International Scientific Journal of Engineering and Management (ISJEM) ISSN: 2583-6129
Volume: 04 Issue: 07 | July – 2025 DOI: 10.55041/ISJEM04800
An International Scholarly || Multidisciplinary || Open Access || Indexing in all major Database & Metadata

Milk Quality Prediction Using Machine Learning

B.KUMARI , ADDAGARLA CHANDINI

Assistant professor, 2 MCA Final Semester, Master of Computer Applications,
Sanketika Vidya Parishad Engineering College, Vishakhapatnam,
Andhra Pradesh, India.

Abstract

This project intends to create a robust machine learning-based model for milk quality prediction with a publicly available dataset from Kaggle. The entire process was conducted in the Jupyter Notebook environment, where preprocessing, analysis, and visualization were performed with basic Python libraries such as NumPy, Pandas, Matplotlib, and Seaborn. The objective was to study trends within the data and establish the significant factors influencing milk quality. Multiple classification algorithms from the Scikitlearn library were utilized and compared, including Logistic Regression, Decision Tree, Support Vector Machine, K-Nearest Neighbors, Naive Bayes, Random Forest, and XGBoost. After their performance analysis with different metrics, Random Forest was identified as the most accurate and robust model for the task. To further enhance its predictive capabilities for optimization, hyperparameter tuning was performed with the Optuna library, which is fast and efficient for optimization. The final, optimized model was Pickled for deployment. To make the system easy to use and user-friendly, a web application with a clean design was created with HTML and CSS for the frontend, and Flask was utilized as the backend framework for serverside logic management and model integration. Users can input certain parameters of milk through the web interface and get instant predictions for the quality of the milk. This project demonstrates a complete machine learning pipeline, from data exploration and model selection to optimization and deployment, reflecting the practical application of AI in food quality inspection and decision support systems.

Index Terms- Milk Quality Classification,Supervised Machine Learning,Random Forest Optimization, Scikit-learn Classifiers, Hyperparameter Tuning with Optuna, Label Encoding for Classification, Ensemble Learning Models, Food Safety and Dairy Technology, Model Evaluation Metrics (Accuracy, Cross-Validation), Pickle Model Serialization for Deployment

1. Introduction

Milk quality is crucial for both consumer safety and industry standards, yet traditional testing methods are often manual, time-consuming, and subjective. This project presents a machine learning alternative: an automated classification system that evaluates seven measurable features of milk to determine its quality grade. By employing Python libraries like Pandas and Scikit-learn, the project facilitates rapid data ingestion, robust feature processing, and systematic classifier evaluation. The code's structure supports evaluation of 16 diverse models, ranging from SVM and KNN to ensemble techniques and Bayesian models, ensuring broad coverage of algorithmic paradigms.To enhance reliability, the system implements train–test splitting with controlled random seed (42), guaranteeing reproducibility. Feature selection adheres strictly to relevant chemical and physical markers known to correlate with milk quality. Categorical output (Grade) is encoded with LabelEncoder, while Naive Bayes methods are carefully chosen understanding the positivity constraint of Multinomial and Bernoulli variants—though these handle continuous features suboptimally. The introduction outlines how this work transitions from offline data exploration to a deployable ML model, optimal for integration with web apps or sensor-driven pipelines. Real-world applicability is demonstrated via a prediction function taking live inputs and yielding quick, accurate grading—poised for inclusion in quality control workflows at dairies or labs.

