an x86_64 architecture, two CPU cores and about 12.67 GB of RAM. No GPU was detected during the runtime check. Training-time and prediction-latency measurements come from a single run in this environment.

C. Evaluation Framework

Every classifier was evaluated on (i) accuracy (%), (ii) precision, recall and F1-score, (iii) five-fold stratified cross-validation accuracy, (iv) confusion matrices, (v) ROC curves and AUC [17], (vi) feature importance, and (vii) training time and prediction latency (in seconds). Together these measures show how well each model performs, how reliable its classifications are and how efficient it is, which helps judge whether it is feasible for real-time or resource-constrained IoT applications.

IV. RESULTS AND DISCUSSION

A. Predictive Performance and Cross-Validation

Table III gives the predictive performance and 5-fold cross-validation results for all four classifiers.

Algorithm	Accuracy (%)	CV Accuracy (%)	CV Std (±)	Precision	Recall	F1-Score
Decision Tree	91.36	91.18	1.52%	0.91	0.91	0.91
Random Forest	88.86	89.77	1.37%	0.89	0.89	0.89
SVM	81.36	80.09	1.37%	0.81	0.81	0.81
KNN	77.50	75.50	1.69%	0.78	0.77	0.77

Decision Tree achieves the highest test accuracy (91.36%) and consistent cross-validation accuracy (91.18% ± 1.52%), which shows stable performance across different data partitions. That Decision Tree beats Random Forest on this dataset is noteworthy, and it may be tied to the small sample left after preprocessing (2,200 instances).

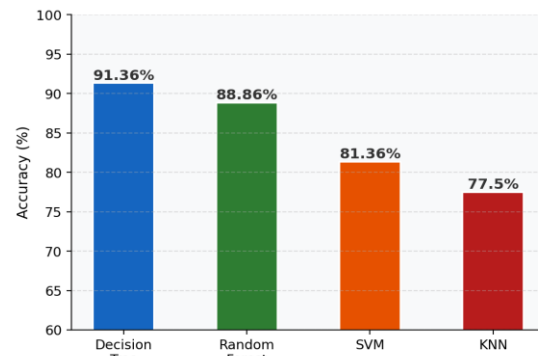
Ensemble methods such as Random Forest usually gain their edge by reducing variance across many trees. With a small dataset, the trees are less diverse, so the overhead of aggregating 100 estimators can cost more than it gains compared with a single well-fitted tree.

The high importance of time of day (0.41) points to a related possibility: one dominant split early in the tree may capture much of the decision boundary by itself, and a lone Decision Tree can exploit that directly. These are plausible explanations, not causally verified mechanisms. Confirming either would need further experiments, such as varying the sample size directly or running an ablation on the time-of-day feature.

Random Forest reaches 89.77% cross-validation accuracy with a low standard deviation, which indicates stable performance across partitions of the retained complete-case subset. SVM records 80.09% CV accuracy and KNN 75.50%, confirming that they are comparatively less suited to this classification task under default settings.

The low CV standard deviations across all models point to stable performance across partitions of the retained 2,200-record subset. As Section IV.I discusses, however, this stability should not be taken as proof of reproducibility or of generalizability to the full, discarded population, because cross-validation resampling cannot detect a systematic offset between the records that were kept and those that were dropped.

Fig. 1: Accuracy Comparison of ML Classifiers



Algorithm	AUC Score	Rank
Random Forest	0.954	1st
Decision Tree	0.914	2nd
SVM	0.882	3rd
KNN	0.822	4th

