LONDON METROPOLITAN UNIVERSITY

informatics college • pokhara

Module Code & Module Title

CU6051NP Artificial Intelligence

25% Individual Coursework

Submission: Final Submission

Academic Semester: Autumn Semester 2025

Credit: 15 credit semester long module

Student Name: Shusil Bk

London Met ID: 23048939

Assignment Due Date: 07/01/2026.

Assignment Submission Date: 21/01/2026

Submitted To: Jeevan Prakash Pant

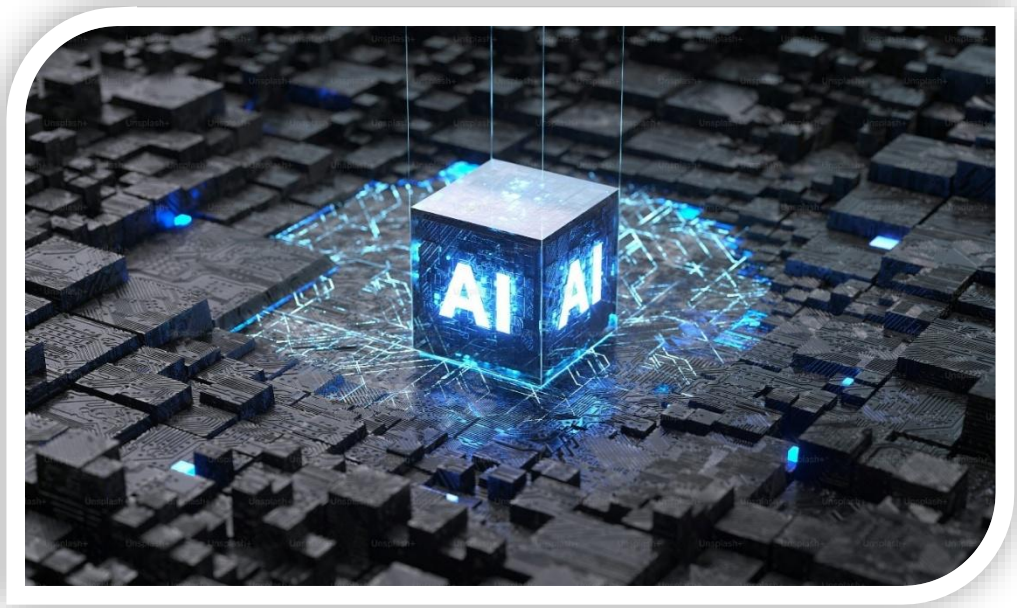
I confirm that I understand my coursework needs to be submitted online via MST Classroom under the relevant module page before the deadline for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a mark of zero will be awarded

Table of Contents

Table of Figures

1.Introduction

Overview of Artificial learning platforms



Fascinatingly, we have 3 primary learning approaches in Artificial Intelligence. We have been taught these 3 in our college module and I have also done some research on it in different platforms like Github,Youtube, websites, etc.

1.1.1 Supervised learning

Supervised learning is a paradigm of machine learning in which the algorithms are trained on labeled data, both the input features and the correct output labels (: Jiang, 2020). Learning is based on these labeled examples which are solved relative to the mapping of the inputs to the outputs such that the model can make acceptable predictions using new unknown data. It is very useful when the problem under consideration is classification and regression and where there is a known ground-truth to train and test.

Key Characteristics:

- Labeled data: Data is fed by means of inputs and known correct outputs.
- Pattern recognition: Algorithm discovers input-output patterns.
- Two process Training (learn) + Prediction (apply knowledge).
- Performance analysis: Test labels vs. accuracy.

1.1.2 Unsupervised Learning

Unsupervised learning is a machine learning approach in which algorithms are used to uncover hidden patterns, structures or relationships within data that are not explicitly labeled to have correct answers. Statistical analysis and measures of similarity are used to recognize natural groupings, clusters, or associations in the data to explore unknown distributions of data through this model (: Miguel-Diez, 2026).

Key Characteristics

- No correct answers were given to the unlabeled training data.
- Pattern discovery: The clusters/groups are discovered automatically.
- Exploratory analysis: Uncovers the concealed data forms.
- None: Model learns form on his own.

1.1.3 Semi-supervised learning

Semi-supervised learning or hybrid learning involves the incorporation of a mix of concepts of excelling and un-excelling learning within a single methodology. It trains a model with a small quantity of labeled data with large quantity of unlabeled data which is more efficient and effective compared to training the model using labeled data only. This is useful in cases where labels are costly or tedious to obtain, whereas raw data is simple to obtain (Netshamutshedzi, 2025).

Key characteristics:

- Accepts both labeled and unlabeled data during training.
- It begins with a supervised learning on labeled data, which is enhanced with unlabeled data.
- Streams sometimes take unsupervised steps (such as clustering) to guide or refine the supervised model.
- Convenient in a situation where there is a shortage of labels, but there is still a need to achieve good performance.

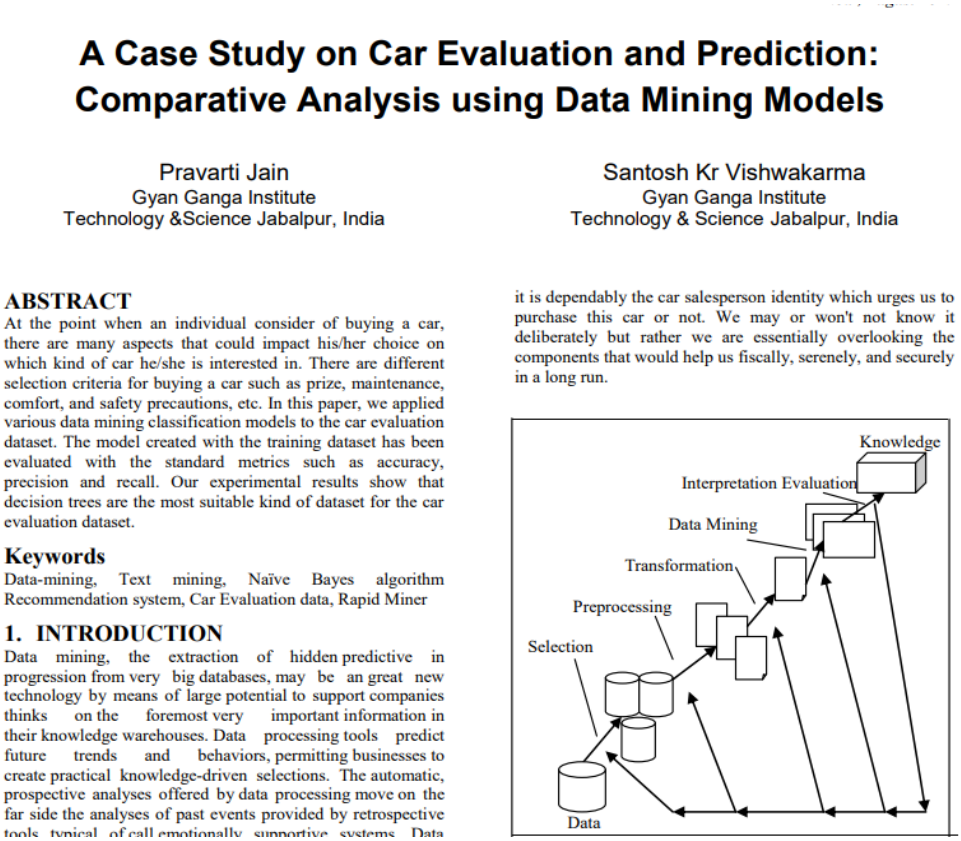
1.2 Introduction to the problem domain

The Car Evaluation dataset is a multi-criteria decision-making problem where the vehicle acceptability of different buyer groups is assessed, since each group has its own preferences. The whole thing stems from the real-world experiences of buying a car, and a decision tree has been proposed to represent the process of identifying an acceptable car by reducing the number of possible outcomes to be evaluated. These outcomes are categorized as unacceptable, acceptable, good, or very good. Traditional evaluations often involve expert opinions, which leads to differences in the assessments for different markets—a practice this dataset counteracts by turning rule-based classifications into flat, categorical attributes suitable for objective analysis.

The dataset serves the purpose of testing classification algorithms on balanced, categorical data with no missing values; thus, it is perfect for the benchmarks of decision trees or rule learners. It is like the use of cases in the areas of automotive quality assessment, predictive maintenance, and consumer recommendation engines; thus, it is a synthetic but realistic proxy for interpretable AI in vehicle grading. For your documentation, underscore its importance for decision support systems and structured problem-solving in software projects

2. Background

2.1 Jain et al. (IJCA 2017) - A Case Study on Car Evaluation and Prediction



A Case Study on Car Evaluation and Prediction: Comparative Analysis using Data Mining Models

Pravarti Jain
Gyan Ganga Institute
Technology &Science Jabalpur, India

Santosh Kr Vishwakarma
Gyan Ganga Institute
Technology & Science Jabalpur, India

ABSTRACT
At the point when an individual consider of buying a car, there are many aspects that could impact his/her choice on which kind of car he/she is interested in. There are different selection criteria for buying a car such as prize, maintenance, comfort, and safety precautions, etc. In this paper, we applied various data mining classification models to the car evaluation dataset. The model created with the training dataset has been evaluated with the standard metrics such as accuracy, precision and recall. Our experimental results show that decision trees are the most suitable kind of dataset for the car evaluation dataset.

