CLOUD RESOURCE OPTIMIZATION THROUGH VM WORKLOAD PREDICTION

Enroll. No. (s) - 20103214, 20103217, 20103228

Name of Students -Tanishq Sharma, Kunwar Ubaid Zafar Khan, Divyansh Garg

Name of Supervisor - Mr. Kashav Ajmera

May - 2023

Submitted in Partial Fulfillment of Degree of

Bachelor of Technology

In

Computer Science Engineering

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING & INFORMATION TECHNOLOGY

JAYPEE INSTITUTE OF INFORMATION TECHNOLOGY, NOIDA

TABLE OF CONTENTS

DECLARATION

We **Tanishq Sharma, Kunwar Ubaid Zafar Khan** and **Divyansh Garg** hereby declare that this submission is my/our own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person nor material which has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgment has been made in the text.

Place : Jaypee Institute of Information Technology, Noida

Date : 08/05/2023

CERTIFICATE

This is to certify that the work titled **“Cloud Resource Optimization through Virtual Machine Workload Prediction”**. submitted by **“Tanishq Sharma, Kunwar Ubaid Zafar Khan** and **Divyansh Garg''** is in partial fulfillment for the award of degree of B.Tech of Jaypee Institute of Information Technology, Noida has been carried out under my supervision. This work has not been submitted partially or wholly to any other University or Institute for the award of this or any other degree or diploma.

Ajmera

Signature of the Supervisor

Name of the Supervisor : Mr. Kashav Ajmera

Designation : Assistant Professor

Date : 08/05/2023

ACKNOWLEDGEMENT

The completion of any inter-disciplinary project depends upon cooperation, coordination, and combined efforts of several sources of knowledge. We are grateful to Mr. Kashav Ajmera for his willingness to give us valuable advice and direction whenever we approached him with a problem. We are thankful to him for providing us with immense guidance for this project. We would also like to thank our College authorities for allowing us to pursue our project in this field.

Date - 08/05/2023

Signature	Tanishq	Ubaid	Divyansh
Name	Tanishq Sharma	Kunwar Ubaid Zafar Khan	Divyansh Garg
Enrollment Number	20103214	20103217	20103228

SUMMARY

The project focuses on improving the efficiency of resource allocation in cloud platforms while reducing operational costs. The objective is achieved through analyzing the workload patterns of virtual machines and predicting their future demands using machine learning techniques. The datasets used in the project consist of CPU utilization of virtual machines at 5-minute intervals for three different dates.

To evaluate the performance of different regression techniques, the project compared the accuracy of eight different models using three evaluation metrics: Mean Squared Error (MSE), Mean Absolute Deviation (MAD), and R-score. This analysis helped identify which models can predict the virtual machines' future workload patterns accurately.

By accurately predicting the virtual machines' future workload patterns, the project aims to identify which virtual machines are likely to face overloading or underloading conditions. This information can help optimize the resource allocation in cloud platforms, prevent physical resource exhaustion, and improve the overall performance and cost-effectiveness of the system.

The project's ultimate goal is to improve cloud resource allocation efficiency by predicting the future workload patterns of virtual machines and preventing overloading and underloading conditions, leading to optimal performance and cost-effectiveness.

CHAPTER - 01 : INTRODUCTION

1.1 General Introduction

Cloud computing has become a vital infrastructure demand in modern organizations for many reasons, including cost-effectiveness, scalability, and security. However, cloud computing also addresses two crucial aspects of the green IT approach: energy efficiency and resource efficiency. From an energy efficiency perspective, cloud computing leverages server virtualization to minimize the total physical server footprint, resulting in less electricity consumed.

Cloud computing also saves resources because less equipment is required to run the workloads, and organizations can proactively reduce data center space and eventual e-waste footprint. Green cloud computing is simply an approach where companies can use what they already have smartly to minimize energy consumption and overall carbon footprint.

Green cloud computing is a buzzword in the IT industry that refers to the potential environmental benefits cloud-based services can offer society. The term combines two words: green, which means environmentally friendly, and cloud computing, which is the delivery of IT services over the internet.

