Internship Project Report

**Name:** Ahmad Zakariya  
**Course & Department:** B.Tech in Mechanical Engineering  
**Internship Duration:** 9 June – 27 July 2025  
**Internship Venue:** Engineering Design & Implementation Club, AMU  
**Mentor:** Junaid Ali Reshi, Assistant Professor.

## Declaration

I hereby declare that the work presented in this report entitled “Internship Project Report – Educational Internship under EDIC, AMU” is the result of my own independent effort and sincere dedication. The tasks, assignments, and mini-project documented herein were undertaken by me as part of the educational internship program organized by the Engineering Design & Implementation Club (EDIC), Aligarh Muslim University (AMU), under the able mentorship of Dr. Junaid Ali Reshi, Assistant Professor, Department of Mechanical Engineering.

All data, code implementations, models, and analyses are original unless explicitly stated otherwise. Any content, references, or material used from external sources has been properly cited and acknowledged in accordance with academic and ethical standards. This report has not been submitted in part or in full for the award of any other degree, diploma, certificate, or academic recognition elsewhere.

This declaration is made with a complete understanding of the importance of academic honesty and integrity, and I fully accept the responsibility for the contents of this report.

Date: 28 July 2025  
Place: Aligarh  
(Ahmad Zakariya)  
B.Tech in Mechanical Engineering  
Department of Mechanical Engineering  
Aligarh Muslim University

Contents

[Declaration 2](#_Toc204602659)

[Table of Contents 3](#_Toc204602660)

[Objectives of the Internship 4](#_Toc204602661)

[Description of Work/Project Undertaken 5](#_Toc204602662)

[Tools, Techniques, and Methodology Used 7](#_Toc204602663)

[Learning Outcomes 8](#_Toc204602664)

[Challenges and Resolutions 9](#_Toc204602665)

[Contributions and Impact 10](#_Toc204602666)

[Conclusion 11](#_Toc204602667)

## Objectives of the Internship

* To gain practical experience in machine learning by implementing algorithms from first principles.
* To develop programming proficiency in Python and its libraries for data analysis and modeling.
* To understand the theory and working of regression, decision trees, K-nearest neighbors, and ensemble classifiers.
* To learn data preprocessing techniques (handling missing data, encoding categorical features, feature scaling) to prepare datasets for modeling.
* To evaluate and compare model performance using appropriate metrics and train-test splits.

## Description of Work/Project Undertaken

During the internship, I completed a series of assignments and a mini project focusing on machine learning tasks. Each assignment involved applying theoretical concepts to practical coding exercises, as described below:

* **Assignment 1 – Simple & Polynomial Regression (from scratch):** I implemented linear regression and polynomial regression models in Python without using high-level libraries. This required coding the cost (mean squared error) and gradient descent algorithm manually. Gradient descent is an optimization algorithm used to iteratively adjust model parameters (slope and intercept) in order to minimize the error between predicted and actual values[[2]](https://www.geeksforgeeks.org/machine-learning/gradient-descent-in-linear-regression/#:~:text=Gradient%20descent%20is%20a%20optimization,more%20about%20Gradient%20Descent%20and). Implementing this from scratch deepened my understanding of how the line of best fit is found and how nonlinear polynomial terms affect model complexity[[3]](https://www.geeksforgeeks.org/machine-learning/gradient-descent-in-linear-regression/#:~:text=For%20simple%20linear%20regression%2C%20we,become%20computationally%20expensive%20due%20to). The final outcome was fitting and plotting both linear and polynomial regression models on sample data, and computing error metrics to verify accuracy.
* **Assignment 2 – Decision Tree for Binary Classification (Titanic Dataset):** I built a decision tree classifier to predict survival (binary label) on the Titanic dataset. The data was split into training and testing sets with ratios 50:50, 70:30, and 80:20. After training the model on each split, I evaluated performance using standard classification metrics: accuracy, precision, recall, F1 score, and confusion matrix. The results showed how changing the train-test split affects metrics. (Decision tree classifiers in scikit-learn can handle both binary and multiclass problems[[4]](https://scikit-learn.org/stable/modules/tree.html#:~:text=DecisionTreeClassifier%20is%20capable%20of%20both,1%5D%29%20classification).) This assignment helped me understand how decision rules segment the feature space and the importance of evaluating a model on unseen data.
* **Assignment 3 – Preprocessing + Classification:** I performed comprehensive preprocessing on a classification dataset before model building. This involved imputing missing values, encoding categorical variables as numeric, and scaling features to a common range. I then applied a logistic regression classifier (using scikit-learn) to the preprocessed data. After preprocessing, the model achieved significantly better accuracy than on the raw data, demonstrating the importance of clean, well-formatted data. Many machine learning algorithms require such preprocessing steps – for example, unlike decision trees, methods like logistic regression typically need normalization and dummy variables[[5]](https://scikit-learn.org/stable/modules/tree.html#:~:text=,Trees%20can%20be%20visualized).
* **Assignment 4 – Comparison: Raw vs Pre-processed Data:** I compared model performance on raw data versus preprocessed data using the same classification algorithm. This comparison involved detailing each preprocessing step (imputation, encoding, scaling) and evaluating performance metrics for both datasets. The analysis clearly showed that preprocessing (handling missing values, encoding, scaling) significantly improved accuracy and other metrics. These findings underscore that data cleaning is a critical part of any machine learning pipeline, as many algorithms assume the input data is numeric and scaled[[5]](https://scikit-learn.org/stable/modules/tree.html#:~:text=,Trees%20can%20be%20visualized).
* **Assignment 5 – Performance on Various Splits (Raw vs Preprocessed):** I extended the previous comparison by using different data splitting strategies. Specifically, I compared model performance using 70:30 and 80:20 train-test splits, and also implemented a train-validation-test split. Models were trained and evaluated on both raw and preprocessed versions of the data. The results indicated that preprocessing consistently improved generalization to unseen data, especially when using a separate validation set to tune hyperparameters. This assignment helped me understand the risks of data leakage and the importance of an unbiased validation process when assessing model performance.
* **Assignment 6 – K-Nearest Neighbors (from scratch):** I implemented the K-Nearest Neighbors algorithm manually for classification. KNN is a non-parametric, instance-based learning method that classifies a new sample by looking at the ‘k’ nearest training points in feature space and taking a majority vote[[6]](https://www.geeksforgeeks.org/machine-learning/k-nearest-neighbours/#:~:text=K,based%20learning%20method). I coded distance calculations for Euclidean, Manhattan, and Minkowski metrics, and tested different values of k (1, 3, 5, 7, 10, N). The classifier’s performance was evaluated via confusion matrix and standard metrics. The experiments showed how the choice of k and distance metric affects accuracy: larger k tends to smooth out noise but can oversimplify, whereas smaller k can be sensitive to outliers[[6]](https://www.geeksforgeeks.org/machine-learning/k-nearest-neighbours/#:~:text=K,based%20learning%20method)[[7]](https://www.geeksforgeeks.org/machine-learning/k-nearest-neighbours/#:~:text=KNN%20uses%20distance%20metrics%20to,we%20use%20below%20distance%20metrics).
* **Mini Project – Cart Abandonment Prediction in E-Commerce:** For the mini project, I analyzed an e-commerce transaction dataset to model cart abandonment behavior. First, I applied association rule mining (using the Apriori algorithm) to discover frequent itemsets and understand purchasing patterns. Apriori is a classic algorithm to find groups of items that frequently occur together, which is useful in market basket analysis[[8]](https://www.geeksforgeeks.org/machine-learning/apriori-algorithm/#:~:text=Apriori%20Algorithm%20is%20a%20basic,valuable%20in%20market%20basket%20analysis). These insights helped in feature engineering. Then, I trained an XGBoost classifier (an ensemble gradient-boosting algorithm[[9]](https://www.analyticsvidhya.com/blog/2018/09/an-end-to-end-guide-to-understand-the-math-behind-xgboost/#:~:text=XGBoost%20is%20a%20machine%20learning,case)) to predict whether a user would abandon or complete a purchase. The best model achieved ~51% accuracy. Although this accuracy is modest, the project provided valuable lessons: high-quality data and thoughtful feature design often matter more than model complexity. In particular, the exercise illustrated end-to-end machine learning workflow: data exploration, pattern mining (Apriori), data cleaning, feature selection, model training, and evaluation.

## Tools, Techniques, and Methodology Used

* **Programming Language:** Python was used for all tasks, leveraging its scientific libraries for data handling and visualization.
* **Libraries:** NumPy and Pandas for numerical computations and data manipulation; Matplotlib and Seaborn for plotting; scikit-learn for implementing algorithms (except where explicitly coded from scratch); MLxtend for Apriori implementation; XGBoost library for gradient boosting.
* **Implementation Approach:** Where required (assignments 1, 2, 6), algorithms were coded from first principles to reinforce understanding of underlying mathematics (for example, coding gradient descent loops and distance calculations). Otherwise, established libraries were used for efficiency and to handle larger datasets (e.g., scikit-learn’s DecisionTreeClassifier, XGBoost).
* **Data Splitting:** For model evaluation, datasets were divided into training and testing sets using various splits (50:50, 70:30, 80:20) as appropriate. For the final project, a train-validation-test scheme was also employed to tune model parameters without overfitting.
* **Model Evaluation:** Performance was measured using confusion matrices and classification metrics (accuracy, precision, recall, F1 score). Understanding these metrics was crucial for interpreting how well models handled imbalanced outcomes (e.g., survival vs. death).

No external preprocessing tools or software were used beyond the Python ecosystem. The methodology emphasized iterative testing: for example, refining data cleaning steps when initial model performance was low, or adjusting k and distance metrics in KNN until reasonable performance was achieved.

