National College of Ireland

Enhancing the Efficiency of Heart Disease Prediction Using Cloud Machine Learning Techniques

MSc Research Project
Cloud Computing

Hui Huang
X22180966

School of Computing
National College of Ireland

Supervisor: Ahmed Makki

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name: ……Hui Huang……

Student ID: ……x22180966……..……

Programme: ……MSc in Cloud Computing…… **Year:** ……2024-2025..

Module: ……Research Project……..……

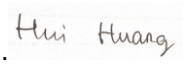
Supervisor: ……Ahmed Makki……..……

Submission Due Date: ……12th December 2024……..……

Project Title: Enhancing the Efficiency of Heart Disease Prediction Using Cloud Machine Learning Techniques

Word Count: ……7352…… **Page Count**……23……..……..

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: ……  ……

Date: ……11th December 2024……

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Enhancing the Efficiency of Heart Disease Prediction Using Cloud Machine Learning Techniques

Hui Huang
x22180966

Abstract

Heart disease has been a critical focus of international medical attention as it remains to be the major cause of global death. Accurate heart disease prediction is a useful part of medical solutions. Traditional prediction approaches often face efficiency limitations in terms of prolonged training time, suboptimal resource utilization, and difficulties in processing large-scale datasets. To address these challenges, this study investigated the potential advantages of using cloud machine learning techniques to enhance the efficiency and performance of heart disease prediction.

Logistic Regression, Random Forest, and XGBoost algorithms were implemented with automated hyperparameter tuning to optimize models in both environments. Random Forest algorithm was used to assess the impact of parallel processing on various nodes and processes. The results show significant improvements in distributed computing and automated hyperparameter tuning scenarios. The training time and resource utilization have been reduced by 75.2% and 18.9 % in Random Forest training with the configuration of 2 nodes and 2 processes for each node, while the accuracy and ROC AUC score increased by 0.30% and 0.25%. In the setup of hyperparameter optimization in a cloud-based environment, the training time in Logistic Regression and XGBoost has been reduced by 55.4% and 8.7 compared with a single-node local environment.

This study offers a practical solution for healthcare institutions to employ cloud computing for medical decisions or their clinical applications, especially those with restricted computational resources. It aims to narrow the gap between limited computational resources and the increasing demand for big data in heart disease instances.

1 Introduction

1.1 Background

According to the estimated data, around 17.9 million people die from cardiovascular diseases annually (Gaidai *et al.* 2023). Celermajer *et al.* (2012) argued that early detection and efficient medical treatment are useful approaches to reduce the risk of patients suffering from heart disease attacks.

Heart disease prediction modeling is considered a critical tool for medical institutions to detect cardiovascular diseases (CVDs) and facilitate timely medical decisions. An increasing number of institutions are adopting Artificial Intelligence (AI) to assess the health risks of patients to develop personalized treatment plans and optimized medical resource allocation. A model of prediction of sudden cardiac arrest with hypertrophic cardiomyopathy developed

by the Mayo Clinic presents a more accurate forecast of sudden cardiac arrest than traditional methods (Bhowmik *et al.*, 2024).

However, considerable attention must be paid when facing increasing various formats of reports and large-size medical examination images being issued by different institutions, the performance and model training time are affected by the limited scalability in the local prediction environment. These limitations are typically reflected in the model's efficiency and larger-scale data processing ability. Comprehensive cloud machine learning techniques with distributed processing capabilities and optimized resource management facilitate efficient solutions to address these challenges and provide integrated implementations for future research in healthcare applications. This research demonstrates how to enhance the performance and efficiency in model training with distributed computing configuration and further hyperparameter optimization setups and provides experimental evidence for addressing the restricted resource challenges of medical institutions.

1.2 Research Challenges

A challenge in heart disease prediction is the efficiency of model training and massive data processing. The main downside of heart disease prediction in an on-premise environment is that limited local resources lead to constraints in processing speed and resource utilization. According to Rajkomar et al. (2018), predictive modeling faces increasing demand for data preprocessing, merging, and data cleaning which accounts for 80% of the modeling workload, while there remains a need to improve the scalability in local systems. The inefficient resource utilization in local systems often makes it impractical to maintain scalable to handle large-scale datasets or more complex modeling.

Algorithms such as Logistic Regression (LR), Random Forest (RF), and XGBoost are commonly utilized in heart disease prediction for their nature of classification and regression abilities that deal with various data patterns. However, these models are limited by the computational constraints in the local environment. Similar to the computational challenges of ECG interpretation tasks with artificial neural networks (Hannun et al., 2019), these limitations restrict their scalability and efficiency when handling large-scale datasets.

Employing cloud machine learning techniques on platforms such as Azure Machine Learning Studio provides an effective solution to address these limitations. The scalability, efficiency, and performance of heart disease prediction can be enhanced using distributed computing and efficient resource management in a cloud-based environment like Azure.

1.3 Research Objectives

This paper aims to address the gap in the literature by conducting a comparative evaluation of on-premise and cloud-based environments for heart disease prediction. The aim of this study is to present the practical advantages of the cloud-based platform advantages, in terms of shortening training time and improving performance in models. The implementation of this

project is expected to benefit clinical decision-making and facilitate the development of new solutions in medical artificial intelligence.

This paper aims to answer the primary research question: Does cloud-based machine learning significantly reduce training times compared to local implementations in healthcare prediction models? This question explores the potential abilities of cloud machine learning in the medical industry. By utilizing distributed computing and available resources, Azure Machine Learning Studio provides an efficient solution to overcome the challenge in modeling that is associated with limitations in computing resources and scalability. The study aims to evaluate whether cloud-based modeling can improve processing speed and predictive performance as applied to large-size healthcare datasets.

This study compared the performance of logistic regression, random forest, and XGBoost models in local and cloud environments, and qualified the impact of cloud computing on processing efficiency and performance in modeling through metrics such as training time, accuracy, F1- score and recall score. In this context, we tried to present cloud computing has its advantages in real-world medical practice.

1.4 Research Overview

This study has two limitations. Firstly, the datasets used in the implementation may be not a valid representative of the real-world heart disease data, and the generalizability of the research results could be affected. The second is the comparability between local and cloud-based environments. Although efforts are made to maintain hardware consistency in them, other external factors such as network latency may cause different results in performance metrics. Worthy to note that this study primarily evaluates training time and predictive performance of models, while other considerations such as long-term cost efficiency are not concluded in its scope.

The remainder of the paper is structured as follows: section 2 presents the related works regarding heart disease prediction using machine learning, the challenges of processing efficiency and models' performance, and the advantages provided by cloud-based platforms in the healthcare industry. Section 3 describes the proposed research design in terms of data collection, data preprocessing, data transformation, and training. To ensure the fairness of experiments, both environments use the same Azure computing hardware configuration. It details the training process by adopting Logistic Regression, Random Forest, and XGBoost algorithms in both environments. Distributed computing and automated hyperparameter tuning are conducted in cloud-based implementation experiments to boost modeling efficiency and performance. Finally, it discusses evaluation metrics used to measure and compare results. Section 4 evaluates the performance with comparative environments. Section 5 discusses the analysis of the results and concludes the paper.

2 Related Work

2.1 Heart Disease and AI

Heart disease has been considered a major global health threat long-term as it occupies almost one-third of the mortality. The disease commonly includes various diseases such as coronary artery disease, heart failure, valve disease, arrhythmia, and stroke. Various studies have been conducted on the significant health impact caused by cardiovascular diseases. Early works mainly focused on identifying related risk factors in terms of high cholesterol, hypertension, obesity, and unhealthy lifestyle. In this research, the authors investigated and summarized 10 CVD risk factors that include unhealthful nutrition, physical inactivity, dyslipidemia, hyperglycemia, high blood pressure, obesity, thrombosis, kidney dysfunction, genetics, and select populations(Bays et al., 2021). With the development of preventive cardiology research, related researchers may benefit from risk factors investigations.

Early detection and treatment can significantly mitigate the impact of heart disease, as emphasized by a considerable amount of research that highlights the benefits of timely interventions. For example, Angeli and co-workers measured that early stain therapy reduced the mortality of patients with acute coronary syndrome (ACS) compared to other control groups who had not initiated saint treatment on the first day in hospital(Angeli et al., 2012). An increase in studies in predictive models becomes a critical tool in the cardiovascular field. Simulation results show that Machine Learning(ML) has significant potential to transform heart disease risk characterization or biomarker recognition in epidemiology. For example, Adler and co-workers compared the AUC results of the boosted decision algorithm training group to 2 different external validation groups (Adler et al., 2020). Their experiments showed a higher Area Under the Curve(AUC) of 0.88 while others were 0.84 and 0.81. Their findings indicate that the use of ML can improve the evaluation of heart failure patients with the challenge of the complex relationship of features. These results highlight that ML improves early diagnosis and intervention for developing efficient heart disease management. Given underlying complex risk factors, adopting ML techniques can boost the accuracy of predictions and provide more timely treatment guidance for medical institutions.