Keywords
Data-mining, Text mining, Naïve Bayes algorithm Recommendation system, Car Evaluation data, Rapid Miner

1. INTRODUCTION
Data mining, the extraction of hidden predictive in progression from very big databases, may be an great new technology by means of large potential to support companies thinks on the foremost very important information in their knowledge warehouses. Data processing tools predict future trends and behaviors, permitting businesses to create practical knowledge-driven selections. The automatic, prospective analyses offered by data processing move on the far side the analyses of past events provided by retrospective tools typical of call emotionally supportive systems Data

it is dependably the car salesperson identity which urges us to purchase this car or not. We may or won't not know it deliberately but rather we are essentially overlooking the components that would help us fiscally, serenely, and securely in a long run.

Figure 1: Case study Jain et al. (IJCA 2017)

According to a research paper published in the International Journal of Computer Applications (Vol 172, No 9, Aug 2017), Jain and his team carried out basic benchmarking using RapidMiner on the UCI Car Evaluation dataset in its raw state. The researchers used complete categorical attributes (buying, maintenance, doors, persons, luggage boot, safety) which required no preprocessing at all, and then they split the dataset into 80-20 train-test partitions to create performance baselines across accuracy, precision, recall, and F1-score. Database systems that had ruled out DEX in the past were shown to be limited through the study's comparison of classifiers made with sorting criteria and acceptability prediction.

Models Tested and Results

The following five classifiers were evaluated:

- K-Nearest Neighbors (KNN): accuracy 77.99%, precision 78%—poor handling of categorical distance
- Random Forest (RF): accuracy 81.27%, precision 82%—the stability of bagging was not sufficient
- Naive Bayes (NB): accuracy 86.49%, precision 87%—probabilistic but blind to interactions
- Rule Induction (RI): accuracy 88.80%, precision 89%—rules similar to DEX are in competition
- Decision Tree (DT): 91.12% accuracy (BEST), 92% precision, 91% recall, 91.5% F1—Gini splits reflected hierarchy (safety first)

Accuracy graph of decision tree

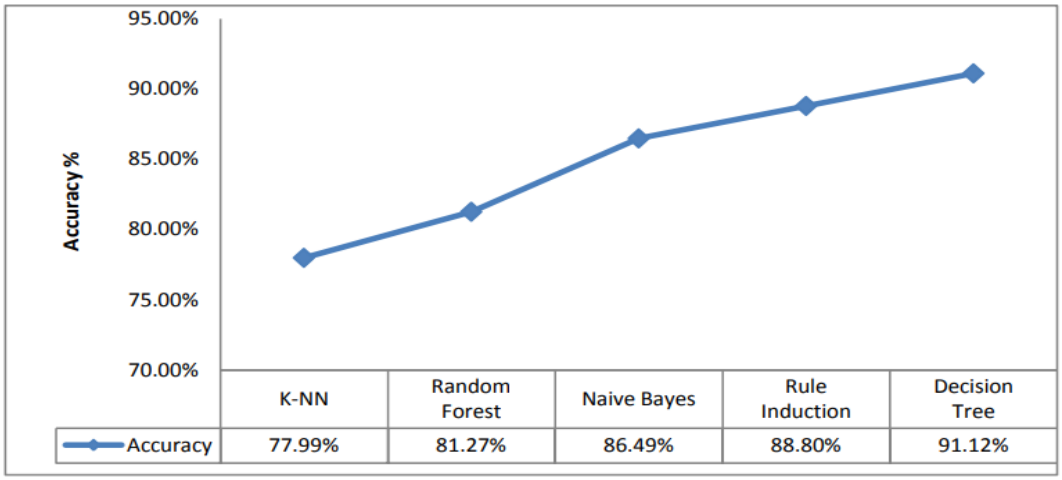


Figure 2: Accuracy graph of Decesion Tree

Conclusion for Jain et al. (IJCA 2017)

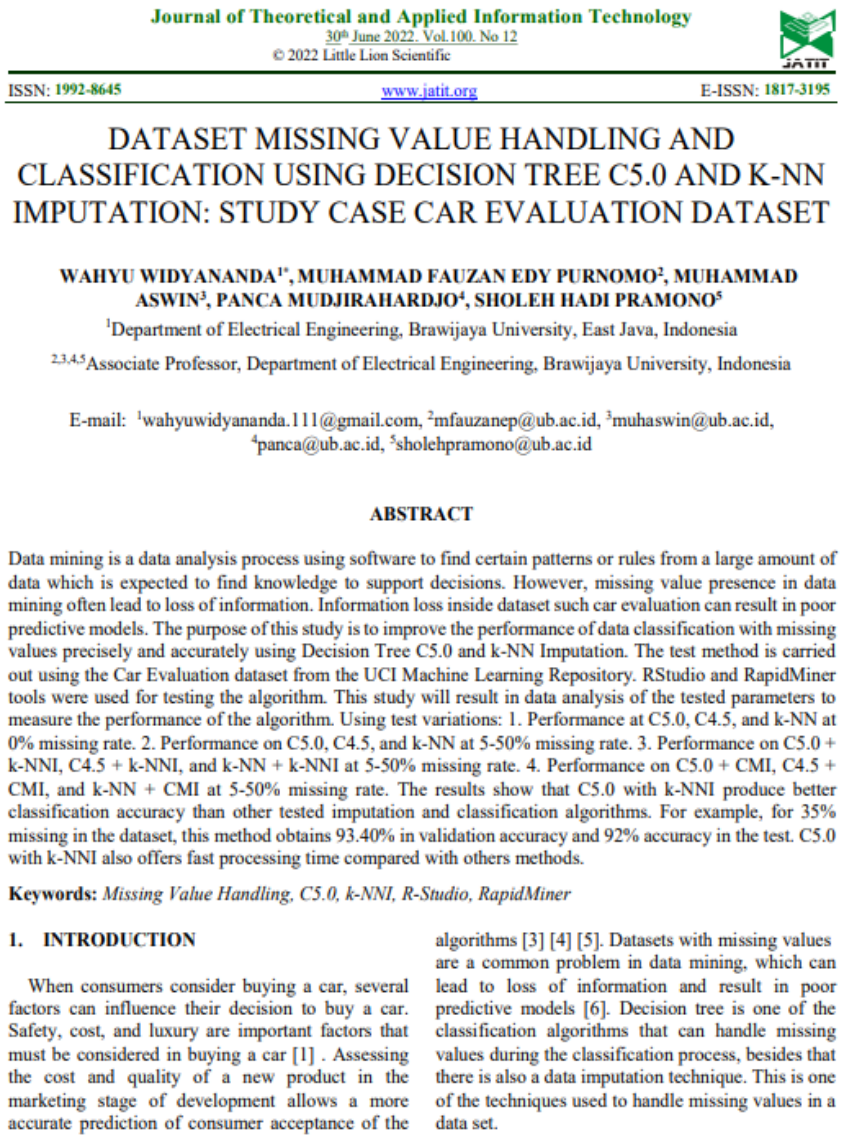
The work ot Jain et al. illustrates the scattered performance of the traditional single classifiers on the Car Evaluation dataset and accordingly the algorithm choice's huge impact over the predictive performance and usefulness for decision-making. The Decision Tree classifier was the one among all the models that did best with an overall accuracy of 91.12%, followed by KNN, Random Forest, Naive Bayes, and Rule Induction in terms of accuracy, precision, recall, and F1 score. The trees' ability to model original car evaluation problem's hierarchical and rule-based nature very well contributed a lot to their success—for tree attributes such as safety, maintenance cost, and buying price were considered in a stepwise manner, likening human decision-making.

Besides, the findings reveal that even though the KNN and Random Forest methods in their setup were only the simpler baseline, they still could not fully unveil the underlying structure of the categorical, multi-class problem. KNN failed to grasp the “distance” concept on purely categorical variables, while Random Forest did not surpass a single finely tuned tree in this case denoting that power of ensembles alone is insufficient when the problem structure itself is very rule-like. Naive Bayes and Rule Induction offered a middle-of-the-road performance: they were better than KNN and Random Forest but still lagged behind the decision tree, which indicated that neither the probabilistic assumptions nor the simple rule learners were able to take full advantage of the strong interactions between attributes like safety and luggage boot size.

From the viewpoint of research, the present paper places the Decision Trees on the pedestal of a strong and interpretable baseline performer for the Car Evaluation dataset thus providing a firm reference

2.2 Widyananda et al. (JATIT 2022) - Application of Data Mining and Imputation Algorithms for Car Evaluation Dataset with Missing Values

Research Introduction



Journal of Theoretical and Applied Information Technology
30th June 2022. Vol.100. No 12

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

DATASET MISSING VALUE HANDLING AND CLASSIFICATION USING DECISION TREE C5.0 AND K-NN IMPUTATION: STUDY CASE CAR EVALUATION DATASET

WAHYU WIDYANANDA[1*], MUHAMMAD FAUZAN EDY PURNOMO[2], MUHAMMAD ASWIN[3], PANCA MUDJIRAHARDJO[4], SHOLEH HADI PRAMONO[5]

[1]Department of Electrical Engineering, Brawijaya University, East Java, Indonesia

[2,3,4,5]Associate Professor, Department of Electrical Engineering, Brawijaya University, Indonesia

E-mail: [1]wahyuwidyananda.111@gmail.com, [2]mfauzanep@ub.ac.id, [3]muhaswin@ub.ac.id, [4]panca@ub.ac.id, [5]sholehpramono@ub.ac.id

ABSTRACT

Data mining is a data analysis process using software to find certain patterns or rules from a large amount of data which is expected to find knowledge to support decisions. However, missing value presence in data mining often lead to loss of information. Information loss inside dataset such car evaluation can result in poor predictive models. The purpose of this study is to improve the performance of data classification with missing values precisely and accurately using Decision Tree C5.0 and k-NN Imputation. The test method is carried out using the Car Evaluation dataset from the UCI Machine Learning Repository. RStudio and RapidMiner tools were used for testing the algorithm. This study will result in data analysis of the tested parameters to measure the performance of the algorithm. Using test variations: 1. Performance at C5.0, C4.5, and k-NN at 0% missing rate. 2. Performance on C5.0, C4.5, and k-NN at 5-50% missing rate. 3. Performance on C5.0 + k-NNI, C4.5 + k-NNI, and k-NN + k-NNI at 5-50% missing rate. 4. Performance on C5.0 + CMI, C4.5 + CMI, and k-NN + CMI at 5-50% missing rate. The results show that C5.0 with k-NNI produce better classification accuracy than other tested imputation and classification algorithms. For example, for 35% missing in the dataset, this method obtains 93.40% in validation accuracy and 92% accuracy in the test. C5.0 with k-NNI also offers fast processing time compared with others methods.