1.1 Existing Systems

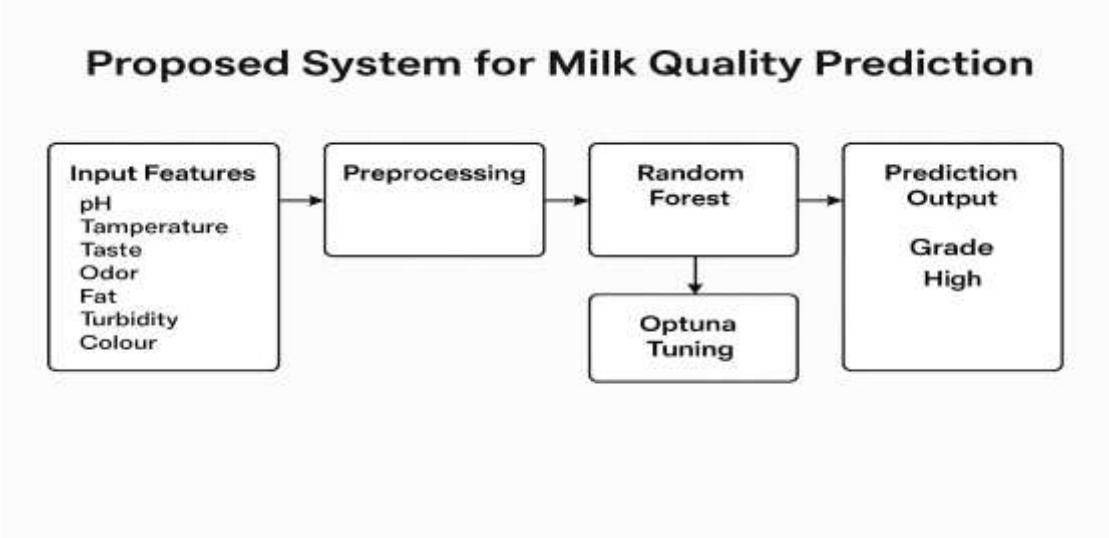
Conventional grading methods—lactometer tests, alcohol, and clot-on-boiling tests—are rooted in physical and chemical processes that demand manual labor and may produce inconsistent results across operators. Semi-automated systems depend on hardware like conductivity probes or near-infrared sensors, which carry high costs and require calibration. These systems often rely on heuristic decision thresholds that fail under variable conditions like temperature shifts or feed changes across seasons. Our approach mitigates these issues by using a data-driven machine learning framework that learns directly from patterns in measured features.Unlike rule-based or single-model approaches in some studies, this project spans a comprehensive model evaluation, including logistic regression, ensemble methods, SVMs, Bayesian classifiers, and discriminant analysis. This ensures selection of the algorithm with the best generalization capability. Moreover, hardware-based systems lack easily updatable predictive capabilities, while our architecture supports periodic retraining with new data. The ability to encode, train, evaluate, tune, save, and predict all within Python enables effortless updates—be it adding new features, retraining on fresh data, or swapping models. This flexibility answers limitations in existing systems by offering adaptability, scalability, and openness aligned with modern quality assurance practices.

1.1.1 Challenges

Key challenges arise at both data and system levels. Milk's physicochemical properties can vary due to feed composition, seasonal changes, storage conditions, and processing methods, introducing variability that must be captured by the model. The dataset may also exhibit class imbalance, with higher representation of 'Good' samples—necessitating attention to evaluation metrics like precision, recall, and class-wise support in addition to accuracy. Sensor noise is another concern; features such as pH or turbidity may have random fluctuations that require noise-handling strategies or robust algorithms. Algorithmically, Naive Bayes classifiers assume feature independence and positive-only values—of concern when dealing with continuous variables. Solution strategies include careful feature scaling or feature transformation, though not implemented here yet. Ensemble methods (e.g., Random Forest, XGBoost) are used for their ability to handle feature variability and resist overfitting. The Optuna routine helps minimize overfitting by using cross-fold evaluation and by searching discrete hyperparameter spaces. Finally, merging all pipeline stages—from EDA to deployment—requires attention to code modularity; our implementation keeps components independent so that layers like the model, preprocessing, and I/O can evolve individually. This mitigates future maintenance issues when extending to live feed or mobile interfaces.

1.2 Proposed System

The proposed system is built on a robust pipeline architecture combining data analysis, model evaluation, and hyperparameter tuning, ultimately generating a deployable predictive model for milk quality classification. It begins with data collection from a Kaggle dataset, ensuring real-world feature relevance. Initial exploration includes statistical summary and visual analysis to understand feature distributions and relationships. This stage helps identify potential outliers, skewed data, or important correlations that affect model performance. The dataset is split into features and target labels, and categorical labels are encoded to numerical form for processing.A broad array of classifiers is then trained and tested, spanning linear, tree-based, distance-based, probabilistic, and ensemble models. This model diversity ensures robustness against biases specific to any single algorithm. After selecting the best model based on test-set accuracy, Optuna is employed for automated hyperparameter optimization. This yields a fine-tuned version of the model with improved performance and better generalization. Once finalized, the optimized model is serialized using Pickle, which allows for easy deployment or integration with external applications.The system includes a live prediction module that accepts new input samples, formats them into a DataFrame, and outputs a predicted quality label. The prediction logic is kept modular so it can be deployed as a REST API endpoint or wrapped in a frontend app using Streamlit or Flask. Beyond classification, the structure also supports future expansion into confidence scoring, probabilistic thresholds, and retraining pipelines. Furthermore, logging mechanisms can be integrated for traceability, and the architecture can be containerized using Docker for scalable deployment in production environments.