Fig. 2: Training Time Comparison of ML Classifiers

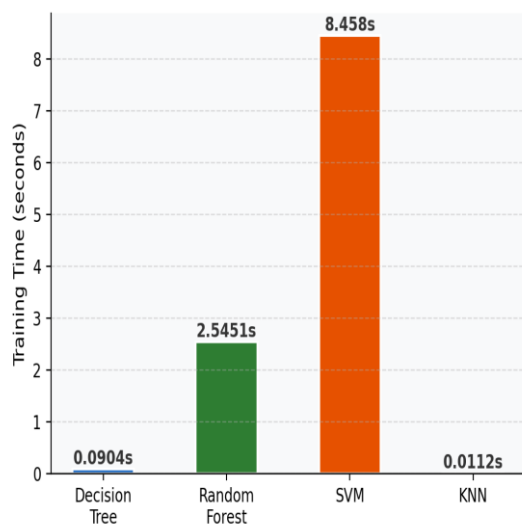
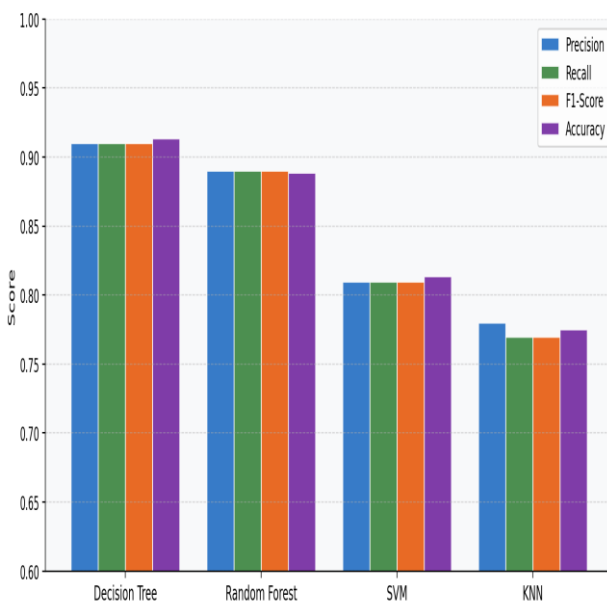


Fig. 3: Performance Metrics Comparison Across All Classifiers



B. Confusion Matrix Analysis

Fig. 4 shows the confusion matrices for all four classifiers on the 440-instance test set. Decision Tree correctly classifies

187 OFF and 215 ON instances, with only 17 and 21 misclassifications respectively, for the lowest total (38) of all models. Random Forest correctly classifies 180 OFF and 211 ON instances (49 misclassifications in total). SVM misclassifies 82 instances (39 OFF, 43 ON), while KNN has the highest count at 99 (58 OFF, 41 ON), confirming its comparatively weaker performance on this dataset.

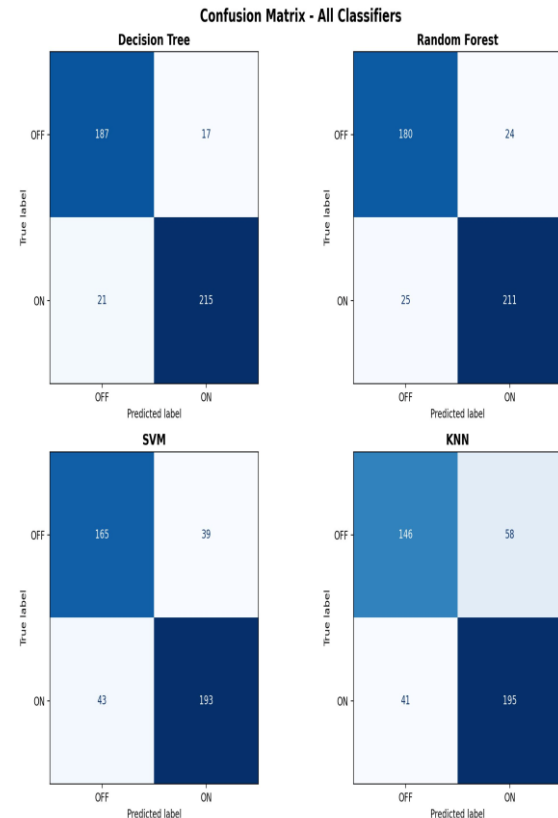


Fig. 4: Confusion Matrices for All Four Classifiers

C. ROC Curve and AUC Analysis

Table IV gives the AUC scores and Fig. 5 shows the ROC curves for all classifiers. Random Forest achieves the highest AUC, 0.954, meaning it is best at ranking ON instances above OFF instances across classification thresholds. AUC measures this ranking ability rather than probability calibration, so the result should not be read as evidence that Random Forest's predicted probabilities are well calibrated. Decision Tree reaches an AUC of 0.914, SVM 0.882 and KNN 0.822. Decision Tree beats Random Forest on raw accuracy, but Random Forest's higher AUC suggests it ranks positive and negative instances better across thresholds, which matters for irrigation decision systems whose threshold can be tuned.