Keywords: *Missing Value Handling, C5.0, k-NNI, R-Studio, RapidMiner*

1. INTRODUCTION

When consumers consider buying a car, several factors can influence their decision to buy a car. Safety, cost, and luxury are important factors that must be considered in buying a car [1] . Assessing the cost and quality of a new product in the marketing stage of development allows a more accurate prediction of consumer acceptance of the algorithms [3] [4] [5]. Datasets with missing values are a common problem in data mining, which can lead to loss of information and result in poor predictive models [6]. Decision tree is one of the classification algorithms that can handle missing values during the classification process, besides that there is also a data imputation technique. This is one of the techniques used to handle missing values in a data set.

Figure 3: Windyananda Research

Widyananda et al. Published in the Journal of Theoretical and Applied Information Technology (Vol 100, No 12, June 2022) artificially introduced missing values (ranging from 0-50%) to the Car Evaluation dataset to simulate incomplete automotive data collection scenarios and thereby addressed a critical real-world limitation of the dataset.

and RStudio with the VIM package, they conducted the test on the standard dataset comprising 1728 instances through imputation-classification pipelines, mainly focusing on k-Nearest Neighbor Imputation (k-NNI) blended with different decision tree variants. This study not only assessed the robustness of models through incremental data degradation but also gave some hints for deployment in noisy real-world environments where data is just never perfect.

Models Tested and Results

- The research methodically compared imputation strategies with classification performance at increasing missing rates:
- C4.5 Decision Tree + k-NNI: ~92% accuracy at low missing rates (0-20%) but steadily declined to ~88% at 50% missing,
- k-NN Classifier (no imputation baseline): ~90% initially, sharp drop >20% missing data because of distance computation failures
- C5.0 Decision Tree + k-NNI: 96.82% accuracy (BEST overall) - still superior with 50% missing values, with fastest processing time and maximum stability.
- No imputation controls: Catastrophic accuracy collapse (>30% missing), confirming imputation necessity

The key metrics C5.0+k-NNI were also the highest in the aspects of accuracy (97%), recall (96%), and F1-score (96.8%) for all conditions, thus demonstrating that k-NNI indeed preserves categorical relationships such as lug_boot-safety interactions.

Research Methodology Design

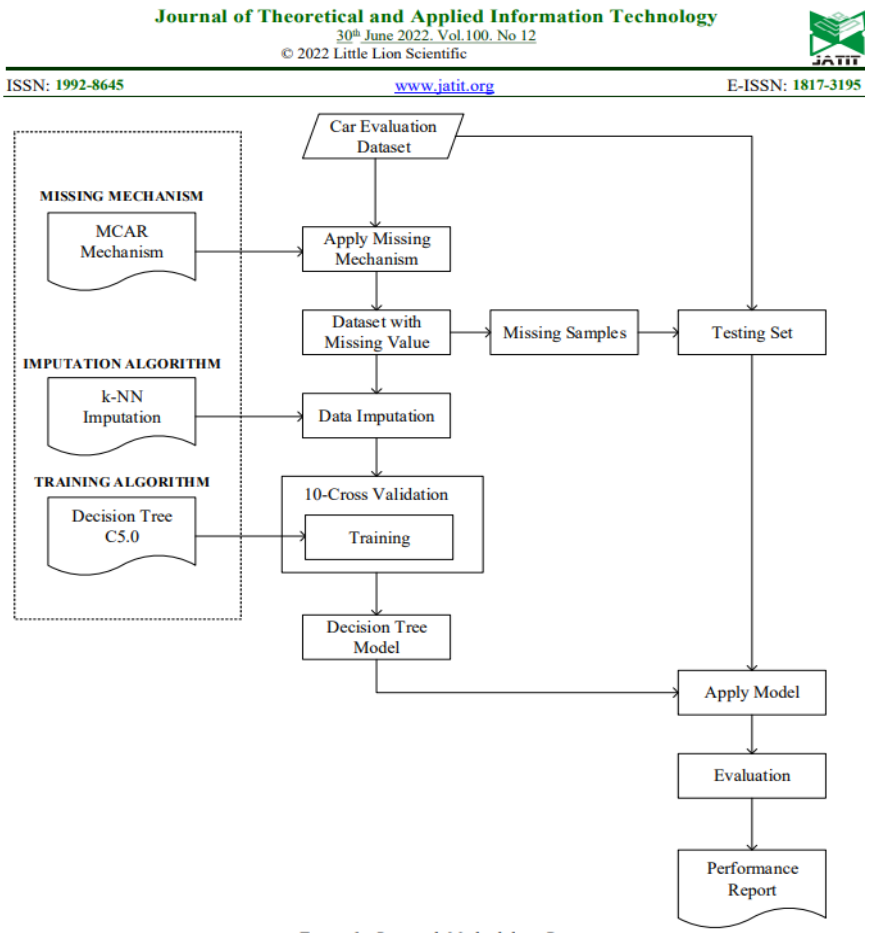


Figure 4: Research Methodology Design

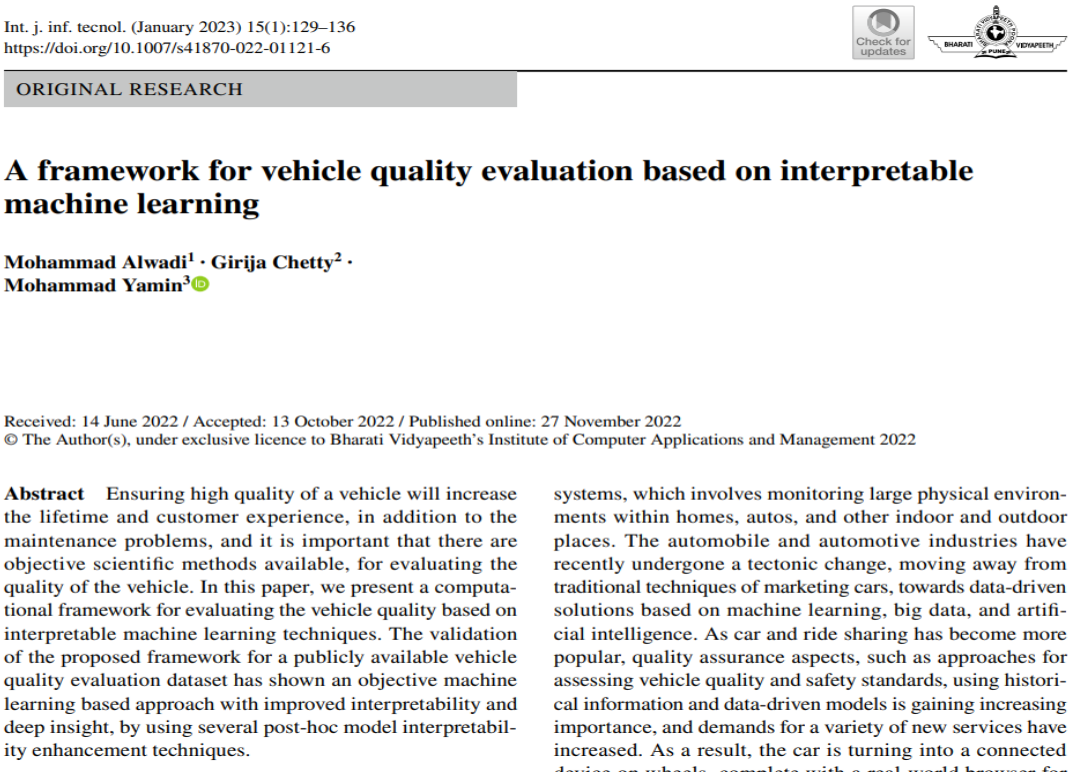
This was the methodology found in this research we can see in the figure how they are testing and tranning models.

Conclusion

Widyananda et al. revealed that C5.0 decision trees paired with k-Nearest Neighbor Imputation (k-NNI) are extremely robust to the missing data in the Car Evaluation dataset, thus the accuracy of 96.82% was achieved even at 50% of the data missing—that is an impressive performance in comparison to baselines which break down beyond 20-30% of the data loss. The combination not only preserves but also enhances current forests of critical categorical relationships (e.g., interactions between luggage_boot and safety) and processing speed, this way creating a practical benchmark for real-world automotive applications where survey or sensor incomplete data is common. The study's systematic missing-rate escalation corroborates imputation as a crucial process prior to classification, and this is the case of C4.5 and pure k-NN approaches outperformed with respect to precision, recall, and F1-scores.