2.2 Modern Machine Learning Applications in Healthcare

2.2.1 AI-Driven Solutions in Healthcare

The combination of various data and advanced modeling techniques can improve the healthcare system by providing accurate prediction results. For example, the study developed an ML model by using AutoPrognosis outperformed traditional cardiovascular risk predictors such as walking pace and self-reported health checks including 473 variables. The accuracy of ML model has been shown higher accuracy when compared to conventional CVD risk factors (Alaa et al., 2019). Google's AI research department initiated the DeepMind Health project to improve healthcare services by analyzing medical data. It is worthwhile noting that for further exploring deep learning methods with ML, this project collaborated with Moorfields Eye Hospital Foundation to carry out eye tests based on the data provided (Mesko, 2017).

Machine learning applications in medical examination imaging have presented significant accuracy in disease diagnosis. Peyret and co-workers used an algorithm that combines with a convolutional neural network to detect and locate invasive carcinoma on breast whole-slide images (Peyret et al., 2023). Their accuracy, recall, and precision results were reported as 92.1%, 95.0%, and 73.9% for the limited reference dataset. It shows that the AI approach can be used to help in pathology practice. Watson for Oncology (WFO) is a clinical-level of support system from IBM that provides prompts to cancer specialists and patients with timely treatment if they apply. A study conducted a comparative evaluation of the accuracy of treatment issued by the WFO and Multidisciplinary Team (MDT). They used RevMan5.3 Software for mete-analysis on 2,463 patients that involved 9 different researches, and the comparative result showed high consistency with an overall concordance rate of 81.52% (Zhou, Zhiying and Li, 2021). Various formats of healthcare data range from electronic health records (EHR) to medical images and other grouped data. ML plays a critical role by integrating different types of medical data to provide insights accordingly. A review study conducted an analysis by searching 34 related research and tried to find a research pattern for evaluating multimodal fusion in clinical predictions (Farida et al., 2022). In their paper, they focused on studies that combine EHR data with medical imaging data to explore the AI methods for clinical applications. The result shows that, compared to the traditional single-modality models used for the control task, multimodality fusion models have better performance. It also emphasized that using ML fusion methods improves AD diagnosis.

ML has been widely applied in healthcare, especially, an increasing number of studies focused on how to use ML techniques to lower heart disease risks. Compared to conventional risk factors, ML models that are applied to cardiovascular diseases have shown significant progress in high-risk individual identification. Mohan and co-workers found that a new approach to feature selection can largely improve the performance of heart disease modeling results (Mohan, Thirumalai, and Srivastava, 2019). This paper begins by introducing the Hybrid Random Forest with Linear Model (HRFLM) that combines mixed random forest algorithms. The accuracy of this approach reached 88.7%, which highlighted the potential for developing new feature selection approaches in ML to obtain a more comprehensive understanding of critical features for assisting in making decisions of early detection and timely treatment. Several expert systems are used to improve the prediction of early-staged heart disease to save more intervention opportunities for patients. A study introduced a diagnostic system with an optimized XGBoost algorithm, Bayesian hyper-parameter adjustment, and One hot encoding, the accuracy achieved 91.8%. Compared with modeling results from random forests and extra trees, the accuracy, sensitivity, specificity, F1- score, and AUC value results are improved. Their study showed that ML techniques that they proposed are reliable for predicting heart disease in medical institutions (Budholiya, Shrivastava, and Sharma, 2022). The progress of deep learning in heart disease prediction indicates that hybrid neural network frameworks that integrate various models to enhance prediction accuracy are examined to have significant potential. This study proposed a Hybrid Deep Neural Networks and achieved a 98.86% accuracy on multiple heart disease datasets (Al Reshan et al., 2023). In their study, the approach designed with complex ML techniques by adopting an integration with Artificial Neural Networks (ANN), Convolutional Neural Networks (CNN), and Long Short-Term Memory (LSTM) models, which present great potential to be embedded into healthcare systems to improve heart disease patient care.

2.2.2 Heart Disease Prediction Models

Logistic Regression (LR), Random Forest (RF), and XGBoosting are among the most widely used models for heart disease prediction due to their powerful strengths in terms of accuracy, interpretability, and scalability. LR is a commonly investigated classification algorithm that excels at capturing linear relationships among risky factors and provides a simple classification framework for assessing heart disease risk. A study used LR with implementing advanced feature selection approaches, and the 87.10% accuracy of the modeling result emphasized that the performance of LR modeling could be reliable in providing insights for complex heart disease datasets (Ambrish et al., 2022), and it values in providing fast and simple prediction insights for health institutions with limited computational resources.

RF excels at identifying critical factors during training by excellently managing relationships of input variables in complex datasets. Saikumar and Rajesh found that using an optimized RF algorithm can efficiently address the lack of real-time diagnosis for issuing an easy heart operation determination procedure. In their study, 98.74% accuracy has been achieved by advanced feature extraction approaches (Saikumar and Rajesh, 2020). The nature of this algorithm has proven that it is a very robust approach for providing highly accurate diagnose in real-time medical diagnosis due to it can aggregate multiple decision trees to dig out the key factors in complex medical instance scenarios.

XGBoost algorithm boosted the prediction performance that leveraging iteratively optimizing weak learners to decrease errors. It excels at processing high-dimensional datasets with variables in non-linear relationships. Yang and Guan introduced a framework that combines SMOTE-XGBoost algorithm to predict heart disease risk and obtained a high accuracy of 94.44% (Yang and Guan, 2022). The improvement shows that XGBoost is effective in handling complex and multivariate cardiovascular risk factors during data processing.

2.3 Challenges in Local Machine Learning Predictions

Two major challenges in implementing machine learning for predicting heart disease in local healthcare environments are limited scalability and inefficient resource utilization. Specifically, these limitations affect heart disease prediction performance for healthcare institutions with restricted computational resources.

2.3.1 Resource Constraints in Healthcare

While implementing machine learning solutions for heart disease prediction, healthcare institutions such as small clinics or medical centers in resource-restricted areas struggle with limited computational power. These disadvantages have grown with the development of big data and complex target models, making it difficult to meet the increasing computational demands. Additionally, modern healthcare applications also handle various formats of data that range from detailed patient records to high-resolution medical imaging that require sufficient computing power. Lacey et al., 2016 point out that as the large amount of data increases, more advanced complex computing infrastructures are required for facilities with deep learning solutions. These limitations not only burden the lack of computing support for the institution but lead to underperformance in modeling and prolonged processing time, which can cause patients to fail to gain timely medical treatments.

2.3.2 Scalability Challenges in Real-time Processing

Another computing challenge has been brought by the continuous monitoring and real-time analysis in healthcare. To date, the popularity of wearable electrocardiogram devices provides valuable insights for continuous monitoring and early detection of heart conditions, while it increases the computing burden for real-time processing and massive data processing. This level of real-time analysis and prediction requires powerful computation support which also is the drawback of local implementation environments or institutions. Liu et al., 2018 developed an IoT-based wearable electrocardiogram system using cloud resources to address the computing demands by managing a large-scale electrocardiogram database with the cloud machine learning platform. The F1-score of the heart disease prediction in their early detection has achieved 99.5% with excellent computing efficiency. Their work shows how cloud-based solutions can efficiently address the scalability limitations compared to local environments.

These challenges highlight that cloud-based environments perform efficient processing speed in real-time, handle the massive data in heart indicators, and maintain predictive accuracy, while tailoring to address the increasing demand for scalable computing solutions.