## Learning Outcomes

* **Deepened Understanding of ML Algorithms:** Implementing regression and classification algorithms from scratch reinforced my conceptual grasp of how cost functions and optimization work. For instance, I saw firsthand how gradient descent iteratively minimizes the mean squared error in linear and polynomial regression[[2]](https://www.geeksforgeeks.org/machine-learning/gradient-descent-in-linear-regression/#:~:text=Gradient%20descent%20is%20a%20optimization,more%20about%20Gradient%20Descent%20and).
* **Programming and Debugging Skills:** Writing algorithms manually improved my coding skills in Python and taught me to carefully debug mathematical implementations (such as ensuring correct gradient calculations or distance computations).
* **Appreciation of Data Preprocessing:** I learned that real-world data often needs significant cleaning before modeling. Handling missing values, categorical encoding, and scaling significantly improved model accuracy, confirming that many algorithms (unlike decision trees) expect normalized numeric input[[5]](https://scikit-learn.org/stable/modules/tree.html#:~:text=,Trees%20can%20be%20visualized).
* **Model Evaluation Insights:** Through the decision tree and KNN assignments, I understood classification evaluation metrics. Changing train-test splits revealed trade-offs between bias and variance. For example, larger training sets generally improved accuracy, but using a validation set was crucial to detect overfitting.
* **End-to-End Project Experience:** The mini project gave me insight into an end-to-end machine learning pipeline: starting from exploratory data analysis (using association rule mining to uncover patterns[[8]](https://www.geeksforgeeks.org/machine-learning/apriori-algorithm/#:~:text=Apriori%20Algorithm%20is%20a%20basic,valuable%20in%20market%20basket%20analysis)), through feature engineering, to training and interpreting a predictive model (XGBoost)[[9]](https://www.analyticsvidhya.com/blog/2018/09/an-end-to-end-guide-to-understand-the-math-behind-xgboost/#:~:text=XGBoost%20is%20a%20machine%20learning,case). I learned to interpret the limitations of the model (e.g., why accuracy might be limited by data quality) and to communicate these findings clearly.

Overall, the internship significantly enhanced both my technical skills (in coding and ML theory) and my soft skills (such as problem solving and self-learning by consulting documentation and literature).

## Challenges and Resolutions

* **Implementing Algorithms from Scratch:** Writing algorithms like gradient descent and KNN manually was initially challenging due to the mathematical detail involved. I overcame this by breaking the problem into smaller parts (e.g., coding cost calculation separately) and by carefully visualizing intermediate results. Reviewing textbooks and online tutorials helped clarify each step’s purpose.
* **Data Preprocessing Decisions:** Determining the best way to encode categorical features or choose imputation values required experimentation. I addressed this by trying multiple approaches (e.g., label encoding vs one-hot encoding) and using visualization to check for any anomalies introduced.
* **Model Overfitting:** In several models, especially with limited data, overfitting was a concern. To mitigate this, I used techniques like cross-validation and regularization (e.g., the built-in regularization in XGBoost[[9]](https://www.analyticsvidhya.com/blog/2018/09/an-end-to-end-guide-to-understand-the-math-behind-xgboost/#:~:text=XGBoost%20is%20a%20machine%20learning,case)).
* **Computational Constraints:** Some computations (e.g., running Apriori on a large itemset) were slow. I dealt with this by limiting support thresholds and by using optimized library functions (MLxtend) where possible.

Throughout the internship, I resolved challenges by iterative testing, consulting academic and technical resources, and seeking guidance when necessary. These experiences improved my resilience and problem-solving skills, which are valuable outcomes of the internship.

## Contributions and Impact

This internship allowed me to contribute to EDIC’s mission of fostering technical knowledge among students. By developing and documenting these machine learning implementations, I added practical learning examples that could benefit future club members. The project on cart abandonment, though modest in predictive accuracy, provided useful insights into e-commerce user behavior and demonstrated the real-world applicability of rule mining and machine learning. On a personal level, the work I accomplished strengthens the academic community by bridging theoretical knowledge and practical application. The skills and code developed here can serve as a reference or starting point for peers interested in machine learning projects within the department.

## Conclusion

In summary, the internship at the Engineering Design & Implementation Club was an enriching educational experience. I achieved the original objectives by gaining hands-on experience with multiple machine learning techniques and by understanding the importance of data preparation in modeling. Each task, from coding regression with gradient descent[[2]](https://www.geeksforgeeks.org/machine-learning/gradient-descent-in-linear-regression/#:~:text=Gradient%20descent%20is%20a%20optimization,more%20about%20Gradient%20Descent%20and) to implementing KNN[[6]](https://www.geeksforgeeks.org/machine-learning/k-nearest-neighbours/#:~:text=K,based%20learning%20method) and using association rule mining[[8]](https://www.geeksforgeeks.org/machine-learning/apriori-algorithm/#:~:text=Apriori%20Algorithm%20is%20a%20basic,valuable%20in%20market%20basket%20analysis), contributed to a comprehensive learning journey. The process of debugging algorithms and evaluating models has improved my analytical abilities and prepared me for more advanced projects. Overall, this internship has strengthened my technical foundation in machine learning and data analysis, and I look forward to applying these skills in future academic and professional endeavors.

*This report is a complete and honest record of my internship activities and learning.*