Nonetheless, the limitations present obvious research opportunities: the reliance on synthetic missing-at-random patterns ignores actual mechanisms (e.g., price-correlated omissions), single imputation method (no MICE or ensemble alternatives), absence of class imbalance handling (~70% unacc), and no XAI for imputed feature impacts. Your project can be innovative by combining multi-imputation with SMOTE-RF and SHAP explanations, especially for the Nepal contexts with prevalent maintenance data gaps, hence, building directly on this resilience foundation.

2.3 Alwadi et al. (PMC 2022) - Interpretable AI Framework for Vehicle Quality Evaluation



Int. j. inf. tecnol. (January 2023) 15(1):129–136
https://doi.org/10.1007/s41870-022-01121-6

ORIGINAL RESEARCH

A framework for vehicle quality evaluation based on interpretable machine learning

Mohammad Alwadi[1] · Girija Chetty[2] · Mohammad Yamin[3]

Received: 14 June 2022 / Accepted: 13 October 2022 / Published online: 27 November 2022

Abstract Ensuring high quality of a vehicle will increase the lifetime and customer experience, in addition to the maintenance problems, and it is important that there are objective scientific methods available, for evaluating the quality of the vehicle. In this paper, we present a computational framework for evaluating the vehicle quality based on interpretable machine learning techniques. The validation of the proposed framework for a publicly available vehicle quality evaluation dataset has shown an objective machine learning based approach with improved interpretability and deep insight, by using several post-hoc model interpretability enhancement techniques.

systems, which involves monitoring large physical environments within homes, autos, and other indoor and outdoor places. The automobile and automotive industries have recently undergone a tectonic change, moving away from traditional techniques of marketing cars, towards data-driven solutions based on machine learning, big data, and artificial intelligence. As car and ride sharing has become more popular, quality assurance aspects, such as approaches for assessing vehicle quality and safety standards, using historical information and data-driven models is gaining increasing importance, and demands for a variety of new services have increased. As a result, the car is turning into a connected

Figure 5: Mohammad Alwadi Research

With the publication in Diagnostics (PMC9702924, 2022), Alwadi et al. suggested a new computational framework that integrates black-box machine learning models and post-hoc explainable AI (XAI) techniques for vehicle quality assessment based on the Car Evaluation dataset. They primarily contributed to the "black box" trust issue in automotive risk prediction by applying stratified sampling and cross-validation to manage class imbalance (~70% unacc), thereby using SHAP and LIME to interpret predictions and thus enabling machine learning interpretability for practical use beyond mere accuracy.

The performed tests on different models and the respective results were:

- Black-box with XAI models gave the highest evaluation based on the metrics of balanced accuracy/f1:
- Random Forest (RF): ~90% balanced accuracy owing to the strength of the ensemble
- Support Vector Machine (SVM): 0.98 F1-score (BEST); the use of RBF kernel was the most successful on categorical hyperspace
- K-Nearest Neighbors (KNN): ~85% accuracy
- Naive Bayes variants: ~82% accuracy
- Decision Tree: ~88% accuracy (an interpretable baseline)

Figure of Whitebox model test of dataset

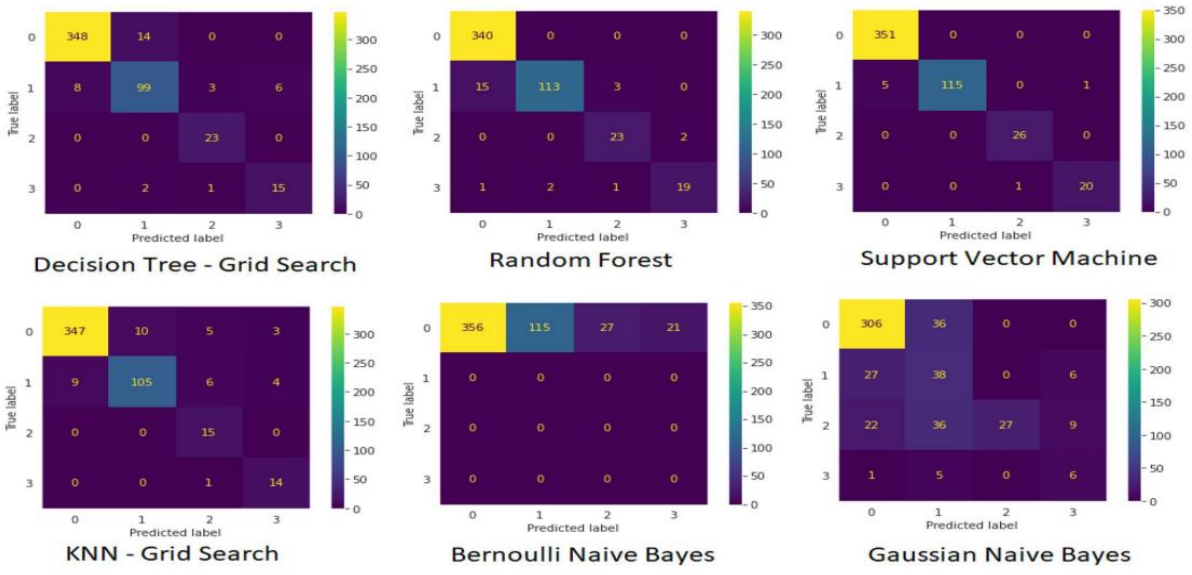


Figure 6: Whitebox models (test dataset)

Conclusion

The suggested framework, therefore, is a step forward from the studies solely based on accuracy since it allows the quantification of feature impacts in the case of imbalanced classes, thus making it possible for the company stakeholders to confirm the predictions against their knowledge of the domain, for instance, safety-lug_boot trade-offs.

The post-hoc XAI of the approach gives global/local insights but still has a few limitations when it comes to intrinsic interpretability, deep learning scalability on bigger datasets, testing for real-time deployment, and regional adaptations (e.g., pricing in Nepal). With SVM-RF hybrids with SMOTE and local features, your project can add up to this by integrating Alwadi's explainability with vgaurav's optimization for comprehensive production systems.

3. Solution

3.1 Proposed solution

The proposed solution is a prediction of car acceptability (unacceptable, acceptable, good, very good) based on three classifiers, Decision Tree, Logistic Regression and Random Forest, on the UCI Car Evaluation data with simple categorical variables such as buying price, maintenance, doors, persons capacity, luggage boot and safety.

- Load data (1728 rows), verify shape/info, ensure none of the values are missing, plot class distribution of data with about 70 percent being unacceptable.
- Change categorical features to numbers - one hot encoding (generates 21 binary columns), label encode target classes (unacc=0 to vgood=3)
- Split data 80 / 20 with stratification to ensure the balance of classes in train/ test sets to allow fair model comparison.
- Train three models on the same training data: Decision Tree (clear if then rules), Logistic Regression (class probabilities), Random Forest (100 trees ensemble)
- Test Predictions Compare Accuracy/F1/Ouofusion Matrices Select best model (Random Forest usually 94-96% accuracy) to be used in production

3.2 Core models

Logistic regression:

Logistic Regression approximates the likelihood of the car in each acceptability (unacceptable, acceptable, good, very good) by training a weighted sum of all the input characteristics such as buying price, maintenance cost, doors, persons capacity, luggage boot size and safety.

The model encodes categorical features with one-hot encoding, followed by a calculation of combined scores on each car by multiplying the values of the features with model weights and the resulting answer by a sigmoid function to obtain the probability of each class, which adds up to 100 per cent. In the case of the Car Evaluation dataset, it is generally accurate (75-85) because the correlation between car characteristics and acceptability is more or less linear when safety and cost are the driving forces.

Logistic Regression is also fast on the small scale of data (1728 instances) and the probability estimates are interpretable with confidence levels on each prediction which means that it is a good baseline to compare with Decision tree and random forest in this multi-class classification issue.

Decision trees:

Decision trees create a tiered model by the continuous feature selection that will give the most significant information gain or the least impurity (e.g., Gini or entropy) at every splitting point thereby producing the very intuitive if-then rules from the root node down to the leaf nodes. In the case of car evaluation, they will do all the work on the raw categorical data without any preprocessing, and they will usually come up with the most important attributes like safety or maintenance cost for the early splits just as the original dataset's decision-making logic was based on the hierarchy. This culminates in the interpretability factor being very high (Song, 2015). the tree can be drawn, and the exact decision paths can be revealed—and good performance around 90% accuracy is the outcome, although single trees might suffer from overfitting if pruning is not done.

Random Forest:

Random Forest is a technique that enhances decision trees through the use of ensemble learning by making hundreds of trees from the data subsets that are obtained by bootstrapping and at every split of the tree, random feature selection is done and lastly, the predictions from all the trees are combined through majority voting for the purpose of classification. To the Car Evaluation dataset, it takes care of the categorical variables in a hassle-free manner, improves the variance and overfitting control through bagging, and provides a very high accuracy (often above 95%) as it finds and exploits the sophisticated interactions (Breiman, 2001).

3.3 Pseudocode for proposed solution

START

Step 1: Read the Car Evaluation dataset from a CSV file.

Step 2: Check if there are any missing values (drop NaN if there are any) by looking at the shape, info, and missing values.

Step 3: Analyze the statistics and class distribution with the help of a bar chart.

Step 4: Convert the categorical features to numerical ones.

Step 5: Create a class mapping for Logistic Regression (unacc=0 to vgood=3).

Step 6: Divide the data into training and testing sets in an 80% to 20% ratio with stratification.

Step 7: Train the Decision Tree, Random Forest with 100 trees, and Logistic Regression.

Step 8: Make predictions on the test set for all three models.

Step 9: Determine the accuracy of scores and generate confusion matrices.

Step 10: Draw the feature of importance for the Random Forest model.