3 Research Methodology

3.1 Experimental Design

This study adopts a comparative experimental design to identify and evaluate the efficiency and performance improvement of cloud-based machine learning techniques in heart disease prediction. The study implemented the cloud-based implementation on the Azure Machine Learning Studio while maintaining the same compute instance in the local environment to ensure fair comparisons. Configurations are as follows in Table 1:

Table 1 – Configurations in Both Environments

Environment	Core Libraries/ Tools	Virtual Machine	Machine Learning Algorithms
Local	numpy, pandas, scikit-learn, Optuna	Standard_DS3_v2 (4 cores, 14 GB RAM, 28 GB disk)	Logistic Regression, Random Forest, XGBoost
Cloud	Azure ML SDK v2, numpy, pandas, scikit-learn, Optuna	Standard_DS3_v2 (4 cores, 14 GB RAM, 28 GB disk)	Logistic Regression, Random Forest, XGBoost

The local environment is implemented on a base of Python libraries, while the cloud environment is set up with computational specifications that provided by the Azure Machine Learning Workplace. Enabling fair comparison, both environments are using the same hardware configuration within the same virtual machine that was built in the Azure ML studio. Figure 1 below shows the framework of the experimental tasks. This comparative framework enables a comprehensive comparison of metrics and maintains the data quality and consistency.

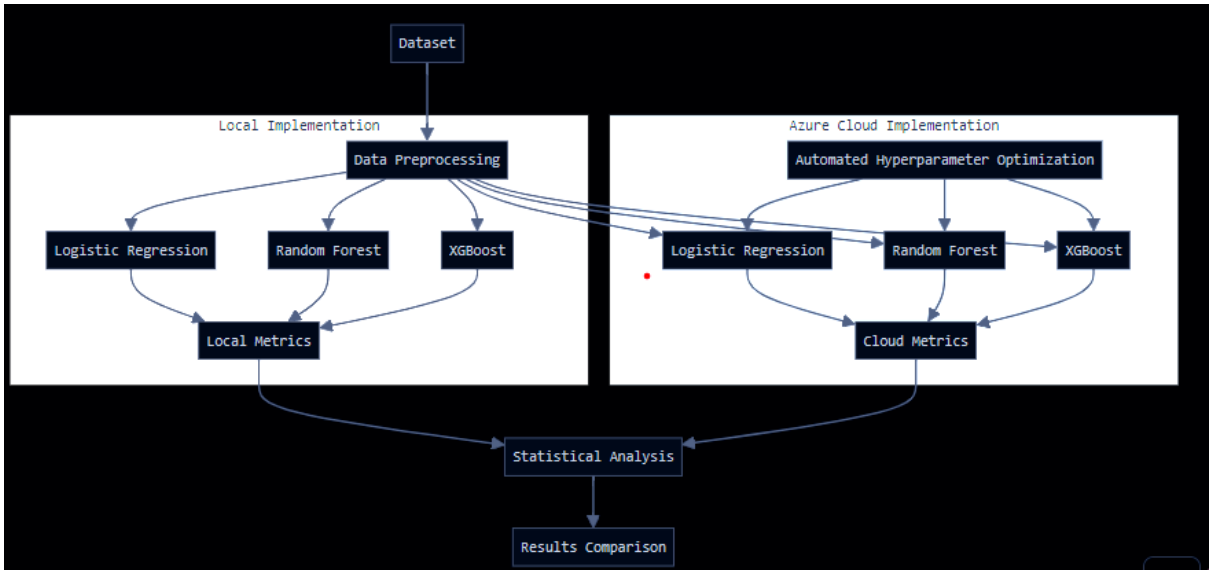


Figure 1 - The Experimental Design Framework

3.2 Algorithm Implementation

This study selected 3 algorithms to implement machine learning in terms of Logistic Regression, Random Forest, and XGBoost. The selection principle is they are greatly used and proved their effectiveness in modeling tasks of binary classification. The implementation phase covers machine learning processes in these 3 algorithms. Additionally, the hyperparameter optimization approach is implemented with the Azure ML SDK in the cloud-based environment.

3.3 Data Collection and Preprocessing

The study utilized a comprehensive healthcare dataset obtained from Kaggle's Heart Disease Indicator Dataset, which was originally from the CDC behavioral Risk Factors Surveillance System survey. This dataset consists of 253,680 survey records with a balanced population distribution aged 18 to 65 years old and above. Multiple dimensions of cardiovascular-related health indicators are covered in this dataset. Among them, societal factors such as demographic indicators include age, education level, gender, and income condition. Clinical measurements contain BMI, blood pressure, cholesterol levels, and other commonly captured health indicators. Especially, the distribution of hypertension and elevated cholesterol levels showed 28.3% and 39.1%, while 13.2% of the cases had diabetes. Lifestyle factors include general health and mental health perceptions that provide an overall vision of cardiovascular health risk factors.

To ensure the data quality and consistency for implementations in both environments, the data preprocessing followed the systematic approach. It utilized binary encoding to handle the categorical features and processed the numerical variables with the One-hot encoding method for effectively capturing non-linear relationships among features. Standard scaling was adopted for all variables' normalization operation, which ensures all variables can contribute to the modeling proportionally. To evaluate the benefits of distributed computing in a cloud-based environment with datasets in size of 200,000 records can reflect the maximum size of heart disease datasets in real-world requirements.

3.4 Data Exploration

Firstly, variables were categorized based on the data types and distribution characteristics in terms of dividing the features into binary variables and numerical variables. This approach guided the process of classification of data that were required through visualization methods. To explore the distribution of binary and categorical variables, bar charts were utilized to present the frequency distributions. For numerical variables, it adopted histograms with density estimation overlays to explore the continuous distributions. Additionally, a heatmap visualization was conducted to analyze the relationships between features. Through a color-coded matrix with correlation coefficients, the heatmap interpreted the relationships visually. Lastly, boxplots and histograms were plotted to show the detailed distribution of numerical variables, which enables potential outliers to be identified thoroughly. To ensure the subsequent data preprocessing operation proceeds, this exploration process set a solid foundation.

3.5 Performance Measurement

In the study, Python's time module is imported and used for measuring modeling procedures that start from preprocessing in both environments. Additionally, both performance is evaluated using accuracy, F1 score, recall score, precision score, and AUC-ROC value. To monitor the resource consumption in both environments, the resource utilization is measured with CPU usage and memory consumption. In the cloud-based implementation, the study also facilitated distributing by assigning processes strategy and hyperparameter tunning strategy with Azure Machine Learning Studio.

3.6 Algorithm Selection Rationale

This research selected LR, RF, and XGBoost in the experimental tasks based on their strengths in dealing with healthcare data and the well-discussed effectiveness during heart disease prediction. The computing efficiency was effectively compared using them in modeling. As the baseline model in healthcare modeling, LR is well known for its interpretability by presenting the capability of providing clear interpretations to understand the relationship between risky factors and response results. Additionally, both environments in this study can benefit from their low requirement in computing which makes it an ideal benchmark algorithm. RF was selected for its robust performance in processing complex relationships in data due to its integration of multiple decision trees for better capturing the non-linear relationship between indicators, which can benefit from the distributed computing approach in the cloud-based environment. Enabling significant performance in capturing complex feature interactions with the complex essence of the algorithm, XGBoost was selected to handle complex scenarios with highly imbalanced data, which is suitable for the selected datasets. Unlike the benchmark algorithm, XGBoost requires intense computational resources based on its gradient boosting framework, while it also sets the testing boundary for better evaluation of the advantages of distributed computing in the cloud-based environment.

3.7 Experimental Controls

Enabling fair comparison for all experimental tasks in both environments, all implementations were conducted in Azure Machine Learning Studio's Jupyter Notebook feature. To eliminate performance results that caused by hardware configurations, the Standard_DS3_v2 virtual machines (4 cores, 14 GB RAM, 28 GB disk) were the identical instance running for the tasks. Additionally, this study set up a controlled and fair framework

for comparing the outcomes of both environments by maintaining a consistent computing specification employed in the identical Jupyter Notebook platform.

To maintain a fair environment for evaluating metrics, this study performed consistent monitoring in terms of the training time, and the utilization of memory and CPU for each training task in both environments. Accurate performance and efficiency comparisons can be evaluated between those environments by implementing these standard measurements with consistent coding conditions.

4 Design Specification

This section aims to introduce the design specifications to guide the implementation and evaluation of machine learning models for heart disease prediction using local and cloud-based environments. The study explores the performance difference in implementations between the local environment and cloud-based methods that use Azure Machine Learning Studio.

These design specifications outline the core technology, architecture, algorithms, and performance. In the local environment, all procedures used Python's built-in libraries, while the cloud-based approach in Azure Machine Learning Studio implemented the identical preprocessing phase to ensure consistency and fairness in experiments. Additionally, the identical compute instance was used in both implementations. With the support of the Azure SDK and integrated features such as Azure Machine learning, the Azure implementation used distributed computing and automated hyperparameter tuning to improve the efficiency and performance of models. Algorithms selection follows the widely used research for classification tasks, this study used Logistic Regression, Random Forest, and XGBoost. A range of metrics are used to measure the performance in terms of training time, accuracy, F1 score, recall score, precision score, AUC-ROC value, and resource consumption. A thorough comparison among these metrics enables the effectiveness of local and cloud-based implementations while highlighting their advantages and limitations.

