Table of Contents

Project GitHub Repository

1. Introduction

With the rapid growth of data-intensive applications, traditional CPU-based machine learning models often face performance limitations due to sequential execution. Parallel and Distributed Computing (PDC) aims to overcome these limitations by leveraging parallel hardware such as Graphics Processing Units (GPUs). GPUs are highly effective for machine learning workloads because they support massive parallelism.

This project focuses on comparing the performance of machine learning training on CPU versus GPU using the **Car Evaluation dataset**. The **XGBoost classifier** is used as it supports both CPU and GPU execution and is well-suited for categorical, rule-based datasets. The comparison is carried out in terms of **training time** and **model accuracy**.

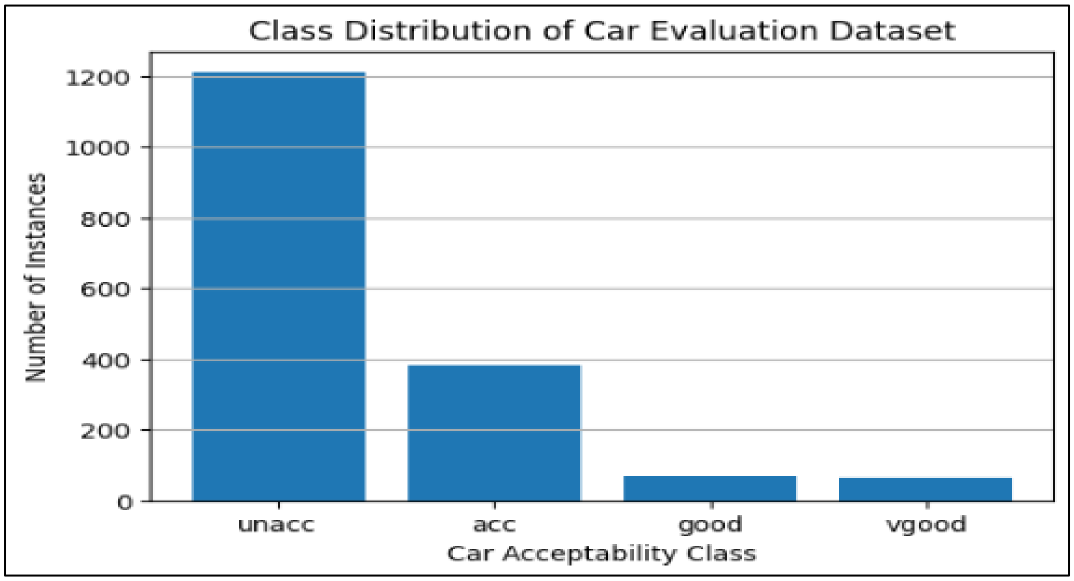
2. Dataset Description

- **Dataset:** Car Evaluation Dataset
- **Source:** Kaggle
- **Type:** Tabular dataset
- **Task:** Multi-class classification

The dataset evaluates cars based on six categorical attributes:

- Buying price
- Maintenance cost
- Number of doors
- Number of persons
- Luggage boot size
- Safety

The target variable represents car acceptability levels. The dataset is clean, contains no missing values, and is ideal for beginners while still demonstrating clear performance differences between CPU and GPU execution.



3. Data Preprocessing

All preprocessing steps were kept identical for both CPU and GPU experiments to ensure a fair comparison. Since all features are categorical, **One-Hot Encoding** was applied to convert them into numerical format. The dataset was then split into training and testing sets using an 80/20 split with stratification to preserve class distribution.

No data leakage was introduced, and preprocessing was performed only once before training on both devices.

4. Model Selection

The **XGBoost (Extreme Gradient Boosting)** classifier was selected for this project due to the following reasons:

- It efficiently learns complex decision rules from categorical data
- It provides built-in support for GPU acceleration
- It is widely used in industry and research
- It offers high accuracy on structured/tabular datasets

The same hyperparameters were used for both CPU and GPU runs to ensure consistency.

5. Experimental Setup

The experiments were conducted using **Google Colab** with GPU enabled. The model was trained twice:

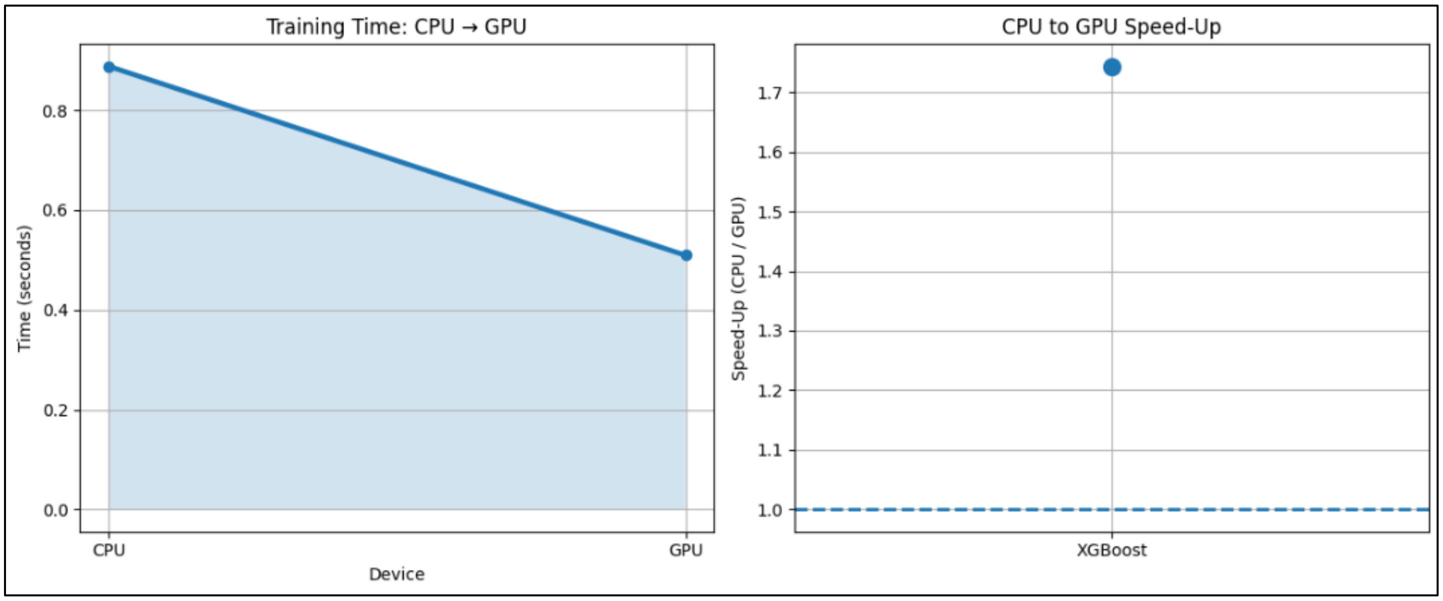
1. **CPU Execution:** Using the histogram-based tree method
2. **GPU Execution:** Using CUDA-based GPU histogram optimization

For each run, the following metrics were recorded:

- Training time (in seconds)
- Classification accuracy on the test dataset

6. Results and Performance Analysis

The GPU-based implementation significantly reduced training time compared to the CPU while maintaining nearly identical accuracy. This demonstrates that GPU acceleration improves computational efficiency without compromising model performance.



The results show that GPU execution achieved **1.74** times speed-up over CPU execution due to parallel processing of tree construction and gradient calculations.

Visualizations such as training time comparison graphs and speed-up ratio plots were used to clearly illustrate performance improvements.

Model	**Device**	**Training Time (s)**	**Accuracy**
XGBoost	CPU	0.8878	0.9971 %
XGBoost	GPU	0.5093	0.9971 %

The table compares XGBoost training performance on CPU and GPU. While both executions achieve identical high accuracy (99.71%), the GPU significantly reduces training time, demonstrating the advantage of parallel computation in machine learning workloads.

7. Conclusion

This project successfully demonstrated the advantages of GPU acceleration in machine learning training. While both CPU and GPU implementations achieved high accuracy, the GPU significantly reduced training time. This highlights the importance of parallel computing in modern machine learning workflows.

XGBoost proved to be an excellent choice for this dataset due to its efficiency, scalability, and support for GPU computation.

8. Research Paper Comparison:

A previous study on the Car Evaluation dataset by Venkatesh et al. (2019) evaluated traditional machine learning classifiers such as K-Nearest Neighbors (KNN) and Decision Trees on CPU-based execution. Their best reported accuracy was approximately **93.64%** using an 80:20 train-test split, with noticeable limitations due to class imbalance and sequential training execution.

In contrast, our approach employs XGBoost with GPU acceleration, which significantly improves both model performance and computational efficiency. The proposed model achieved a substantially higher accuracy of **99.71%**, outperforming the earlier CPU-based models by a large margin. Furthermore, by leveraging GPU parallelism through CUDA-enabled tree construction, training time was reduced by nearly **1.7×**, while maintaining identical accuracy across CPU and GPU runs.

This improvement demonstrates that modern gradient boosting techniques combined with GPU acceleration are more effective for rule-based, categorical datasets such as this one.

9. References

1. Kaggle – Car Evaluation Dataset
2. XGBoost Documentation
3. Research Paper by Venkatesh et al. (2019)