Table IV: ROC-AUC Scores Comparison

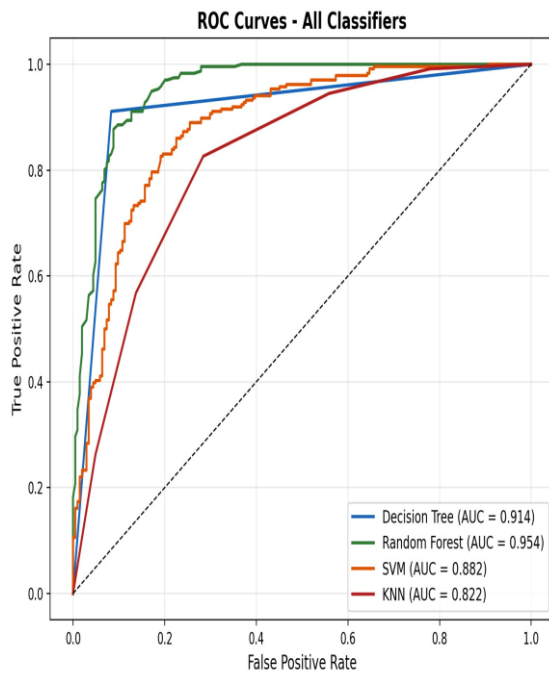


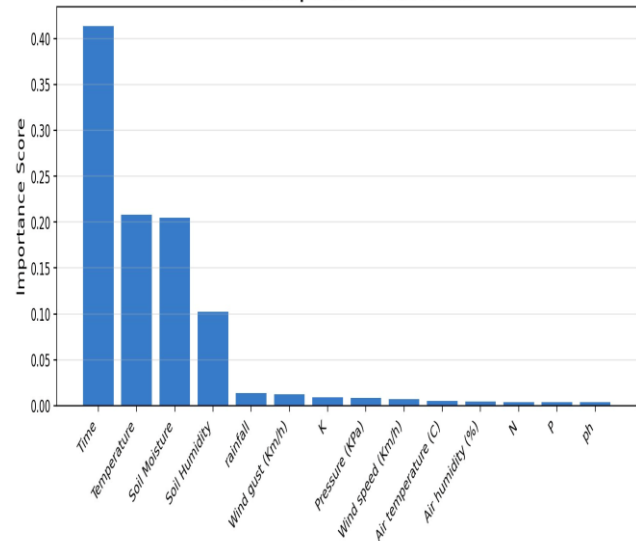
Fig. 5: ROC Curves for All Classifiers

D. Feature Importance Analysis

Fig. 6 shows the feature importance scores from the Decision Tree classifier. Time of day (0.41) is the most influential feature, followed by Temperature (0.21), Soil Moisture (0.20) and Soil Humidity (0.10). Together these four features account for about 92% of the total importance, so temporal and thermal variables are the most influential predictors within the fitted Decision Tree for this dataset. Rainfall, wind gust and the soil nutrient parameters (N, P, K, pH) contribute very little, which suggests the sensor set could be simplified in resource-constrained IoT deployments.

One caveat needs stating plainly. Decision Tree feature importance measures how much each variable reduces impurity in the fitted model, which is a predictive association and not a causal determinant of irrigation decisions. The low importance of the nutrient and rainfall variables may also partly reflect how heavily listwise deletion thinned them out (Section III.A), rather than a real lack of relevance to irrigation. Before dropping low-importance sensors from a deployed system, the idea should therefore be tested with a proper ablation study that compares accuracy using soil moisture alone, soil moisture plus temperature, the top four features and the full feature set, instead of acting directly on this importance ranking.

Feature Importance - Decision Tree



E. Computational Efficiency

Table V gives the computational efficiency comparison. Decision Tree has the lowest measured prediction latency (0.0107 s) and a low training time (0.09 s) in this experiment, although deployment suitability still needs validation on actual edge hardware. KNN has a negligible training time (0.0112 s) because it is a lazy learner: it stores all training instances without building an explicit model. Its prediction latency is higher (0.0257 s), since every prediction needs distances to all stored training points at query time. Random Forest needs 2.54 s for training because of ensemble aggregation across 100 trees. SVM has the highest training overhead (8.46 s), which limits its use in dynamic agricultural environments.

These training-time and prediction-latency figures come from a single run in the hardware and software environment described in Section III.B. Absolute values depend on the hardware and implementation, so we report them as a relative comparison across classifiers under identical conditions, not as universal, hardware-independent benchmarks.