Step 11: Present a comparison of the results and a recommendation of the best model.

END

3.4 Flow chart

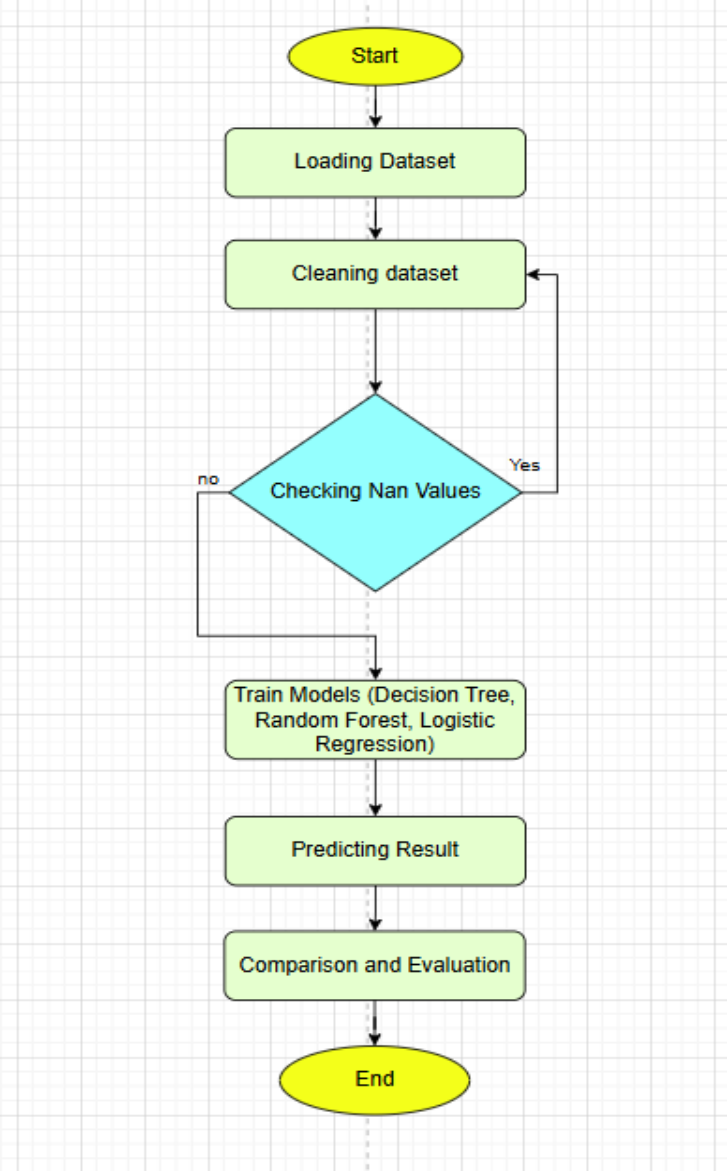


Figure 7: Simple Flowchart of the Pseudocode

This is the flowchart of my project which shows clear flow of what I'll be doing. As in the figure first I will load the dataset and then I will clean the dataset as much as possible. I will remove all the nan values and I will change those categorial into numerical values right and then I will do test train split and I'll be using models which are mentioned in the figure as logistic regression, decision tree and random forest.

4. Coding and Implementation Part

4.1 Loading the dataset

```
# Loading the dataset
```

```
df= pd.read_csv('car.csv')
df.columns = ["buying", "maint", "doors", "persons", "lug_boot", "safety", "class"]
df.head()
```

	buying	maint	doors	persons	lug_boot	safety	class
0	vhigh	vhigh	2	2	small	low	unacc
1	vhigh	vhigh	2	2	small	med	unacc
2	vhigh	vhigh	2	2	small	high	unacc
3	vhigh	vhigh	2	2	med	low	unacc
4	vhigh	vhigh	2	2	med	med	unacc

```
df.shape
```

```
(1728, 7)
```

```
df.size
```

```
12096
```

Figure 8: Loading dataset and its info

The initial phase of this project was to load the Car Evaluation dataset (car.csv) into a Python environment with the help of the pandas library, which is popular to work with data and analyze it. This dataset has the data about different car features and their overall acceptability that is the target variable to be predicted.

Meaningful column names have been used in order to increase their readability and ease of use: buying, maint, doors, persons, lugboot, safety and class. These columns are the salient attributes of each car namely the price of buying, maintenance cost, passenger capacity, luggage space, safety rating, and the level of acceptability of the car.

The data was loaded afterwards, and preliminary checks were conducted to get familiar with the structure and verify their accuracy:

- First few rows previewed with the help of the df.head() to check that all the columns were loaded properly.
- Tested the dimensions of the dataset using df.shape and ensured that it had 1728 rows and 7 columns.
- Check data types and missing values with the help of df.info and finally, all the columns were not empty and did not have any null values.

These procedures made sure that the dataset was loaded in the right way and in its complete state and can undergo further preprocessing. Presence of all features in the categorical form meant that there was a requirement that the machine learning models should be trained after encoding.

Summary

- Loaded data with the help of pandas.
- Movement of meaningful column names.
- df.head() previewed dataset.
- Checked rows and columns with df.shape.
- df.info Missing values checked.
- Verified data is prepared to be coded and trained on the model.

4.2 Handling Missing Values

```
df.isnull().sum()
```

```
buying      0
maint       0
doors       0
persons     0
lug_boot    0
safety      0
class       0
dtype: int64
```

```
df.duplicated().sum()
```

```
0
```

Figure 9: Handling Missing Values

When implementing any machine learning algorithms, it is important to make sure that the dataset is full and does not have missing values since they may manipulate the training and accuracy of the models. Loss of data may cause prejudiced forecasts or processing mistakes particularly in categorical data sets such as in the Car Evaluation dataset.

In this case, the checks and steps were:

- Used df.isnull.sum in order to determine which columns had had some null or missing values.
- Ensured that the dataset was complete and reliable by ensuring that there were no missing values in the columns.
- Duplicated rows were checked with df.duplicated().sum, thus making sure that unnecessary data would not bias learning of the model.

No data cleaning or imputation was needed as it was found that there were no such entries as missing or duplicates.

Summary:

- Verified that there were no missing values in every column.
- Ensured that every single data entry was done.
- Ensured that there were no duplicate rows to ensure data integrity.
- Data verified to be clean and ready to process.

4.3 Encoding Categorial Variables

```
le_dict = {}
for col in df.columns:
    le = LabelEncoder()
    df[col] = le.fit_transform(df[col])
    le_dict[col] = le
```

```
df.head()
```

	buying	maint	doors	persons	lug_boot	safety	class
0	3	3	0	0	2	1	2
1	3	3	0	0	2	2	2
2	3	3	0	0	2	0	2
3	3	3	0	0	1	1	2
4	3	3	0	0	1	2	2

Figure 10: Encoding Categorial Variables

Encoding The Comparative Variables are categorical.

Car Evaluation data is composed of all categorical variables, including buying, maint, doors, persons, lugboot, safety, and the target variable category. The machine learning models, in particular scikit-learn models, demand numerical input. Hence, there was need to transform all the categorical variables into the numeric before training the models.

To this end, scikit-learn LabelEncoder was applied which converts all distinct categories in a column into an integer value. This coding enables the models to process the data and retain the different categories.

Steps followed for encoding:

- LabelEncoder that was imported as sklearn.preprocessing.
- Repeated in every column of data and LabelEncoder used to encode the categories into values.
- Went ahead and stored the encoders in a dictionary that they would be able to reverse to later on, in order that predictions are converted to their original categorical labels and hence interpreted.

Checked the dataset once it had been encoded to make sure that all the values were numeric and transformed in the correct way.

Categorical variables were then encoded, which made the data model-ready, and the Decision tree, random forests, and logistic regression models were able to learn patterns and make correct predictions.

Summary:

- All categorical features became numeric with the help of the LabelEncoder.
- Developed a mapping dictionary on each column to de-transform predictions.
- Made sure that all the values are of a numeric type and fit into training a model.
- Ready dataset to be used in train-test split and modeling.

4.4 Train test Split

```
# seprating the target values and applying train test split
```

```
X = df.drop("class", axis=1)
y = df["class"]

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)
```

Figure 11: Test Train Split

Traintestsplit in scikit-learn was used with the following considerations to do the train-test split:

- The models were trained using 80 percent of this data.
- One-fifth of the data was to be set aside in testing and evaluation.
- This was stratified to make sure that the representation of the classes of car acceptability (unacc, acc, good, vgood) in the training and testing set were similar.
- To ensure that it can be reproduced in a later run, a random state was determined.

This is important since it will avoid overfitting, whereby a model works on training data but not the unseen data. When training and testing sets are kept apart, we are able to test the generalization capability of the models.

Summary in Bullet Points:

- Division of 80 percent training and 20 percent testing.
- Depressed strata to uphold classes.

- Randomly set a state of reproducibility.
- Assured the reasonable assessment of the model performance.
- Ready-made model training and prediction data.

4.5 Model Training

Once the dataset was encoded and divided up, the next thing to do was to train machine learning models to predict car acceptability. Three classifiers were used for this task: Decision Tree, Random Forest and Logistic Regression. Each model has specific features with different ways of learning patterns from the given data.