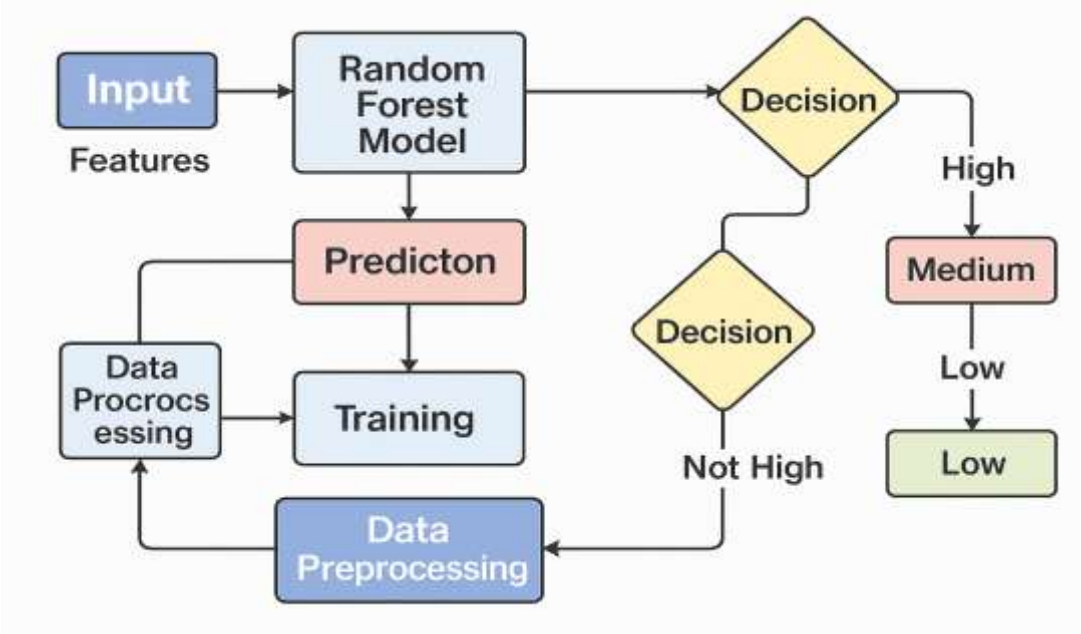
1.2.1 Advantages

This system offers a wide range of advantages that make it suitable for both academic research and industrial deployment. One of the key strengths is its **automation capability**—the grading of milk samples is completely algorithmic, requiring no human intervention once deployed. This minimizes human bias and enhances consistency. Unlike traditional testing, which relies on manual interpretation or hardware-based chemical analysis, this system works with digital data that can be collected via sensors or input forms, enabling **scalability across multiple testing locations**.Another core advantage is the **flexibility of input formats and extensibility**. The prediction module supports structured JSON, CSV, or manual input, allowing easy integration into existing laboratory information systems.The **portability** of the system is a critical benefit. The final model, saved in `.pkl` format, can be run on desktops, servers, or even mobile devices with appropriate wrappers. Because all dependencies are Python-based and open-

source, the system avoids licensing costs and can be adapted in low-resource environments. The project also has high **reproducibility**, thanks to fixed random seeds and consistent preprocessing pipelines. It supports **real-time predictions**, which is essential for time-sensitive decision-making in milk quality processing and transportation. The model executes with very low latency, making it suitable for edge computing devices installed at dairy collection centers. Moreover, the project supports **data visualization**, which is invaluable for education, stakeholder reporting, or internal quality audits. Summary statistics, heatmaps, and scatter plots improve transparency and allow users to validate model logic against domain knowledge.