4.1 System Architecture Design

This study implemented a comprehensive system architecture that includes local and cloud-based environments to develop and evaluate heart disease prediction models. The experimental design focused on direct comparison that was introduced with cloud features to achieve optimization in models.

4.2 Development Environment

The process of the development stage employed Jupyter Notebooks as the integrated development environment to facilitate the coding and real-time visualization of evaluation results. Furthermore, future work can be conducted in this repeatable environment to further explore potential benefits for research or practice.

4.3 Local Environment Architecture

On-premise implementation leveraged the built-in libraries of Python to conduct the data mining and model training processing. Both environments utilized pandas to handle the data, numpy to process the numerical features and scikit-learn for machine learning. Three widely

discussed algorithms were implemented to compare the results based on various advantages in terms of LR within class weight balance, RF with ensemble learning function, and XGBoost for optimizing the gradient boosting.

4.4 Cloud Architecture

The Cloud implementation adopted the Azure Machine Learning Studio with integrated components:

(1) Environment Configuration:
   i. Azure compute instance
   ii. Azure Jupyter Notebooks
   iii. GPU-enabled environment within PyTorch
   iv. Distributed training approach for parallel processing

(2) Processing Management:
   i. Azure machine learning workplace
   ii. Credential management
   iii. Data asset management

4.5 Data Pipeline Architecture

The selected dataset is sourced from the Kaggle, which includes 253,680 survey records. Enabling the consistent preprocessing stage in both environments, the dataset was imported locally through a local file address. The datasets cover demographic information, heart-related health indicators, and other lifestyle factors.

A systematic preprocessing pipeline was set for data transformation:

(1) Pipeline management
   i. consistent implementation in both environments
   ii. persistent preprocessing parameters storage procedure
   iii. Reproducible transformation
(2) Feature Engineering
   i. Label encoding to handle 17 categorical variables
   ii. One-hot encoding to handle specific features
   iii. Standardization processing for numerical features

The hyperparameter optimization approach was implemented in Azure Machine Learning:

(1) Parameter Space Exploration:
   i. Optimizing the tree depth
   ii. Learning rate calibration
   iii. optimizing the estimator count
   iv. tuning the child's weight
(2) Processing Optimization
   i. ROC-AUC value
   ii. 20 trials exploration sequence
   iii. automated parameter selection
(3) Distributed Computing Framework
   i. distribution based on PyTorch
   ii. Multiple process mechanism
   iii. single node configuration

4.6 Model Performance Evaluation Framework

(1) Model Performance Metrics:
   i. Classification accuracy
   ii. ROC curve value
   iii. training time
(2) Resource Monitoring
   i. Memory consumption record
   ii. CPU consumption tracking

4.7 Dual Environment Comparison

The design specification ensures a systematic configuration and development for heart disease prediction in both environments and conducts a fair comparison of performance:
(1) ensures the consistency of the metric evaluation in both environments
(2) validates the performance difference statistically
(3) assesses the model training efficiency
(4) analysis of the resource utilization

5 Implementation

This section outlines an overview of the final stage of the solution, while it primarily introduces the output, adopted tools, and methods that are used during experiments.

5.1 Data Sampling Strategy

I created various sub datasets in sizes of 5000, 20000, 100,000, and 200000 from the shuffled raw dataset. Figure 2 below shows the data shuffling and splitting process. Enabling rapid iterations in training and obtaining an optimized performance with sufficient data, the size of 20,000 datasets was used in the parameter tuning experiments. Additionally, I used the dataset that contains 200,000 samples to effectively evaluate the scalability and efficiency across various configurations in different nodes and processes. This strategy ensures the requirements for accuracy improvement using parameter tuning methods and the potential scalability with cloud computation power. Meanwhile, the data consistency in all experiments can be ensured by random sampling during the data sampling stage.

```
train_data_shuffled = data_local.sample(frac=1, random_state=42).reset_index(drop=True)
train_data_shuffled.iloc[:5000, :].to_csv('../data/train_5000.csv', index=False)
train_data_shuffled.iloc[:5000, :].to_csv('../data/test_5000.csv', index=False)

train_data_shuffled.iloc[:20000, :].to_csv('../data/train_20000.csv', index=False)
train_data_shuffled.iloc[:20000, :].to_csv('../data/test_20000.csv', index=False)

train_data_shuffled.iloc[:100000, :].to_csv('../data/train_100000.csv', index=False)
train_data_shuffled.iloc[:100000, :].to_csv('../data/test_100000.csv', index=False)

train_data_shuffled.iloc[:200000, :].to_csv('../data/train_200000.csv', index=False)
train_data_shuffled.iloc[:200000, :].to_csv('../data/test_200000.csv', index=False)

train_5000 = train_data_shuffled.iloc[:5000, :]
train_20000 = train_data_shuffled.iloc[:20000, :]
train_100000 = train_data_shuffled.iloc[:100000, :]
train_100000 = train_data_shuffled.iloc[:200000, :]
```

Figure 2 - Data Shuffling and Splitting

5.2 Hyperparameter Tuning for Model Performance

To achieve an enhanced performance in LR, RF, and XGBoost for heart disease modeling, I mainly implemented multiple hyperparameter adjustments and compared the results

including default parameters, a manual configuration in the local environment, and an automated hyperparameter tuning approach using Azure Machine Learning in the cloud-based environment.

Firstly, the baseline of the comparison was implemented with the Optuna framework imported in the local approach.

```
# Run Optuna optimization for each model
for model_name in models:
    print(f"\nOptimizing {model_name}...")

    study = optuna.create_study(direction='maximize')  # Maximize ROC AUC
    study.optimize(lambda trial: objective(trial, X_train_res, X_test, y_train_res, y_test, model_name), n_trials=50)

    # Output the best trial
    best_trial = study.best_trial
    print(f"\nBest trial for {model_name}:")
    print(f"  ROC AUC: {best_trial.value}")
    print(f"  Params: {best_trial.params}")
```

Figure 3 - Optuna Optimization in the Local Environment

Finally, the cloud-based approach also used the Optuna framework to optimize the parameters of the trained models. In addition, it adopted distributed training across 4 compute instances that enabled the improvement of model performance by speeding up the training processing. The key point of this approach is largely made use of the scalability of cloud resources. Figure 4 and Figure 5 show the parameter optimization operation used in the cloud-based environments for distributed training.

```
command_job = command(
    code="./scripts",
    command="pip install scikit-learn xgboost optuna imbalanced-learn && python hyper1128.py --input_dat
    environment=env,
    compute="x221809662",
    # allocate 4 instances for distributing training
    distribution={
        "type": "pytorch",
        "process_count_per_instance": 1,
        "instance_count": 4
    },
    inputs={"data": Input(type="uri_file", path=data_asset.path)}
)
```

Figure 4 - 4 Nodes & 1 Process Allocation in the Cloud Approach

```
# Train and evaluate each model
for model_name, config in models.items():
    print(f"\n{'='*60}\nOptimizing {model_name}\n{'='*60}")

    study = optuna.create_study(direction='maximize')
    study.optimize(
        lambda trial: config['objective'](trial, X_train_res, y_train_res, X_val, y_val),
        n_trials=args.n_trials
    )

    best_params = study.best_params
    best_params.update(config['extra_params'])
    print(f"\nBest Parameters for {model_name}:", best_params)
```

Figure 5 - Optuna Optimization in the Cloud-based Training Script

5.3 Distributed Computing for Training Efficiency

In this implementation, various distributed computing approaches for RF modeling were examined to improve the computational efficiency by reducing the training time.