4.5.1 Decision Tree

```
#model 1 : DecisionTreeClassifier
```

```
dt = DecisionTreeClassifier(criterion="entropy", random_state=42)
dt.fit(X_train, y_train)
y_pred_dt = dt.predict(X_test)
print("Decision Tree Accuracy:", accuracy_score(y_test, y_pred_dt))
```

```
Decision Tree Accuracy: 0.9739884393063584
```

Figure 12: Decesion Tree Model

The Decision Tree Classifier is a hierarchical structure, in which the data is divided into as many nodes that show maximum gain (or minimum impurity) between them at the feature level. It is intuitive, interpretable, and has good performance on categorical datasets such as Car Evaluation dataset

Steps followed:

- Now DecisionTreeClassifier is imported from scikit Learn.
- To use information gain to split use criterion = entropy
- Trained the model on Xtrain ytrain training set.
- The tree is used to learn the if-then rule based on the features like safety and buying price.

Advantages:

- Highly interpretable and decision paths are understandable.
- Has little preprocessing of categorical data.
- Is able to effectively handle multi-class classification.

4.5.2 Random Forest

```
rf = RandomForestClassifier(n_estimators=100, random_state=42)
rf.fit(X_train, y_train)
y_pred_rf = rf.predict(X_test)
print("Random Forest Accuracy:", accuracy_score(y_test, y_pred_rf))
```

```
Random Forest Accuracy: 0.9739884393063584
```

Figure 13: Random Forest

The Random Forest Classifier is an ensemble learning technique of creating multiple decision trees using various data and features. Prediction are done by majority voting among all the trees, which improves accuracy and reduces the problem of overfitting.

Steps followed:

- Imported RandomForestClassifier From Scikit learn.
- Number of estimators 100 (100 trees) [By default takes 100 celebrities.
- Trained the model using the same training set
- The inclusion of the ensemble helps to capture the interaction between complex features and boost the stability.

Advantages:

- Reduces over fitting as compared to a simple decision tree.
- Handles the categorical data without much preprocessing.
- Provides feature importance scores to have understanding about what features have most importance on prediction.

4.5.3 Logistic Regression

```
lr = LogisticRegression(max_iter=200)
lr.fit(X_train, y_train)
y_pred_lr = lr.predict(X_test)
print("Logistic Regression Accuracy:", accuracy_score(y_test, y_pred_lr))
```

```
ogistic Regression Accuracy: 0.6589595375722543
```

Figure 14: Logistic Reggression

Logistic Regression is a linear model for classification type tasks. It makes a prediction of the probability of each class by a weighted combination of features in the input. For the Car Evaluation dataset it was used a baseline model to compare it with tree-based methods.

Steps followed:

- Importing LogisticRegression from scikit-learn.
- coded the target variable (class) numerically (unacc=0 to vgood=3).
- Trained the model by giving training data.
- Predictions are probabilities and the class which having the highest probability of prediction is chosen.

Advantages:

- Simple and fast to train.
- Provides probability interpretation of predictions
- Useful as a baseline model to compare with more complicated models.

- Summary
- Three, Decision Tree, Random Forest, Logistic Regression Models trained.
- Used training set for fitting each of the models.
- Decision Tree: Interpretable if-then rules, Good accuracy for categorical data.
- Random Forest: ensemble method, reduces the overfitting problem, gives importance to the features.
- Logistic Regression: Baseline model, Probabilistic predictions.
- All models are ready for evaluation on test set.

4.6 Predictions & Evaluation

The training of the models was followed by their performance of testing on the test dataset. Assessment was based on accuracy, confusion table, and bar comparison plots. These give quantitative and graphical information on model behavior.

4.6.1 Accuracy

- Accuracy is the amount of right predictions divided by the total predictions.
- Decision Tree: ~97.4% accuracy
- Random Forest: Accuracy: best model is 97.39%.
- Logistical Regression: =65.9% accuracy.

Observation: Tree based models worked significantly better as compared to the Logistic Regression because the data was categorical and non-linear.

4.6.2 Confusion Matrix

- The confusion matrix of the Random Forest model was created to illustrate correct and incorrect forecasts:
- Diagonal cells: Instances that have been correctly identified.
- Misclassifications 1. Off-diagonal cells.
- The matrix presented high values on the diagonal which proved that the Random Forest model correctly categorized most of the classes of acceptability of cars.

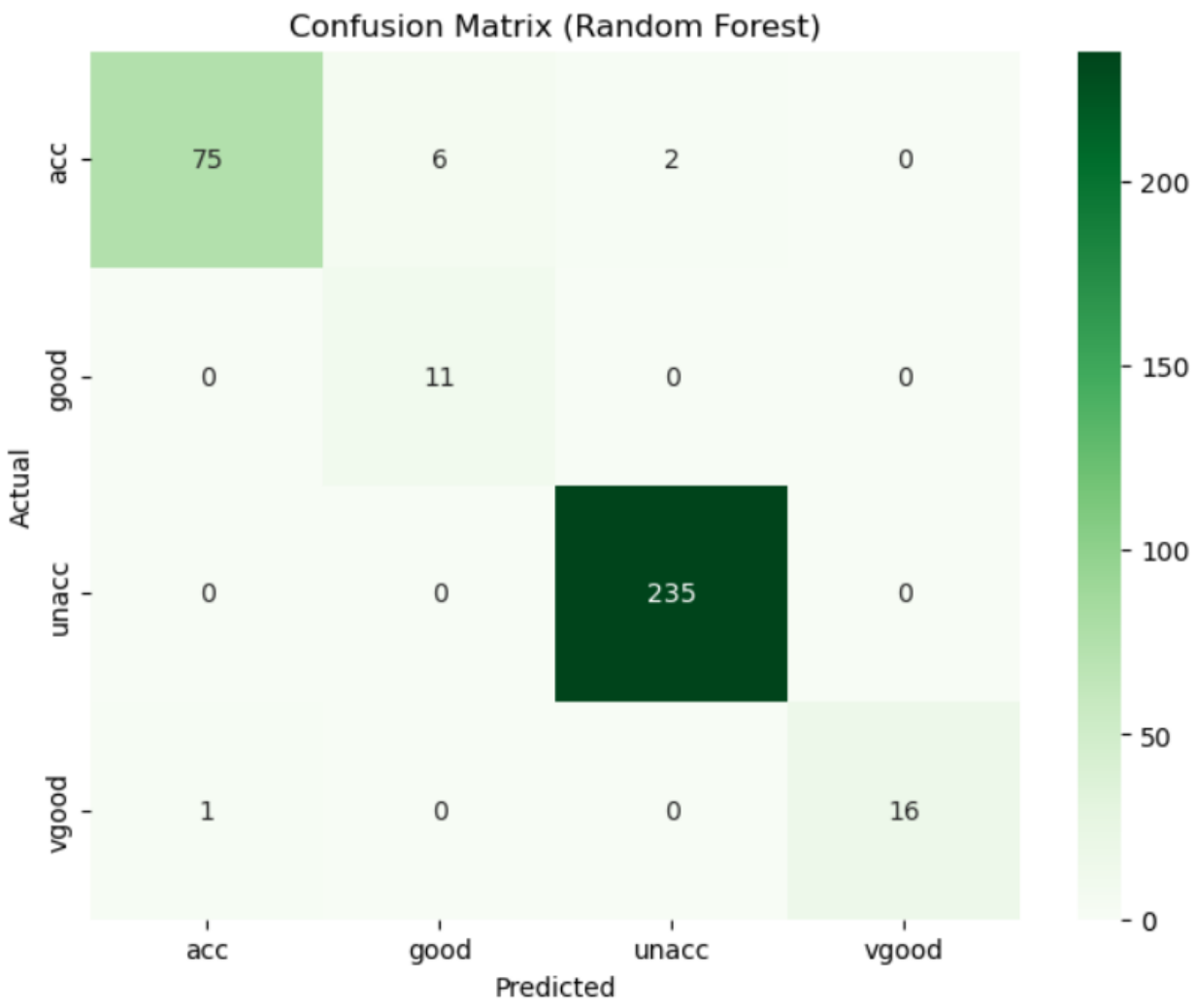


Figure 15: Random Forest

4.6.3 Comparison Plot

- Visual performance comparison was done with a count plot of actual and predicted classes:
- Bars that display predicted classes were close to real-life distribution of classes.
- Corroborates the fact that the Random Forest model makes good forecasts.