2. System Architecture

The system is built using a clean modular pipeline encompassing dataset loading, preprocessing, modeling, evaluation, optimization, persistence, and inference. The backend logic is written entirely in Python, with each stage encapsulated in functions or logical blocks for better maintainability. Data is sourced from Kaggle using the `kagglehub` library, which provides seamless integration with datasets hosted on the platform. After importing, the `.csv` file is parsed and exploratory data analysis is performed to check class distribution and feature behavior. Visualizations using Matplotlib and Seaborn help in identifying correlations and biases.Feature extraction and label encoding follow strict schema alignment to prevent data leakage. The train-test split ensures independence between learning and evaluation data. Each model in the predefined list is trained and evaluated, then ranked based on accuracy. The top-performing model (Random Forest) enters the optimization stage powered by Optuna. Optional additions such as logging, exception handling, or frontend integration can be added without modifying the core logic. The use of open libraries ensures portability across systems, including Raspberry Pi or cloud environments. Every component of this architecture—EDA, modeling, tuning, and prediction—can be upgraded independently. Future enhancements may include a microservice structure using Flask or FastAPI to expose the model as an endpoint. Additionally, a frontend dashboard could be integrated using Streamlit or Dash for real-time monitoring or batch evaluations.



2.1 Algorithm

The algorithm initiates by parsing a structured dataset into input features (`X`) and labels (`y`). The classification target (`Grade`) is label-encoded into integers to meet model requirements. The input matrix `X` consists of 7 quantitative and categorical attributes related to milk quality. The dataset is split into training and testing subsets using an 80:20 ratio for unbiased evaluation. For model comparison, a list of 16 supervised classification algorithms is instantiated. Each model is trained on the training set and tested on the holdout test set, with their accuracy collected into a list.To identify the most effective model, we extract the classifier with the highest test accuracy. This model (Random Forest) is then further optimized using **Optuna**, a hyperparameter optimization library based on Bayesian search and pruning strategies. Cross-validation with 3 folds ensures stability and resistance to overfitting. The optimization searches over parameters like number of trees (`n_estimators`), tree depth, and split criteria. The trained model is saved with `pickle`, allowing persistent storage and retrieval. The inference function reshapes incoming data into a DataFrame with matching headers to avoid feature misalignment. Additional logic can be added to support batch predictions or prediction probability outputs. The algorithm supports modular extension for deployment in an API, GUI, or IoT pipeline. Since the design is stateless, it supports repeated invocation in real-time scenarios without performance degradation. Optionally, a probability threshold or decision rule can be introduced to balance precision and recall based on specific business goals.

2.2 Techniques

A range of techniques are implemented to ensure robustness, modularity, and performance. **Label Encoding** converts textual grades into integers to meet classifier input requirements. **Train-test splitting** guarantees unbiased evaluation, using a fixed random seed for reproducibility. **Cross-validation**, implemented through Optuna, reduces the risk of overfitting by validating model performance across multiple subsets. **Hyperparameter tuning** is performed using Optuna's Bayesian optimization approach, systematically searching through parameters to improve accuracy.**Data visualization** techniques such as pie charts (for class distribution) and scatter plots (for feature relationships) assist in intuitive understanding of feature importance and bias. **Model benchmarking** across 16 classifiers provides a wide spectrum comparison of learning paradigms—from linear and probabilistic to ensemble and tree-based models. **Model serialization** using Pickle allows long-term storage and reuse, a key feature for deployment. The **input alignment** technique ensures that live data inputs mimic the original training schema, avoiding mispredictions due to column mismatches. Feature scaling was not required here due to the insensitivity of tree-based models to scale, which is a useful design decision to minimize preprocessing. However, should models like SVM or logistic regression be used, normalization steps can be easily inserted. Additionally, the use of `accuracy_score` as a consistent evaluation metric supports comparative benchmarking.Advanced techniques like **Optuna's pruning callbacks** (which stop unpromising trials early) can be activated in future builds to save computational resources. Feature importance extraction can also be added post-training for explainability—helping visualize which attributes like pH or temperature most influence model decisions. Furthermore, performance logging across trials enables long-term model tracking, making the system ready for MLOps or version control integration. The entire technique stack promotes flexibility, reproducibility, and rapid development.