In the local environment, a baseline benchmark was set up using RF training with default configuration. In the cloud-based environment, single-node single-process and single-node dual-process configurations were conducted to maintain a consistent configuration environment for comparison. To emphasize the different results of training time, dual-nodes dual-process was implemented by utilizing Azure's distributed computing power. Figure 6 and Figure 7 below demonstrate the cloud-based implementations:

```
# connect to my workplace
ml_client = MLClient.from_config(credential=DefaultAzureCredential())

# create the environment
env = ml_client.environments.get(name="AzureML-ACPT-pytorch-1.13-py38-cuda11.7-gpu", version="1")

# define the data asset
data_asset = Data(
    name="heart-disease-data",
    path="../data/train_200000.csv",
    type=AssetTypes.URI_FILE
)

data_asset = ml_client.data.create_or_update(data_asset)

# create the training job
job = command(
    code="./scripts",
    command="python train1127.py --input_data ${{inputs.data}} --output_model outputs/heart_disease_model --model_type all",
    inputs={
        "data": Input(type="uri_file", path=data_asset.path)
    },
    environment=env,
    # specify the azure compute instance
    compute="x221809662",
    display_name="heart-disease-model-training"
)

# uploade the job
returned_job = ml_client.jobs.create_or_update(job)

# check the job result
ml_client.jobs.stream(returned_job.name)
```

Figure 6 - Default Configurations in the Azure Implementation

```
# create the distributed training job
job = command(
    code="./scripts",
    command="python azure_distributed_1130.py "
            "--input_data ${{inputs.data}} "
            "--output_model outputs/heart_disease_model "
            "--model_type all "
            "--distributed",
    inputs={
        "data": Input(type="uri_file", path=data_asset.path)
    },
    environment=env,
    compute="x221809662",
    display_name="distributed-heart-disease-model-training",

    # allocate the number of the process and the node
    distribution={
        "type": "PyTorch",
        "process_count_per_instance": 2,
        "node_count": 1
    }
)
```

```
# create the distributed training job
job = command(
    code="./scripts",
    command="python 2nodes_opt.py "
            "--input_data ${{inputs.data}} "
            "--output_model outputs/heart_disease_model "
            "--model_type all "
            "--distributed",
    inputs={
        "data": Input(type="uri_file", path=data_asset.path)
    },
    environment=env,
    compute="x221809662",
    display_name="distributed-heart-disease-model-training",

    # allocate the number of the process and the node
    distribution={
        "type": "PyTorch",
        "process_count_per_instance": 2,
        "node_count": 2
    }
```

Figure 7 - Parallel Computing in the Azure Implementation

5.4 Tools

5.4.1 Programming Language

Python

5.4.2 Libraries

Scikit-learn, Azure Auto ML, PyTorch, Optuna

5.4.3 Platforms

Azure Machine Learning Studio, Azure Virtual Machine instance

6 Evaluation

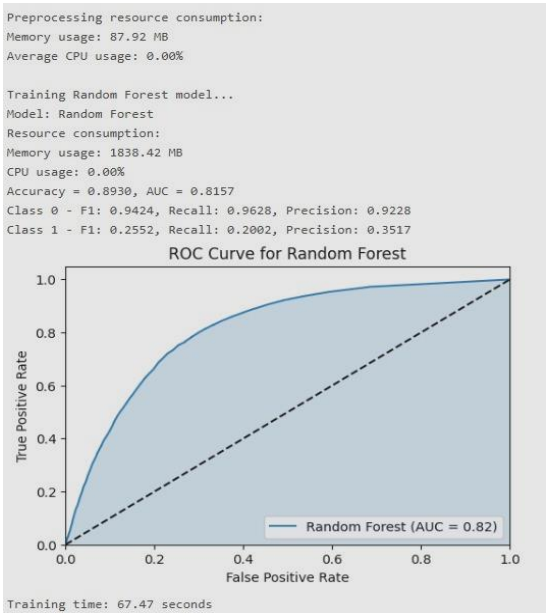
This section presents an analysis of heart disease prediction training in both local and cloud-based environments. It focused on effectively evaluating the impact of cloud resources on boosting the efficiency of machine learning models compared with on-promise implementation, which lacks the scalability of cloud resources. This evaluation is structured into various case studies to form a discussion based on findings of the impacts of cloud resources.

6.1 Experiment Setup and Fairness Assurance

In this experiment, I used the Azure Machine Learning platform across all model training tasks. The fairness of the experiment is ensured by both implementation scenarios were conducted on the same hardware configuration, which is the instance in size of "Standard_DS3_v2 (4 cores, 14 GB RAM, 28 GB disk)". Additionally, both implementations were using the same data preprocessing procedure. This approach ensures a fair experiment configuration and focuses on the comparative benefits of cloud resources by removing the impact of hardware and data preprocessing changes.

6.2 Case Study 1: Local & Cloud Environment (1 Node, 1 Process)

In case study 1, I compared the performance of the Random Forest (RF) algorithm in both environments with the default configuration, which is 1 node and 1 process. This setup aims to establish a benchmark performance for both settings while providing a fundamental reference for further experiments. It also shows the baseline performance without distributed training. Figure 8 below shows the training time and resource utilization in the 2 environments that present the efficiency in both environments.



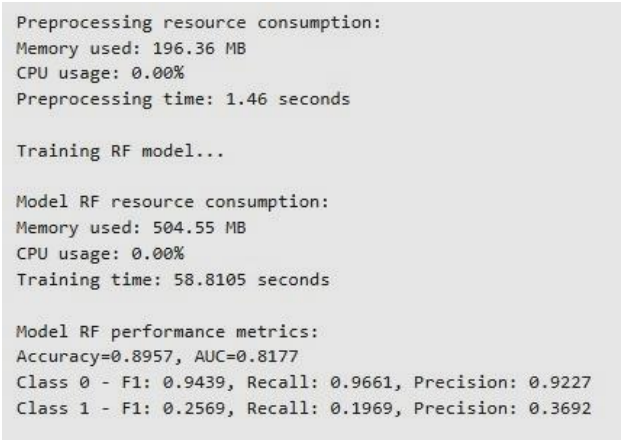


Figure 8 - The Experimental Results in Both Environments

Table 2: Performance Comparison between 2 Environments in Default Configuration

Metric	Local	Cloud
Training Time(seconds)	67.47	58.81
CPU Usage (%)	0.00	0.00
Memory Usage (MB)	1838.42	504.55
AUC Score	0.8157	0.8177
Accuracy	0.8930	0.8957
Resource Efficiency (time/resource ratio)	0.0367	0.1165

Note: No Distributing Computing, 1 Node - 1 Process

As shown in Table 2, to set up a benchmark metric experiment with no distributed practice, the cloud implementation reduced training time by 13% and significantly reduced memory consumption, while maintaining the same accuracy.

6.3 Case Study 2 The Cloud Environment (1 Node, 2 Processes)

Based on the result of the previous case, this study adds up 2 processes in the same training task to explore whether the modeling efficiency could be impacted by more processes in one node. Additionally, the training time and the resource ratio were compared. The results are shown in Figure 9 as follows:

```
Training RF model...

Model RF resource consumption:
Memory used: 506.70 MB
CPU usage: 100.20%
Training time: 55.2945 seconds

Model RF performance metrics:
Accuracy=0.8957, AUC=0.8177
Class 0 - F1: 0.9439, Recall: 0.9661, Precision: 0.9227
Class 1 - F1: 0.2569, Recall: 0.1969, Precision: 0.3692
```

Figure 9 - The Experimental Result of the Parallel Computing Approach

Table 3: Performance Comparison with Different Process in Cloud Practice

Metric	Cloud (1 node – 1 process)	Cloud (1 node -2 process)
Training Time(seconds)	58.81	55.29
CPU Usage (%)	0.00	100.20
Memory Usage (MB)	504.55	506.70
AUC Score	0.8177	0.8177
Accuracy	0.8957	0.8957
Resource Efficiency (time/resource ratio)	0.1165	0.1087

Table 3 above compares the metrics with the cloud implementation with 1 node and 1 process. The result in Table 3 presents that by adding 1 more process for RF model training, the 100% CPU usage indicated that the requirement of computing power has a significant increment when demanding dealing with large-scale datasets or more complex algorithms. It reflects that cloud platforms can dynamically allocate more resources for computing tasks parallelly and the efficiency is relatively reduced by handling extra processes effectively without extra memory usage.