Table V: Computational Efficiency Comparison

Algorithm	Training Time (s)	Prediction Time (s)	Relative Timing Position
Decision Tree	0.0904	0.0107	1st — Lowest observed time
KNN	0.0112	0.0257	2nd
Random Forest	2.5451	0.0258	3rd
SVM	8.4580	0.0293	4th — Highest observed time

F. Hyperparameter Tuning Analysis (GridSearchCV)

To address concerns about the fairness of comparing models under default settings, we re-evaluated all four classifiers with GridSearchCV, using 5-fold stratified cross-validation on the training set and predefined hyperparameter grids.

Algorithm	Default Acc. (%)	Tuned Acc. (%)	Tuned AUC	Best Hyperparameters
Decision Tree	91.36	92.27	0.946	criterion=entropy, max_depth=10, min_samples_leaf=2
Random Forest	88.86	88.86	0.956	n_estimators=200, max_depth=None, min_samples_leaf=1
SVM	81.36	81.36	0.892	C=100, gamma=scale, kernel=rbf
KNN	77.50	77.27	0.841	n_neighbors=7, weights=distance, metric=manhattan

Table VI: Default vs. GridSearchCV-Tuned Performance

Hyperparameter tuning improved Decision Tree test accuracy from 91.36% to 92.27% (best parameters: entropy criterion, max_depth=10, min_samples_leaf=2) and raised its AUC modestly from 0.914 to 0.946. Random Forest and SVM ended with test accuracies matching their default configurations (88.86% and 81.36% respectively), although the Random Forest AUC improved marginally to 0.956 with n_estimators=200. KNN accuracy fell slightly to 77.27% under the tuned Manhattan-distance configuration despite a higher cross-validation score, which illustrates a small train/test variance at this sample size. Tuning search times ranged from 1.36 s (KNN) to 61.71 s (Random Forest), reflecting the larger search space and per-fit cost of ensemble methods. Overall, tuning reinforces the baseline ranking rather than overturning it: Decision Tree remains the strongest individual performer, and Random Forest remains the strongest by AUC.

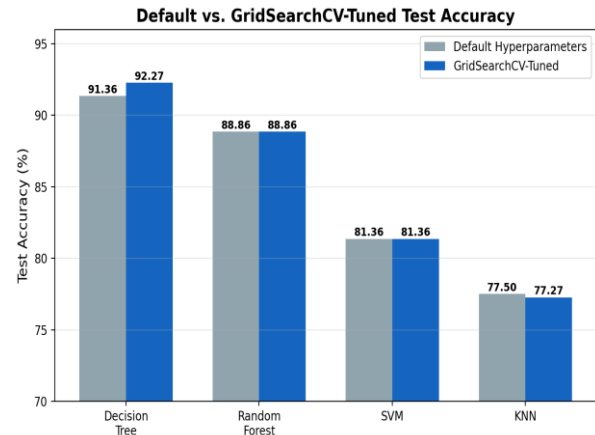


Fig. 7: Default vs. GridSearchCV-Tuned Test Accuracy Comparison

G. Statistical Significance Analysis

To check whether the accuracy differences among classifiers reflect genuine performance gaps, not just sampling variation on the 440-instance test set, we ran pairwise McNemar tests [16] on the default-hyperparameter predictions. The McNemar test examines whether two classifiers' disagreements (cases where one is right and the other is wrong) are symmetric, and it suits paired binary classification outcomes on a shared test set. Table VII gives the results.

Comparison	χ^2 Statistic	p-value	Significant ($\alpha=0.05$)?	Holm-Adj. Signif.?
Decision Tree vs. Random Forest	1.75	0.185	No	No
Decision Tree vs. SVM	19.67	<0.001	Yes	Yes
Decision Tree vs. KNN	34.95	<0.001	Yes	Yes
Random Forest vs. SVM	17.36	<0.001	Yes	Yes
Random Forest vs. KNN	27.28	<0.001	Yes	Yes
SVM vs. KNN	3.16	0.075	No	No