Green cloud computing has three main goals: to maximize energy efficiencies during the device's life cycle, promote the use of recyclable materials, and minimize the use of hazardous IT components. You can observe green cloud computing from two perspectives:

- **Green hardware [1] -** This includes energy-efficient and environmentally friendly information and communications technology (ICT) tools such as servers, network appliances, and storage devices used in data centers. It also comprises the power supply units, the cooling equipment, and the building that houses these components.

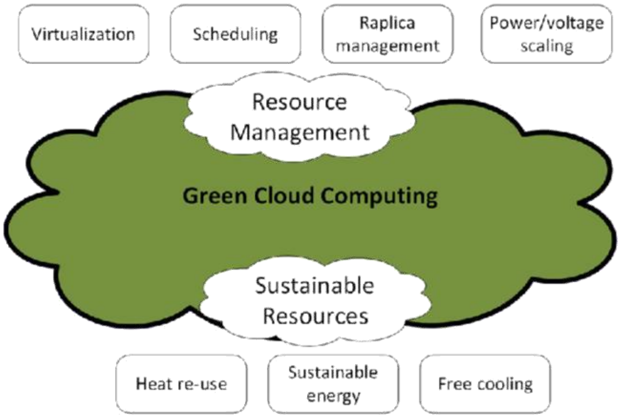
- **Green software engineering methodologies [2] -** This includes all the applications that manage data centers and other cloud-based services. The main idea behind green software engineering methodologies is to build reliable applications that not only meet organizations' requirements but are also energy efficient.

Virtually all CSPs are going green and adopting net-zero practices that have the sole objective of minimizing environmental damage caused by GHG emissions. However, not all cloud practices are energy efficient. You can use these three strategies to migrate to a sustainable green cloud:

- **<u>Virtualization</u> [3]- .** Virtualization is the solution that resolves the problem of enormous electricity consumption by on-premises data centers. For example, an organization can leverage server virtualization to run multiple virtual machines (VMs) on the same physical server. This essentially partitions the physical host into numerous virtual servers, allowing the organization to achieve significant cost savings.An organization can also use desktop virtualization solutions such as Parallels Remote Application Server (RAS) to deliver virtual applications and desktops to employees in remote locations. Employees can, in turn, use low-end devices that don’t consume much power, such as thin clients, to access these resources from any location.

- **<u>Cloud optimization tools [2]</u> -** Many cloud optimizations that reduce server utilization rates also minimize carbon emissions. However, this may not work in some scenarios, especially those agreements where the company pays in bulk or gets discounted services. To reduce energy consumption in such a scenario, you’ll need to have complete visibility of the entire IT infrastructure to optimize the cloud setup.

- **<u>Carbon-aware CSPs [2]</u> -** When selecting a cloud service vendor, it’s always best to go with one that is carbon-aware rather than those that run on nonrenewable energy. The Green Web Foundation (GWF) has a database consisting of all the carbon-aware CSPs that you can select from if you want to transition to the cloud.



[1] <u>Different techniques in which green computing can be achieved</u>

1.2 Problem Statement

Cloud computing platforms face the challenge of allocating resources efficiently to meet the dynamic demands of their users while avoiding under-utilization or over-utilization of physical resources. To address this challenge, we present a study that analyzes the workload patterns of three cloud computing workloads and compares the accuracy of eight different regression techniques in predicting future CPU utilization of virtual machines. Our results show that Random forest Regression model and Robust Regression model outperforms the other models in terms of mean squared error, mean absolute deviation, and R-squared score. By leveraging the accuracy of our predictions, we propose a VM selection policy that optimizes resource allocation and reduces operational costs. By predicting the cpu-utilizations before helps us in selecting better virtual machine placement policy. Our study demonstrates the value of data analysis and machine learning techniques in improving the performance and efficiency of cloud computing platforms.