6.4 Case Study 3 The Cloud Environment with Multiple Processes (2 Nodes, 2 Processes)

In this study, 2 cloud nodes were allocated with 2 processes for each to introduce a more efficient distributed practice by parallelization. The comparison consists of different results across local environment to cloud-based environments to explore the benefits of effectively leveraging of scalability of cloud resources using distributed practices. The results are shown in Figure 10 as follows:

```
Training RF model...

Model RF resource consumption:
Memory used: 561.89 MB
CPU usage: 378.80%
Training time: 16.7575 seconds

Model RF performance metrics:
Accuracy=0.8957, AUC=0.8177
Class 0 - F1: 0.9439, Recall: 0.9661, Precision: 0.9227
Class 1 - F1: 0.2569, Recall: 0.1969, Precision: 0.3692
```

Figure 10 - The Experimental Result in the Configuration of 2 Node 2 Process

Table 4 below shows a significant outcome in model training time that was set up by 2 nodes 2 process setup in cloud practice, which reduced the training time from 67.47 seconds to 16.76 seconds. As shown in Figure 11, the cloud-based approach reduced the training time and memory usage by 75.2% and 69.4% compared to the local implementation. The resource efficiency was improved by 18.9% measured in the training time to the tracked memory usage. With the maintained accuracy result, the improvement in training time and the better resource consumption efficiency demonstrate that scalable cloud resources benefit the performance of larger size of model prediction effectively.

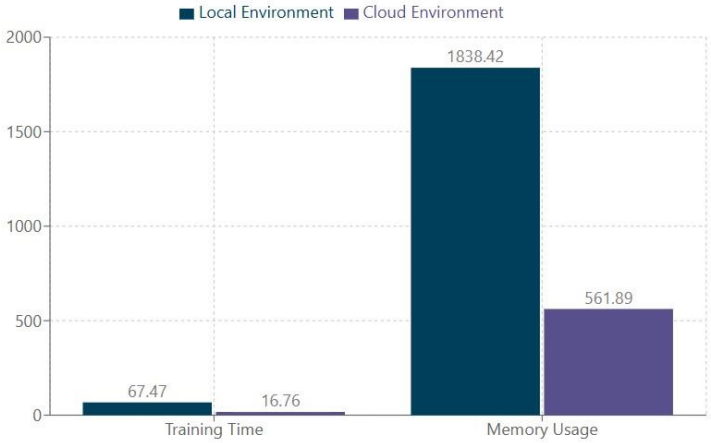


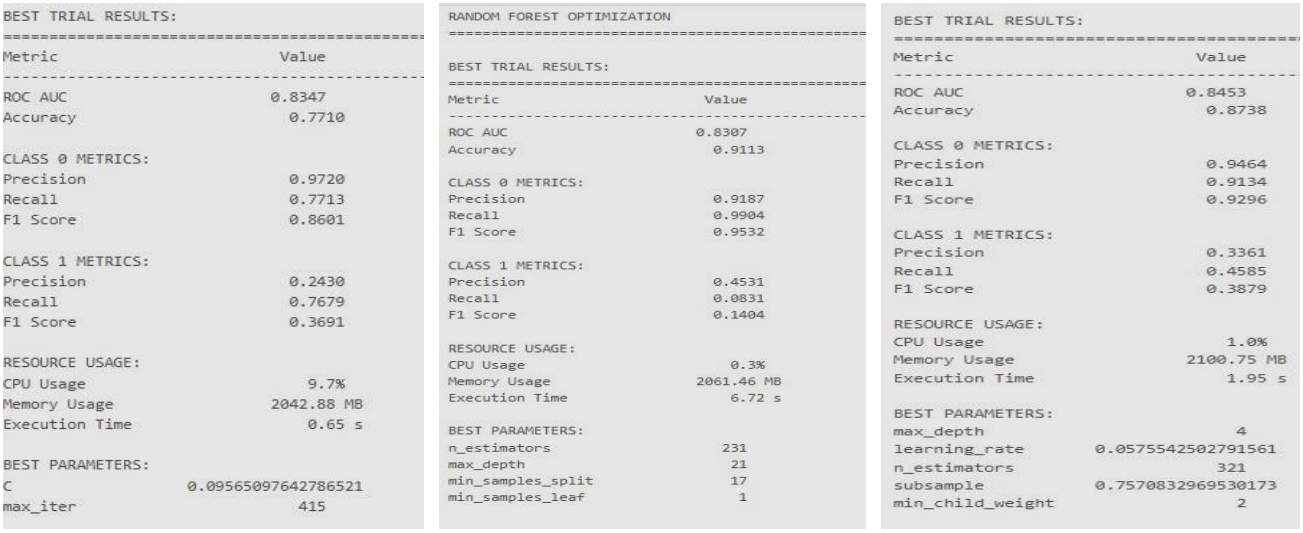
Figure 11 - Training Time and Memory Usage Comparison

Table 4: Performance Comparison between 2 Environments

Metric	Local (1 node – 1 process)	Cloud (2 node -2 process)
Training Time(seconds)	67.47	16.76
CPU Usage (%)	0.00	378.80
Memory Usage (MB)	1838.42	561.89
AUC Score	0.8157	0.8177
Accuracy	0. 8930	0.8957
Resource Efficiency (time/resource ratio)	0.0367	0.0298

6.5 Case Study 4: Hyperparameter Tuning with the Optuna in Both Environments

This case study gives a more extended introduction to gaining a better performance by auto hyperparameter tuning in both environments. Specifically, the cloud practice set up distributed training in a configuration with 4 nodes along with 1 process. It focused more on how cloud resources have an impact on the model performance during hyperparameter tuning. The Figure shows the result of model performance in the local environment after automatic hyperparameter tuning with a single process.



BEST TRIAL RESULTS:

Metric	Value
ROC AUC	0.8347
Accuracy	0.7710
CLASS 0 METRICS:	
Precision	0.9720
Recall	0.7713
F1 Score	0.8601
CLASS 1 METRICS:	
Precision	0.2430
Recall	0.7679
F1 Score	0.3691
RESOURCE USAGE:	
CPU Usage	9.7%
Memory Usage	2042.88 MB
Execution Time	0.65 s
BEST PARAMETERS:	
C	0.09565097642786521
max_iter	415

RANDOM FOREST OPTIMIZATION

BEST TRIAL RESULTS:

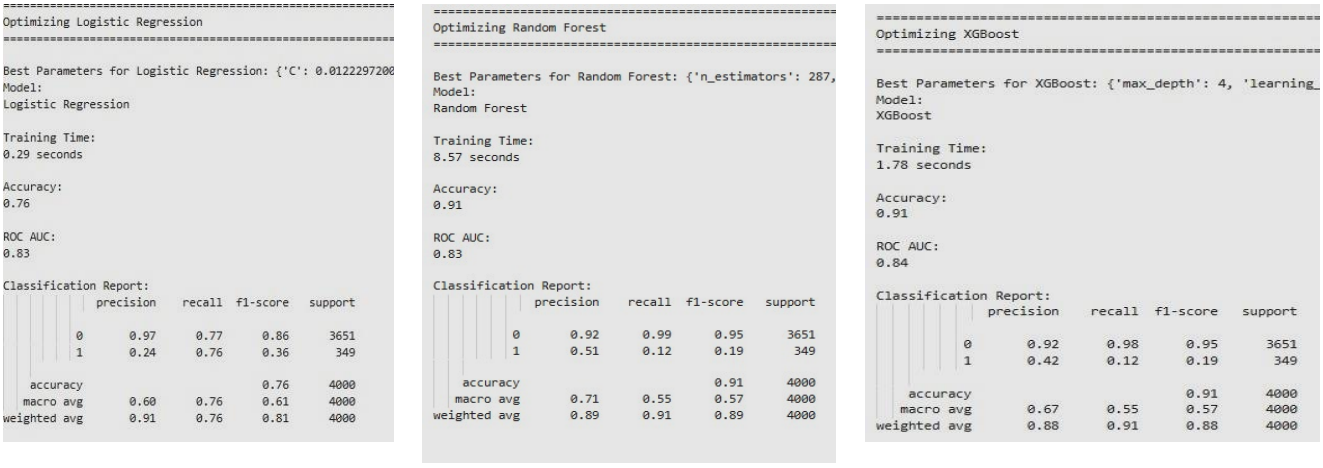
Metric	Value
ROC AUC	0.8307
Accuracy	0.9113
CLASS 0 METRICS:	
Precision	0.9187
Recall	0.9904
F1 Score	0.9532
CLASS 1 METRICS:	
Precision	0.4531
Recall	0.0831
F1 Score	0.1404
RESOURCE USAGE:	
CPU Usage	0.3%
Memory Usage	2061.46 MB
Execution Time	6.72 s
BEST PARAMETERS:	
n_estimators	231
max_depth	21
min_samples_split	17
min_samples_leaf	1

BEST TRIAL RESULTS:

Metric	Value
ROC AUC	0.8453
Accuracy	0.8738
CLASS 0 METRICS:	
Precision	0.9464
Recall	0.9134
F1 Score	0.9296
CLASS 1 METRICS:	
Precision	0.3361
Recall	0.4585
F1 Score	0.3879
RESOURCE USAGE:	
CPU Usage	1.0%
Memory Usage	2100.75 MB
Execution Time	1.95 s
BEST PARAMETERS:	
max_depth	4
learning_rate	0.0575542502791561
n_estimators	321
subsample	0.7570832969530173
min_child_weight	2

Figure 12 - Optimized Training Results in the Local Environment

Figure 12 and Figure 13 show the metrics were evaluated in the cloud environment that implemented distributed computing with automated hyperparameter tuning.