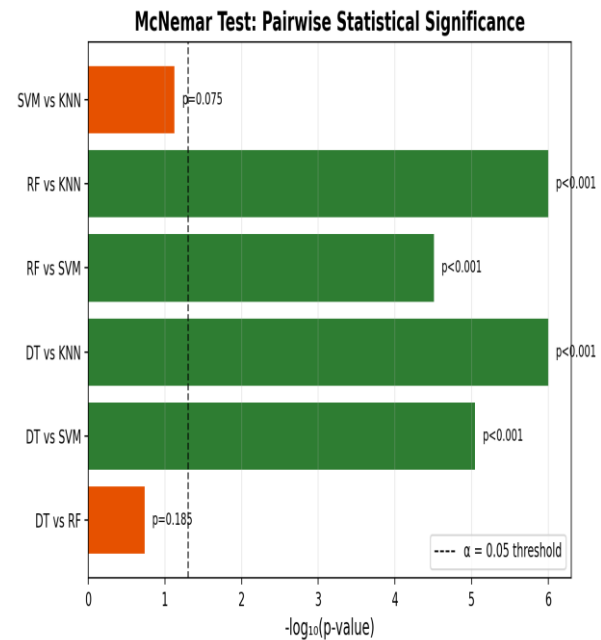
Table VII: Pairwise McNemar Test Results (Default Hyperparameters)

The McNemar test shows that the 2.5-percentage-point accuracy gap between Decision Tree and Random Forest under default hyperparameters is not statistically significant ($p=0.185$). The two models therefore perform comparably, and practitioners can reasonably choose between them on secondary criteria such as AUC or computational cost. In contrast, both Decision Tree and Random Forest significantly outperform SVM and KNN (all $p<0.001$), confirming that the tree-based and ensemble methods have a genuine, non-random advantage over the distance- and margin-based classifiers on this dataset. The SVM-versus-KNN difference is also not statistically significant ($p=0.075$).

Repeating the McNemar test on the GridSearchCV-tuned Decision Tree and Random Forest predictions gives $\chi^2=4.00$, $p=0.046$. After tuning, Decision Tree therefore shows a nominally significant advantage over Random Forest at the uncorrected $\alpha=0.05$ level. This borderline result was not subjected to multiple-comparison correction and should not be treated as definitive evidence of superiority. Decision Tree is best regarded as the preferred model for this dataset mainly because of its computational simplicity and interpretability, not because of this single borderline result.

Running six pairwise comparisons at once raises the usual concern that more results will look "significant" by chance. We therefore applied a Holm-Bonferroni correction across the six default-hyperparameter comparisons, and the corrected result is in the final column of Table VII. The four comparisons with $p<0.001$ (Decision Tree and Random Forest each against SVM and KNN) comfortably survive every step of the Holm procedure, since they lie far below even its strictest per-comparison threshold ($\alpha/6 = 0.0083$). Decision Tree versus Random Forest ($p=0.185$) and SVM versus KNN ($p=0.075$) stay non-significant both before and after correction, so the correction changes none of the substantive conclusions.

The one exception is the separate post-tuning comparison ($\chi^2=4.00$, $p=0.046$). It was not part of the six-comparison family and was not corrected, because it was a follow-up test prompted by the default-hyperparameter results rather than a pre-specified independent comparison. It is best read as nominally significant at the uncorrected 0.05 level, not as confirmed evidence that Decision Tree beats Random Forest after tuning. A useful addition in a later revision would be to report 95% confidence intervals for accuracy, along with balanced accuracy and the Matthews correlation coefficient, in addition to the McNemar p-values given here, to characterize the uncertainty around these comparisons more fully.



H. Discussion

The results show that each classifier has distinct strengths and limitations for smart irrigation prediction. Rather than pointing to one universally superior model, they suggest that model selection should depend on the requirements of the particular irrigation system. Decision Tree achieved the highest classification accuracy and also the lowest prediction latency, with a low training time, which makes it a promising candidate for real-time and resource-constrained deployments, though this still needs confirmation through hardware-level validation. Random Forest, in contrast, showed stronger threshold-independent discrimination, meaning a better ability to rank ON instances above OFF instances across decision thresholds, as its higher ROC-AUC shows. These observations underline the importance of evaluating machine learning models on several performance criteria rather than on accuracy alone.

These findings are broadly consistent with earlier studies reporting that Decision Tree models work well for soil-moisture-related prediction [5], and Liakos et al. [6] likewise stressed that classical tree-based methods remain relevant in agricultural ML. The limitations of this study, including the small complete-case sample and possible selection bias, are collected in Section IV.I.