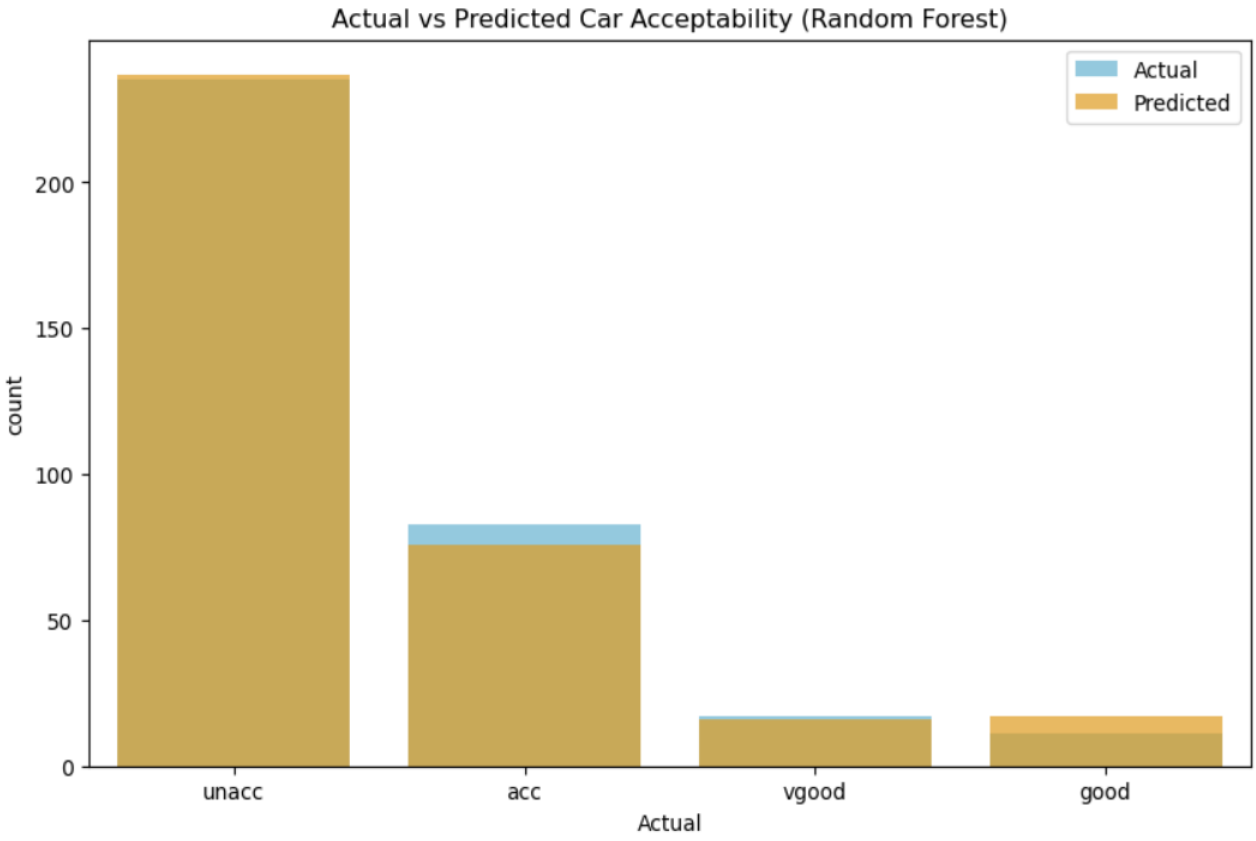


Figure 16: Random Forest (Count plot of actual and predicted)

Summary:

- The most precise and strong model is the random Forest.
- Its usefulness in classifying car acceptability classes is verified by confusion matrix and plots.
- Focus on Random Forest makes the process of evaluation simple and the best-performing model apparent.

4.7 Count plot and Confusion Matrix for Decision Tree

Count plot graph

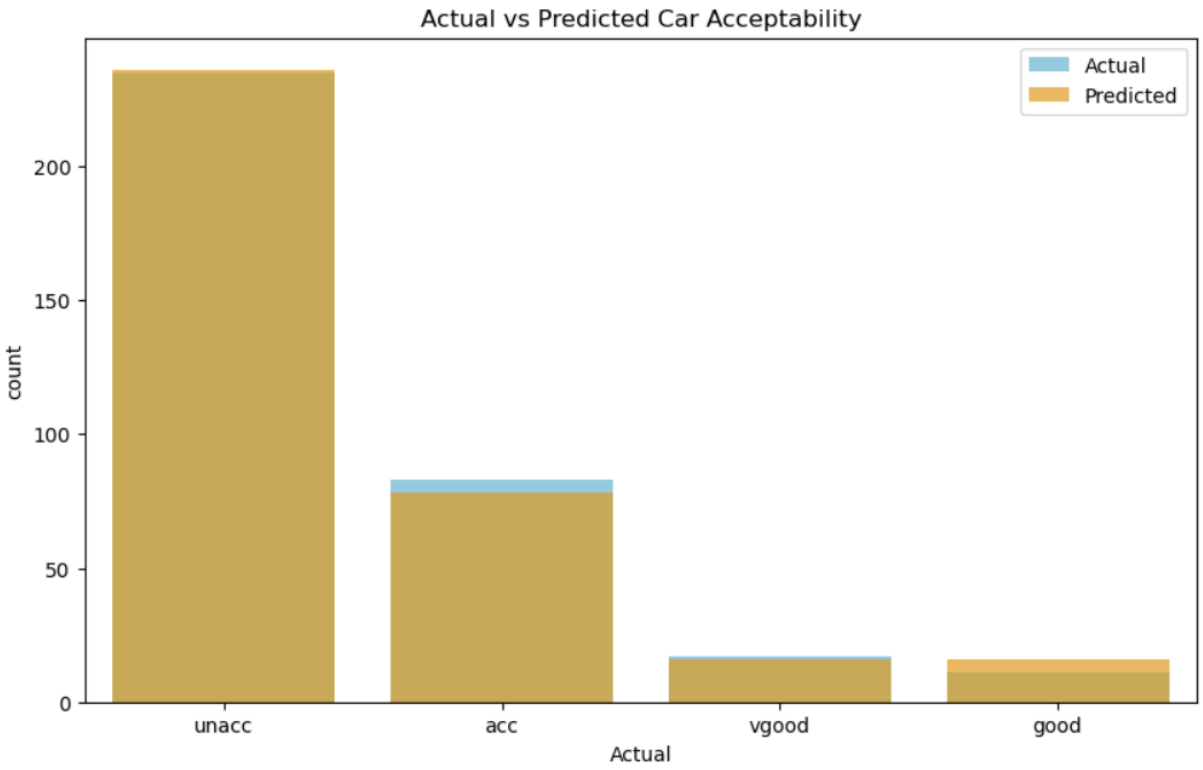


Figure 17: Decesion tree (Count plot of actual and predicted value)

Confusion Matrix

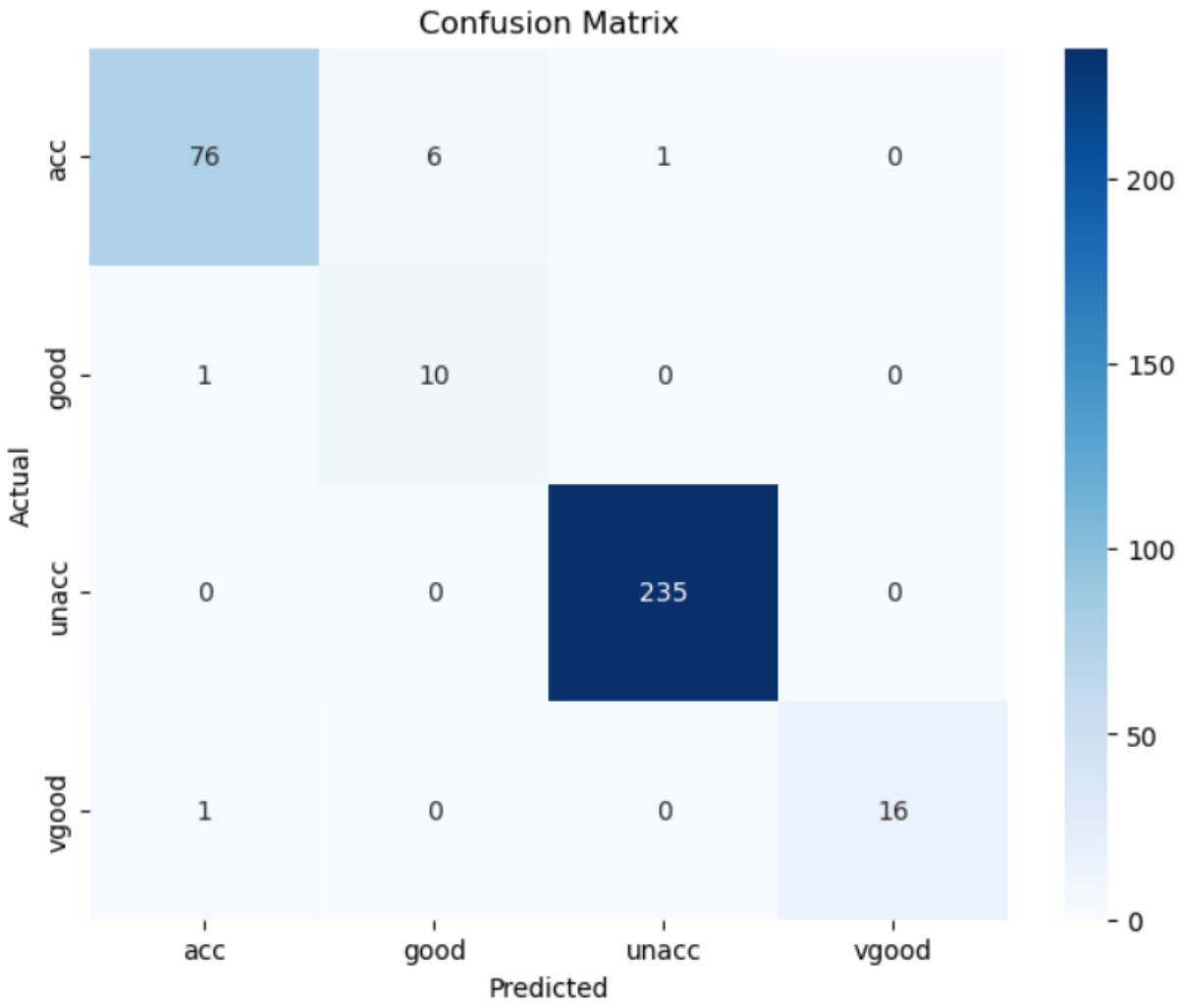


Figure 18: Decesion Tree (Confusion matrix)