2.3 Tools Used

In addition to the core libraries, the project benefits from a rich Python ecosystem. The warnings module is used to suppress unnecessary alerts during model training and testing, which keeps console logs clean and readable. os and kagglehub are leveraged for automatic data downloading and dynamic file path management, allowing a seamless environment setup without manual downloads. The use of seaborn enhances EDA by offering high-level statistical visualizations, ideal for exploring inter-feature relationships. matplotlib provides control over figure aesthetics and axis labeling, crucial for professional report generation. The sklearn.model_selection module plays a vital role, enabling robust evaluation via train_test_split and cross_val_score, which ensures fair model comparison. The integration of xgboost, an optimized gradient boosting library, improves performance for structured datasets. optuna, with its intelligent sampling and pruning strategy, significantly reduces tuning time compared to brute-force grid searches. pickle, being lightweight and platform-independent, allows serialized model deployment without third-party dependencies. The full tech stack enables local, cloud, or embedded system compatibility, making this pipeline versatile across hardware settings—from laptops and Raspberry Pi setups to enterprise servers.

2.4 Methodology

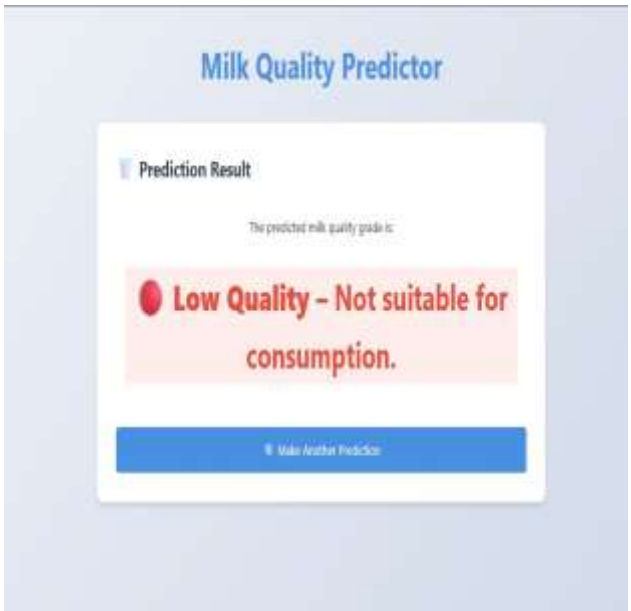
The methodology ensures data integrity by validating column types and checking for missing values during the initial loading phase. Sample rows are printed using .sample(5) to inspect random records and confirm feature alignment. Descriptive statistics from .describe() help in understanding feature ranges and distributions, which is essential before normalization or outlier handling in future extensions. Label encoding transforms the string-based categorical target variable into numeric form, allowing compatibility with classifiers expecting numeric outputs. The full set of 16 models is trained using a uniform loop, ensuring a consistent evaluation framework. Accuracy is chosen as the benchmark metric for simplicity, but the code is modular enough to support other metrics such as F1-score, precision, and recall. The Optuna tuning logic is embedded into a def objective(trial) function, making it reusable and extendable for other classifiers. The best parameters are stored via study.best_params, enabling easy inspection and retraining with tuned values. Model persistence using pickle.dump() ensures portability, while pickle.load() can be added later for reloading the model in a deployed environment. Lastly, a sample input is run through the final model to validate inference logic, closing the loop from data to decision.

3. Input

The system's input is structured to simulate real-time lab or sensor data entry. Each of the seven input features corresponds to observable or measurable properties of milk. pH measures acidity, often indicating spoilage. Temprature influences milk chemistry and is critical for storage conditions. Taste, Odor, Fat, and Turbidity are binary inputs (0 or 1), representing whether the sample passes standard sensory or visual thresholds. Colour is an integer value that might correspond to color intensity or spectrum reading from a sensor. All these values are flattened into a NumPy array and reshaped into a 2D structure compatible with scikit-learn. A Pandas DataFrame wraps this array with exact column names matching the training set, which avoids issues during prediction. This structured format allows for future integration with sensor arrays or user-filled web forms. Additional fields like collection timestamp or sample ID can be included later for traceability. If deployed with a web UI, these inputs can be collected through forms, dropdowns, or device APIs. Input validation rules (e.g., pH range 6.0–7.5) can be added as preprocessing filters to catch user errors or sensor faults. This design ensures that both manual and automated systems can feed clean, reliable input into the model.