Optimizing Logistic Regression

Best Parameters for Logistic Regression: {'C': 0.0122972080
Model:
Logistic Regression

Training Time:
0.29 seconds

Accuracy:
0.76

ROC AUC:
0.83

Classification Report:

	precision	recall	f1-score	support
0	0.97	0.77	0.86	3651
1	0.24	0.76	0.36	349
accuracy			0.76	4000
macro avg	0.60	0.76	0.61	4000
weighted avg	0.91	0.76	0.81	4000

Optimizing Random Forest

Best Parameters for Random Forest: {'n_estimators': 287,
Model:
Random Forest

Training Time:
8.57 seconds

Accuracy:
0.91

ROC AUC:
0.83

Classification Report:

	precision	recall	f1-score	support
0	0.92	0.99	0.95	3651
1	0.51	0.12	0.19	349
accuracy			0.91	4000
macro avg	0.71	0.55	0.57	4000
weighted avg	0.89	0.91	0.89	4000

Optimizing XGBoost

Best Parameters for XGBoost: {'max_depth': 4, 'learning_
Model:
XGBoost

Training Time:
1.78 seconds

Accuracy:
0.91

ROC AUC:
0.84

Classification Report:

	precision	recall	f1-score	support
0	0.92	0.98	0.95	3651
1	0.42	0.12	0.19	349
accuracy			0.91	4000
macro avg	0.67	0.55	0.57	4000
weighted avg	0.88	0.91	0.88	4000

Figure 13 - Optimized Training Results with Distributing Configuration

The results show that the overall performance is consistent in both environments. In local practice, the accuracy of LR is 0.77 and the ROC AUC value is 0.8347, while the cloud implementation obtained a similar performance in terms of 0.76 and 0.83. The accuracies of the RF algorithm are maintained at 0.91 in both environments. Worthy notice, the accuracy of XGBoost training reached 0.91 similar to RF training. It indicates that complex algorithms can largely benefit from distributed implementation with cloud resources. Additionally, in the

parameter tuning process of RF training, 287 trees were searched in the distributed cloud environment while there are 231 trees in the local practice. It shows that the distributed computing provided by available cloud resources allows better optimization in parameter tuning that enables more reliable and efficient heart disease prediction in practice, especially when implementing automatic hyperparameter tuning in complex algorithms such as XGBoost.

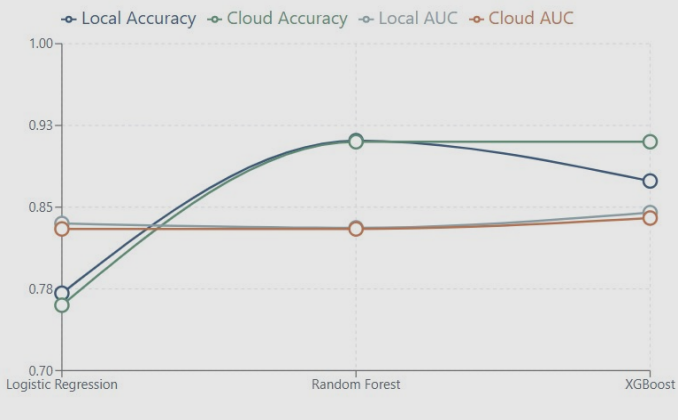


Figure 14 - Comparisons of Accuracy and AUC Score in Both Environments

Figure 14 illustrates an overall consistency in predictive performance for both environments, which shows that the modelings' integrity was maintained with identical algorithms and hyperparameter optimization while it demonstrates computational advantages.

Figure 15 - Comparison of Training Time in Both Environments

Figure 15 presents the training time comparison across the local environment to the cloud-based environment. This chart particularly shows the relationship between computing efficiency with algorithm complexity. Combined with the training time and the amount of decision trees in RF training, extended searching space was conducted via the cloud-based environment which represent a significant improvement by implementing RF modeling with distributed computing environments.

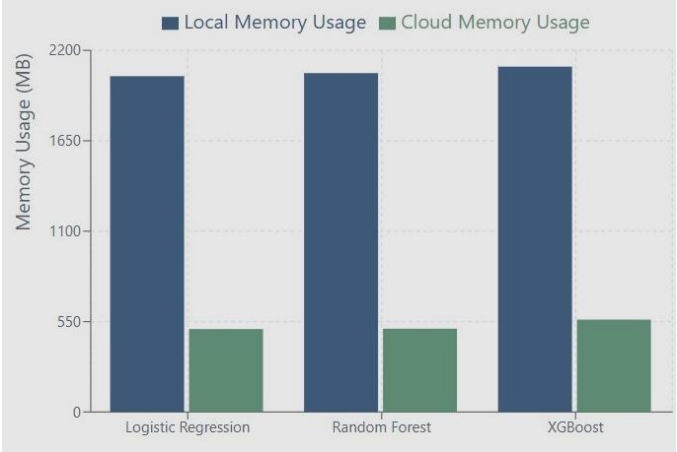


Figure 16 - Comparison of Resource Utilization in Both Environments

Figure 16 demonstrates that memory utilization was enhanced through the cloud-based implementation. The performance pattern of consistent memory consumption with overall maintained or improved accuracy offers quantified proof for transiting heart disease prediction into cloud-based environments. Specifically, the finding presents that the implementation of complex models faces computing constraints due to their restricted resources, which is a critical finding for healthcare institutions to obtain a valuable reference.

6.6 Discussion

Based on the experimental results across all case studies, this section discussed the findings in terms of model performance, resource utilization, and scalability in modeling and highlighted the potential computing advantages in cloud-based implementation for heart disease predictions.

6.6.1 Efficiency Improvement Through Cloud

By the implementation results across the development from 1 node to distributed computing that integrates multiple nodes and processes, restricted modeling scenarios can benefit from a scalable cloud platform as the more efficient resource utilized pattern. The training time comparison between default configurations in both local and cloud environments reduced the training time by 13%. Significantly, the training time was reduced in distributed computing which enables the training time to be shortened from 67.47 seconds in local practice to 16.76 seconds in cloud environment. This improvement was contributed by the support of scalable resources and efficient workload allocation with a parallel strategy from Azure. As expected, the experimental result proves that the adoption of cloud machine learning techniques can enhance the efficiency in heart disease prediction, while its excellent resource management can burden off the demand for costly hardware for healthcare institutions in restricted areas.

6.6.2 Model Performance Optimization Through Cloud

In the hyperparameter tuning experiments, a wider parameter searching space with 287 trees in cloud approaches and 231 trees in the local environment during RF experimental results shows the significant optimization performance by the integration of cloud resources. As the impact assessment for the complex algorithm, the improvement in ROC-AUC score during XGBoost training presents the computational advantages that are provided by the distributed computing adoption in Azure. These findings prove that the value of this study lies in efficiently implementing complex heart disease predictions in resource-limited healthcare

environments, while it provides improved optimization in modelings instead of requiring extra hardware resources in local environments.

6.6.3 Efficiency in Resource Management

Compared with 1838.42 MB consumed during the single node implementation in the local practice, the default configuration in cloud practice utilized 504.55 MB that presents a substantial reduction in resource consumption. It shows that the allocation system is more efficient in the cloud platform. In addition, the configuration in adding processes demonstrates productive parallelism ability through the cloud practice and has efficient CPU usage without extra usage of memory. More importantly, the optimal resource efficiency in the practice of 2 nodes and 2 processes highlights the potential benefits while conducting more complex machine learning in heart disease prediction using cloud computing practice to contribute a scalable approach for efficient resource management in healthcare institutions.

6.6.4 Cost-Benefit Analysis for Healthcare Institutions

This section analyses the economic impact of transitioning heart disease prediction from local environments to cloud-based implementation, while it also evaluates costs and long-term benefits by scaling the heart disease prediction tasks.