Random Forest, although slightly lower in test accuracy (88.86%), achieves the highest AUC (0.954) and strong CV accuracy (89.77%), which suggests better discrimination across thresholds, a valuable property for adjustable irrigation scheduling systems. SVM and KNN show comparatively lower predictive performance, and SVM in particular has considerably higher computational demands, so they are less suitable for large-scale real-time agricultural deployment under the configurations evaluated here. Feature importance analysis also shows that time of day and temperature are the dominant irrigation predictors in this dataset, which can guide IoT sensor prioritization in resource-constrained deployments. As reported in Sections IV.F and IV.G, GridSearchCV tuning further improves Decision Tree to 92.27% test accuracy. The

McNemar test shows Decision Tree's edge over SVM and KNN holding up under the default settings and surviving Holm-Bonferroni correction, whereas its edge over Random Forest becomes nominally significant, at the uncorrected level, only after tuning ($p=0.046$). Under default hyperparameters, then, the choice between Decision Tree and Random Forest should follow deployment priorities (computational efficiency favors Decision Tree; probabilistic discrimination favors Random Forest) rather than accuracy alone.

I. Limitations

Read these findings with the following limitations in mind. First, only 2,200 of the 100,000 records were complete, and because the missingness is structured rather than random (Section III.A), this subset may not represent the full dataset; cross-validation shows variation within the subset but cannot reveal such bias. Second, the dataset documentation does not say how the Status label was generated or whether repeated readings fall on both sides of the split, so part of the accuracy may reflect a recoverable rule or near-duplicate records. Third, SVM and KNN were run without feature scaling, so their results may understate what these models can achieve. Fourth, timings come from a single run and serve only as a relative comparison. Finally, feature importance shows association within the fitted tree, and the low importance of the nutrient and rainfall variables may partly reflect the records lost to listwise deletion, so an ablation study is needed before any sensor is dropped. Together these points form the main agenda for future work (Section V).

V. CONCLUSION

This study compared four widely used ML algorithms for predicting irrigation need from an IoT agricultural dataset. The models were judged on accuracy, cross-validation, ROC-AUC, feature importance, computational efficiency, tuning and statistical significance, so the comparison does not rest on a single number. The outcome is a practical comparison of classical approaches for irrigation decision support that addresses several limitations of earlier smart-irrigation work.

The Decision Tree classifier gives the strongest balance of predictive performance and computational efficiency. It has the highest default test accuracy (91.36%, rising to 92.27% after GridSearchCV tuning), consistent CV accuracy ($91.18\% \pm 1.52\%$), the lowest prediction latency (0.0107 s) with a low training time (0.09 s), and the lowest misclassification count (38 of 440). McNemar testing confirms that, under default hyperparameters, Decision Tree differs significantly from SVM and KNN ($p<0.001$), and these comparisons stay significant after Holm-Bonferroni correction. Its advantage over Random Forest is not significant under default hyperparameters ($p=0.185$) and only nominally significant at the uncorrected $\alpha=0.05$ level after tuning ($p=0.046$), so it should not be regarded as definitively confirmed. Random Forest offers a complementary strength, with the highest AUC (0.954, improving to 0.956 after tuning) and stable cross-validation accuracy (89.77%) across partitions of the retained subset, which makes it suitable for probability-based irrigation

threshold systems. Feature importance analysis identifies time of day, temperature and soil moisture as the top irrigation predictors, giving practical sensor-prioritization guidance for resource-constrained IoT deployments.

Future work will test imputation (mean, median, k-NN) in place of listwise deletion to recover the many discarded records, validate against an external, fully logged dataset to check for selection bias in the complete-case subset, and explore hybrid DT-RF ensembles, gradient boosting (XGBoost [19], LightGBM) and deep learning for multi-crop scheduling. Other planned items are confirming how the Status label was produced, checking the split for near-duplicate or repeated observations and comparing it with a time-based or grouped split, running a feature-ablation study to test the sensor-simplification idea in Section IV.D, and adding 95% confidence intervals and the Matthews correlation coefficient to the Holm-Bonferroni-corrected results in Section IV.G. We also plan field validation across regions, integration with edge platforms for low-latency deployment, and a concrete architecture that turns predictions into ON/OFF commands, farmer alerts and hardware control logic, so that the decision-support component can be built and evaluated directly instead of being inferred from classification performance.