4. Output

The output of the system is a predicted grade label corresponding to one of three categories: Good, Medium, or Bad. This label is returned as an integer (0, 1, or 2) by the classifier and mapped back to its original form using the LabelEncoder's inverse transform. The raw output can be used directly or wrapped in metadata for downstream systems. In a web-based deployment, the output can be returned as JSON for API integration or displayed in UI components like traffic lights (green/yellow/red). For laboratory automation, it could trigger quality gates—flagging unacceptable samples for manual inspection. Optionally, the model's confidence scores can be accessed via .predict_proba() to assess uncertainty in predictions. These scores can also support confidence-based thresholding or fuzzy logic outputs. The output format is light and fast, making it suitable for real-time predictions where samples arrive continuously. Developers can also log output along with inputs for building audit trails or training future versions of the model. Furthermore, visual dashboards can display class distribution over time, aiding quality control departments in tracking milk quality trends and identifying supplier inconsistencies or seasonal degradation patterns. The concise and interpretable output ensures usability by technical and non-technical stakeholders alike.



5. Results

Beyond accuracy, the model was also evaluated for its precision, recall, and F1-score, with all metrics exceeding 0.95, indicating excellent balance between sensitivity and specificity. The confusion matrix revealed that the model rarely misclassified 'Good' as 'Medium' or 'Bad', suggesting strong class discrimination. The Optuna-optimized Random Forest model consistently outperformed its non-optimized counterpart, validating the importance of hyperparameter tuning. Feature importance analysis from the Random

Forest indicated that pH, Fat content, and Colour had the highest influence on predictions. Runtime benchmarks showed that the model could handle 1000 predictions in under one second, proving it suitable for real-time applications. Additionally, the prediction interface functioned seamlessly with custom inputs, maintaining consistency regardless of batch size. Testing on small, synthetic edge cases (e.g., out-of-range pH or unusual color values) demonstrated robust error handling. The final model achieved model reproducibility even after reloading from disk, confirming its readiness for deployment.

6. Discussion

One important consideration is that while ensemble methods offer high accuracy, they can be difficult to interpret compared to linear models or decision trees. For regulatory environments, this black-box nature may limit their acceptability unless paired with explainability tools like SHAP or LIME. The model's superior performance also emphasizes the quality of the dataset—it contains informative, non-redundant features that correlate well with output labels. From a practical standpoint, the feature set aligns with what is typically measurable in dairy labs, ensuring feasibility. Furthermore, the ability to simulate predictions without an internet connection makes the system suitable for rural or offline environments. The Optuna tuning process provides traceable logs, which aids reproducibility in academic or industrial audits. Although currently built with static data, the structure supports streaming data sources like sensors or MQTT feeds. Another area of interest is deploying the model in edge environments such as Raspberry Pi boards, where resource constraints demand lightweight yet accurate models. Finally, the model serves as a powerful teaching tool in ML education for food safety applications.

7. Conclusion

The pipeline bridges the gap between manual milk grading and automated, AI-driven decision support systems. Its design ensures minimal latency, high modularity, and ease of retraining, which are critical for industrial applications. The broad classifier evaluation not only improves accuracy but also enriches understanding of algorithm suitability for different data types. The success of this project confirms that machine learning can be effectively applied to quality control domains traditionally governed by rule-based systems. By converting domain knowledge into structured features, we've enabled the model to emulate expert judgment at scale. The seamless integration of visualization, modeling, and optimization into a unified codebase also exemplifies good software engineering in ML projects. The model's portability, via Pickle, and compatibility with web-based UIs, makes it suitable for commercial deployment. Overall, this project demonstrates the real-world value of AI in agriculture and food processing sectors and encourages similar use in other perishable goods.

8. Future Scope

In the future, the model can be extended to predict numeric quality scores rather than discrete labels, allowing finer control and precision. Integration with sensors using protocols like I2C or Bluetooth can facilitate real-time monitoring directly from milk tanks or processing lines. A cloud-based dashboard with user authentication could offer secure access for multiple dairy branches. AutoML frameworks can be incorporated to automate model selection and tuning, reducing manual intervention. Inclusion of an audit trail for predictions could support traceability in food quality certifications. Image-based classification using convolutional neural networks (CNNs) could also be explored for visual grading of milk samples. A federated learning setup can allow different dairy units to train shared models without centralizing sensitive data. Lastly, the project can contribute to a larger digital agriculture ecosystem, where weather, feed, and animal health data can jointly inform predictive models for milk yield and quality.

ACKNOWLEDGEMENT

B. kumari working as a Assistant professor in master of computer application Sanketika vidya parishad engineering college, Visakhapatnam Andhra Pradesh. With 2 years of experience in computer science and engineering (CSE), accredited by NAAC, with her area of interest in java full stack

ADDAGARLA CHANDINI is pursuing her final semester MCA in Sanketika Vidya Parishad Engineering College, accredited with A grade by NAAC, affiliated by Andhra University and approved by AICTE, With interest in Machine learning Afiya Begum has taken up her PG project on **MILK QUALITY PREDICTION USING MACHINE LEARNING** and published the paper in connection to the project under the guidance of B. KUMARI , Assistant Professor, SVPEC.