Several significant expenditures in the local implementation range from hardware investment to IT personnel. In European countries, 20,000 to 30,000 euros is required for enterprise-level servers and other infrastructure that serves as the initial hardware investment. Accordingly, 5,000 to 8,000 euros is used for annual system maintenance and hardware updates. Moreover, the cost of IT personnel requires 30,000 euros for the average annual cost, which is a significant portion of expenditure. On the other hand, the flexible cost with various scaling requirements for the cloud-based implementation can only require 8,000 to 12,000 euros of annual subscription cost for the usage of computing resources.

The comparison results across both environments, show three significant advantages in the cloud-based implementation. The reduction of 72.6% in memory consumption and 75.2% in training time shows a significant improvement in resource optimization during training. Compared to local training development, the cloud-based environment enables dynamic resource allocation and can save up to 30% to 40% cost of infrastructure. In addition, the extended parameter search function introduced by optimization frameworks can increase the amount of searching trees during RF training, which enhances operational efficiency.

A significant economic disparity is shown over a five-year cost analysis. 212,000 euros is required for local deployment including 60,000 euros for initial investment and 38,000 euros of annual operational cost. On the contrary, 160,000 euros is required for cloud deployment, while 32,000 euros is used for annual expenditure. As the comparative result is shown in Figure 17, a significant long-term cost-efficient advantage by adopting the cloud-based implementations.

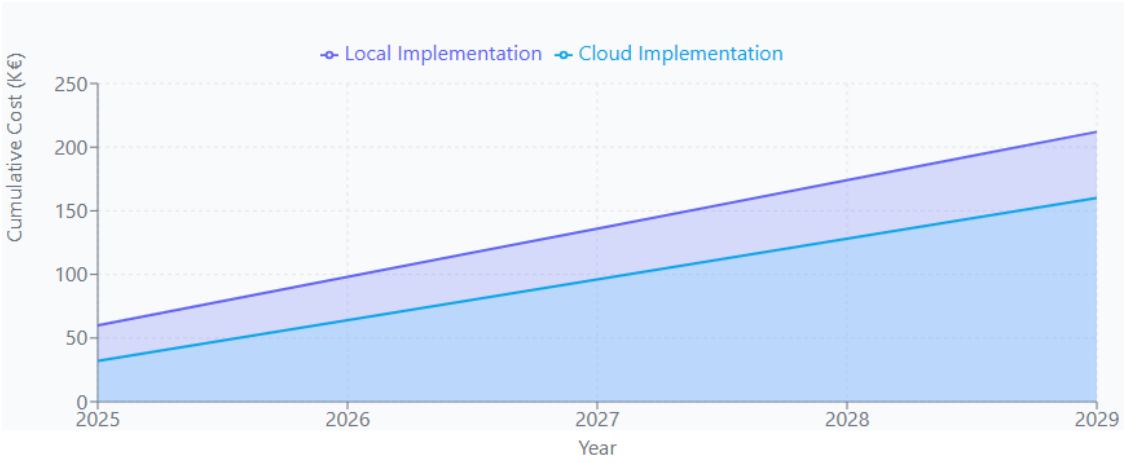


Figure 17 – The Cumulative Cost Over 5-Year Economic Analysis

7 Conclusion and Future Work

This research investigated the potential advantage in heart disease prediction using cloud computing, through comprehensive comparative experimental analysis on implementations across a local environment to the cloud-based environment. Models' performance and resource utilization were evaluated in aspects of distributed computing and automated hyperparameter tuning. The experimental results demonstrate significant improvements in computational efficiency in the cloud implementation, while the training time was reduced by 75.2% and the memory consumption was reduced by 72.6%. Cloud-based machine learning approach expands parameter search space to felicitate enhanced prediction performance in Random Forest modeling.

From the economic perspective, an investment return can be achieved within 18-24 months through cloud-based implementations, with a 5-year cumulative savings of 52,000 euros compared to local deployments. Overall, the cloud-based techniques offer a compelling combination of cost-efficient advantages and technological improvements for providing a convincing case for adopting cloud in heart disease prediction in resource-restricted medical institutions.

Considering the results, cloud machine learning techniques offer efficient solutions for addressing the computational resource limitations in healthcare applications. Especially, the enhancement of resource utilization demonstrates healthcare institutions can benefit from the cloud-based approach to obtain efficient heart disease prediction with increasing demand for computing power.

Inevitably, there were some inaccuracies due to the used datasets did not fully represent the complexity of features of the real world. Additionally, external factors such as Network latency might influence the evaluation in metrics, while the hardware configuration consistency was ensured for both environments.

Future work should explore further the scalability of cloud-based systems through larger size of datasets to validate the efficiency of heart disease prediction in the real world.

References

Adler, E.D., et al., 2020. Improving risk prediction in heart failure using machine learning. *European Journal of Heart Failure*, 22(1), pp.139-147.

Alaa, A.M., Bolton, T., Di Angelantonio, E., Rudd, J.H.F. and van der Schaar, M., 2019. Cardiovascular disease risk prediction using automated machine learning: a prospective study of 423,604 UK Biobank participants. *PloS One*, 14(5), p.e0213653.

Ambrish, G., et al., 2022. Logistic regression technique for prediction of cardiovascular disease. *Global Transitions Proceedings*, 3(1), pp.127-130.

Angeli, F., Verdecchia, P., Bittner, V., and Reboldi, G., 2012. Very early initiation of statin therapy and mortality in patients with acute coronary syndrome. *Acute Cardiac Care*, 14(1), pp.34-39.

Bays, H.E., Taub, P.R., Epstein, E., Michos, E.D., and Perak, A., 2021. Ten things to know about ten cardiovascular disease risk factors. *American Journal of Preventive Cardiology*, 5, p.100149.

Bhowmik, P.K., Zhang, L., Kumar, R., and Lee, J.H., 2024. Advancing heart disease prediction through machine learning: techniques and insights for improved cardiovascular health. *British Journal of Nursing Studies*, 4(2), pp.35-50.

Budholiya, K., Shrivastava, S.K., and Sharma, V., 2022. An optimized XGBoost based diagnostic system for effective prediction of heart disease. *Journal of King Saud University - Computer and Information Sciences*, 34(7), pp.4514-4523.

Celermajer, D.S., Chow, C.K., Marijon, E., Anstey, N.M., and Woo, K.S., 2012. Cardiovascular disease in the developing world: prevalences, patterns, and the potential of early disease detection. *Journal of the American College of Cardiology*, 60(14), pp.1207-1216.

Farida, M., Smith, J., Kumar, P., and Wang, L., 2022. Artificial intelligence-based methods for fusion of electronic health records and imaging data. *Scientific Reports*, 12(1), p.17981.

Gaidai, O., Cao, Y. and Loginov, S., 2023. Global cardiovascular diseases death rate prediction. *Current Problems in Cardiology*, 48(5), p.101622.

Hannun, A.Y., et al., 2019. Cardiologist-level arrhythmia detection and classification in ambulatory electrocardiograms using a deep neural network. *Nature Medicine*, 25(1), pp.65-69.

Lacey, G., Taylor, G.W. and Areibi, S., 2016. Deep learning on FPGAs: past, present, and future.

Liu, C., Zhang, H., Wang, J., and Li, Y., 2018. Signal quality assessment and lightweight QRS detection for wearable ECG SmartVest system. *IEEE Internet of Things Journal*, 6(2), pp.1363-1374.

Mesko, B., 2017. The role of artificial intelligence in precision medicine. *Expert Review of Precision Medicine and Drug Development*, 2(5), pp.239-241.

Mohan, S., Thirumalai, C. and Srivastava, G., 2019. Effective heart disease prediction using hybrid machine learning techniques. *IEEE Access*, 7, pp.81542-81554.

Peyret, R., et al., 2023. Multicenter automatic detection of invasive carcinoma on breast whole slide images. *PLOS Digital Health*, 2(2), e0000091.

Rajkomar, A., et al., 2018. Scalable and accurate deep learning with electronic health records. *NPJ Digital Medicine*, 1(1), pp.1-10.

Saikumar, K. and Rajesh, V., 2020. A novel implementation heart diagnosis system based on random forest machine learning technique. *International Journal of Pharmaceutical Research*, 12(1).

Yang, J. and Guan, J., 2022. A heart disease prediction model based on feature optimization and SMOTE-XGBoost algorithm. *Information*, 13(10), p.475.

Zhou, J., Zhiying, Z. and Li, L., 2021. A meta-analysis of Watson for Oncology in clinical application. *Scientific Reports*, 11(1), p.5792.