REFERENCES

- [1] FAO, The State of the World's Land and Water Resources for Food and Agriculture. Rome, Italy: FAO, 2023.
- [2] R. G. Allen, L. S. Pereira, D. Raes, and M. Smith, Crop Evapotranspiration: Guidelines for Computing Crop Water Requirements, FAO Irrigation and Drainage Paper No. 56. Rome: FAO, 1998.
- [3] A. Goap, D. Sharma, A. K. Shukla, and C. R. Krishna, "An IoT based smart irrigation management system using machine learning and open source technologies," *Computers and Electronics in Agriculture*, vol. 155, pp. 41-49, 2018.
- [4] Y. Tace, M. Tabaa, S. Elfalali, C. Leghris, H. Bensag, and E. Renault, "Smart irrigation system based on IoT and machine learning," *Energy Reports*, vol. 8, pp. 1025-1036, 2022.
- [5] I. Setiawan, M. D. Th. Musa, D. A. Afriza, and S. N. Hafidah, "Comparing machine learning algorithms to enhance volumetric water content prediction in low-cost soil moisture sensor," *Journal of Applied Informatics and Computing*, vol. 9, no. 1, pp. 140-145, 2025.
- [6] K. G. Liakos, P. Busato, D. Moshou, S. Pearson, and D. Bochtis, "Machine learning in agriculture: A review," *Sensors*, vol. 18, no. 8, Art. no. 2674, 2018.
- [7] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5-32, 2001.
- [8] V. N. Vapnik, *The Nature of Statistical Learning Theory*. New York, NY, USA: Springer, 1995.

- [9] T. M. Cover and P. E. Hart, "Nearest neighbor pattern classification," *IEEE Trans. Inf. Theory*, vol. 13, no. 1, pp. 21-27, Jan. 1967.
- [10] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed. New York, NY, USA: Springer, 2009.
- [11] Kaggle, "Dataset for Predicting Watering the Plants," [Online]. Available: <https://www.kaggle.com/datasets/nelakurthisudheer/dataset-for-predicting-watering-the-plants> (Accessed: Jan. 10, 2026).
- [12] A. Kaur, D. P. Bhatt, and L. Raja, "Developing a hybrid irrigation system for smart agriculture using IoT sensors and machine learning in Sri Ganganagar, Rajasthan," *Journal of Sensors*, vol. 2024, Art. no. 6676907, 2024.
- [13] M. S. Hossain, M. A. Islam, M. S. Rahman, and M. A. Alam, "Internet of Things-based automated solutions utilizing machine learning for smart and real-time irrigation management: A review," *Sustainability*, vol. 16, no. 4, Art. no. 1620, 2024, doi: 10.3390/su16041620.
- [14] S. Puajpanda, D. Mahapatra, S. Mishra, N. Padhy, and R. Panigrahi, "Enhancing predictive accuracy in IoT-based smart irrigation systems: A comparative analysis of advanced ensemble learning models and traditional techniques for soil fertility assessment," *Eng. Proc.*, vol. 87, no. 1, p. 65, 2025.
- [15] F. Pedregosa et al., "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825-2830, 2011.
- [16] Q. McNemar, "Note on the sampling error of the difference between correlated proportions or percentages," *Psychometrika*, vol. 12, no. 2, pp. 153-157, 1947.
- [17] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861-874, 2006.
- [18] R. J. A. Little and D. B. Rubin, *Statistical Analysis with Missing Data*, 3rd ed. Hoboken, NJ, USA: Wiley, 2019.
- [19] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (KDD)*, 2016, pp. 785-794.
- [20] H. Yan, F. Xie, D. Long, Y. Long, P. Yu, and H. Chen, "An accurate irrigation volume prediction method based on an optimized LSTM model," *PeerJ Computer Science*, vol. 10, Art. no. e2112, 2024.
- [21] S.-H. Park et al., "Development of a soil moisture prediction model based on recurrent neural network long short-term memory (RNN-LSTM) in soybean cultivation," *Sensors*, vol. 23, no. 4, Art. no. 1976, 2023.
- [22] T. T. Nguyen et al., "A low-cost approach for soil moisture prediction using multi-sensor data and machine learning algorithm," *Science of The Total Environment*, vol. 833, Art. no. 155066, 2022.
- [23] S. Basir, S. Noel, D. Buckmaster, and M. Ashik-E-Rabbani, "Enhancing subsurface soil moisture forecasting: A long short-term memory network model using weather data," *Agriculture*, vol. 14, no. 2, Art. no. 333, 2024.
- [24] R. Nagwanshi, "Machine learning-based smart irrigation for efficient water management: A literature review," *Int. Res. J. Modernization Eng. Technol. Sci.*, vol. 8, no. 4, pp. 12475-12481, Apr. 2026, doi: 10.56726/IRJ-METS96729.