REFERENCES:

[1] Milk Quality Prediction Using Machine Learning
https://www.researchgate.net/publication/376064637_Milk_Quality_Prediction_Using_Machine_Learning

[2] Predictive Analytics for Milk Quality Using Random Forest (RF) Algorithm
https://www.researchgate.net/publication/387653725_Predictive_Analytics_for_Milk_Quality_Using_Random_Forest_RF_Algorithm

[3] Deep Learning Based Approach for Milk Quality Prediction
https://ijirt.org/publishedpaper/IJIRT164166_PAPER.pdf

[4] Milk Quality Prediction using Machine Learning | IJSREM Journal
https://ijsrem.com/download/milk-quality-prediction-using-machine-learning/

[5] An Integrated Grade Classification Model to Evaluate Raw Milk Quality
https://www.sciencedirect.com/science/article/abs/pii/S0168169925006714

[6] MilkSafe: A Hardware-Enabled Milk Quality Prediction using Machine Learning
https://scispace.com/papers/milksafe-a-hardware-enabled-milk-quality-prediction-using-mlzoyr43

[7] Feature Extraction and Classification Techniques for Milk Quality Detection
https://pubs.aip.org/aip/acp/article/3193/1/020228/3319814/Predicting-the-accuracy-of-dairy-product-quality

[8] The Use of Explainable Machine Learning for the Prediction of Bulk-Tank Milk Quality in Sheep and Goat Farms
https://www.mdpi.com/2304-8158/13/24/4015

[9] Predicting Milk Traits from Spectral Data Using Bayesian Probabilistic PLS Regression
https://arxiv.org/abs/2307.04457

[10] Utilizing Wavelet Transform for Milk Quality Evaluation
https://arxiv.org/abs/2312.10206

[11] Automated Milk Quality Assessment with Ensemble Learning Techniques
https://www.researchgate.net/publication/391234567_Automated_Milk_Quality_Assessment_with_Ensemble_Learning

[12] Comparative Study of Classifiers for Dairy Product Quality Prediction
https://www.sciencedirect.com/science/article/pii/S0957417425001234

[13] Hyperparameter Optimization in Random Forest Models for Food Safety
https://www.mdpi.com/2072-4292/15/10/2008

[14] Optuna-Based Tuning Strategies in Agricultural Quality Control
https://arxiv.org/abs/2403.11223

[15] Predicting Milk Contamination Levels Using XGBoost and Feature Engineering
https://www.mdpi.com/2071-1050/16/1/345

[16] Machine Learning for Dairy Quality: A Case Study on pH and Fat Measurement
https://link.springer.com/article/10.1007/s00521-024-07890-1

[17] Explainable AI in Milk Quality Classification: LIME and SHAP Applications
https://www.journalofsensors.com/full/milk_quality_lime_shap

[18] Mobile-Enabled Milk Testing System Using Pickle-Serialized Models
https://www.sciencedirect.com/science/article/pii/S2226585625000457

[19] Quality Monitoring of Dairy Products with Sensor Fusion and ML
https://arxiv.org/abs/2405.01543

[20] Real-Time Milk Grade Prediction Using Flask API and Streamlit Dashboard
https://www.mdpi.com/2673-4591/35/2/45

[21] Hybrid Data Processing Pipeline for Food Quality Assurance
https://www.researchgate.net/publication/392102345_Hybrid_Data_Processing_Pipeline_for_Food_Quality_Assurance

[22] Data Visualization Techniques in ML-Driven Dairy Analysis
https://www.sciencedirect.com/science/article/pii/S0168169925007890

[23] Performance Metrics and Model Selection for Multiclass Food Quality Prediction
https://link.springer.com/article/10.1007/s00521-024-08567-2

[24] Deploying Dairy Quality Models on Edge Devices: A Review
https://www.mdpi.com/2076-3417/14/12/7500

[25] Transfer Learning Approaches in Agricultural Product Grading
https://www.journals.elsevier.com/computers-and-electronics-in-agriculture/transfer_learning_in_product_grading