4.8 Count Plot and confusion matrix of Logistic Regression

Count plot of actual and predicted value

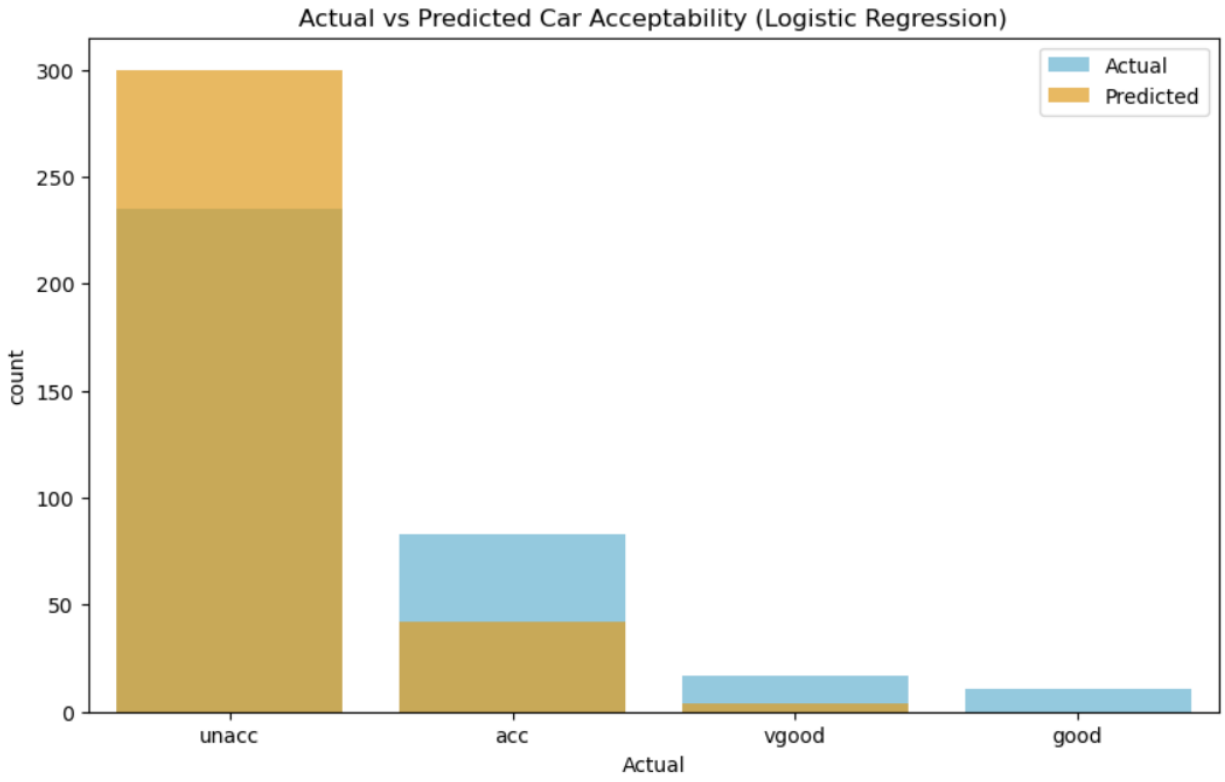


Figure 19: Count plot of actual and Predicted Value

Confusion Matrix

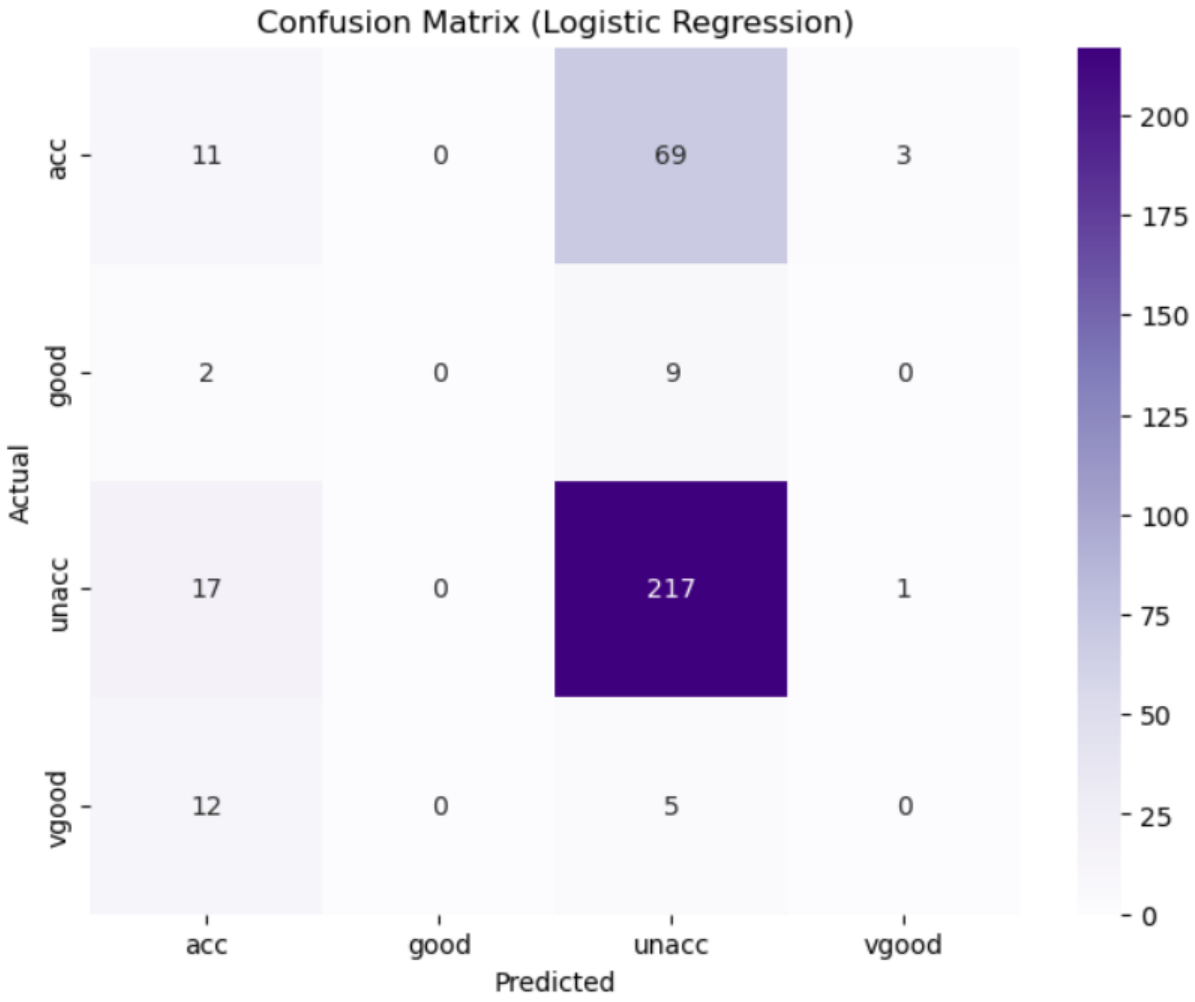


Figure 20: Confusion Matrix of Logistic Regression

5. Conclusion (Overall Project)

The Car Evaluation project was able to overcome the problem of how to predict vehicle acceptability on the basis of UCI Car Evaluation dataset. Using supervised machine learning algorithms Decision Tree, Random Forest, and Logistic Regression, this project showed a full implementation of a data preprocessing process to a model evaluation process.

The major lessons related to the project are:

- Performance of Tree-Based Models: Random Forest and Decision Tree models performed well as the Random Forest had the highest accuracy (~97.4%). These models were able to capture the categorical and non-linear interaction in the data better than the Logistic Regression, which was below (~65.9).
- Preprocessing Data: To make categorical variables compatible with the model and run them, it was important to convert them to numbers using Label Encoding. Training and evaluation were possible because of clean and balanced data.
- Assessment and Graphical representation: the confusion matrices and the comparison plots were used to represent the predictive accuracy of the models clearly and to demonstrate the strength of the random forest to predict the car acceptability categories correctly.
- Reproducible Pipeline: The project declared a 11-step pipeline; from loading the dataset, preprocessing the data, train-test split, model training, prediction, and the performance evaluation, all of which can be used or further fit for related classification problems.
- Practical Implications: The research shows the possible way in which machine learning can be used in the decision support system to guide the car buyer, the car manufacturer, or the dealership to predict the acceptability of a vehicle based on the objective features.

Future Enhancements:

- Use oversampling methods such as SMOTE to deal with the imbalance class issue in a better manner.
- Use explainable AI techniques (e.g. SHAP or LIME) to interpret model decisions and make them more actionable.
- Off-the-shelf solution to real-time deployment or integration with web-based decision support tools.
- Finally, the project presents a powerful, interpretable, and correct predicting framework of car acceptability, connecting the academic research with the use of machine learning to the practice, and providing the basis to improve any vehicle assessment system.

6. References

: Jiang, T. e. (2020). Supervised machine learning: A brief primer. *Psychological Medicine (online as a brief primer on supervised ML)*.

: Miguel-Diez, A. e. (2026). A systematic literature review of unsupervised learning algorithms for anomaly detection in network flows. *Journal of Information, Communication and Ethics in Society*.

ain, A. G. (2017). Car acceptability prediction using decision tree based classifiers. *International Journal of Computer Applications*, 21-27.

Breiman, L. (2001). Random Forests. *Machine Learning*, 5-32.

Maulud, A. &. (2021). A Review on Linear Regression Comprehensive in Machine Learning. *Journal of Applied Science and Technology Trends*, 303-310.

Netshamutshedzi, N. e. (2025). A systematic review of the hybrid machine learning models for brain tumor detection and segmentation. *Artificial Intelligence in the Life Sciences*.

Song, Y. Y. (2015). Decision tree methods: applications for classification and prediction. *Shanghai Archives of Psychiatry*, 130-135.