1.3 Significance/Novelty of the Problem

Efficient resource allocation is a crucial challenge in cloud computing platforms. The study addresses this challenge by analyzing workload patterns and predicting future demands. The study uses machine learning techniques to predict future CPU utilization of virtual machines. Comparing the accuracy of eight different regression techniques is a novel approach that adds value to the field.

The study proposes a VM selection policy is based on the accuracy of predictions. This policy can optimize resource allocation and reduce operational costs, which is a significant contribution to the field.

The study demonstrates the value of data analysis and machine learning techniques in improving the performance and efficiency of cloud computing platforms. The results can help in developing better virtual machine placement policies, which is a critical area of research in cloud computing.

1.4 Comparison of existing approaches to the problem framed

To compare our approach with existing approaches, we can look at previous studies that have addressed the problem of resource allocation in cloud computing platforms. Many previous studies have used statistical and machine learning techniques to predict resource utilization, such as linear regression, time series analysis, and neural networks.

However, most of these studies have only considered a single regression technique or a small set of techniques, and they have not compared the accuracy of multiple techniques. In contrast, our study compares the accuracy of eight different regression techniques in predicting future CPU utilization of virtual machines.

Moreover, our study focuses on analyzing the workload patterns of multiple cloud computing workloads, whereas many previous studies have only analyzed a single workload. By comparing multiple regression techniques and analyzing multiple workloads, we aim to provide a more comprehensive understanding of the performance and limitations of different prediction models. In contrast, Siddharth Kulkarni et al. proposed a framework for predicting virtual machine workload using time series analysis and machine learning techniques, but they only evaluated their approach on a single workload. Similarly, Arif Ahmed and Narges Shahidi presented a machine learning-based approach for predicting cloud application resource scaling needs, but they did not compare their approach with other regression techniques. We have provided the literature survey of the research works of existing research work done by the same.

Overall, our study contributes to the existing literature by providing a more thorough and systematic analysis of the accuracy of different regression techniques in predicting resource utilization in cloud computing platforms.

CHAPTER - 02 : LITERATURE SURVEY

It's only been possible because of the below listed research papers that we gathered the vital information required for completing the project.

Apart from the below listed research papers, we used to refer to the internet for our various problems. Most importantly, our supervisor provided us vital information regarding the project and this field whenever it was necessary for us.

2.1 Summary Of the Research Paper Studied

Research paper title	**Aims and Objective**	**Reseach questions asked in the paper**	**Technique / Method used and why ?**	**Interpretation of results**	**Good points**	**Remark/ your proposal if any**
1.Resource Central: Understanding and Predicting Workloads for Improved Resource Management in Large Cloud Platforms[5]	The aim of the paper is to improve resource management in large cloud platforms by accurately predicting the resource requirements of cloud applications and dynamically allocating resources to them.	1. How can we accurately predict the resource requirements of cloud applications to optimize resource management in large cloud platforms? 2.How can we set the predicted resource requirements to dynamically allocate resources to cloud applications to improve resource utilization	They use machine learning techniques such as regression analysis and neural networks to develop predictive models based on the identified patterns. The paper uses a combination of statistical analysis and machine learning techniques to develop a predictive resource management approach that can improve resource utilization and application	1. The proposed approach achieved 18% higher resource utilization and 12% lower cost compared to static allocation, and 16% higher resource utilization and 10% lower cost compared to proportional allocation. 2. The proposed approach also improved application performance by up to 30% compared to the baseline approaches, as measured by response time and throughput.	1. Comprehensive Approach 2. Real-world evaluation 3. High Prediction accuracy 4. Dynamic resource allocation	1. To evaluate the proposed approach on a larger and more diverse dataset. 2. To investigate the impact of different machine learning models on the performance of the proposed approach.

		and performance?	performance in large cloud platforms.			
2.Virtual Machine Workload Prediction using Time Series Analysis and Machine Learning[6]	To develop a hybrid approach that combines time series analysis and machine learning techniques for predicting the workload of virtual machines in cloud computing environments.	1. Can a hybrid approach that combines time series analysis and machine learning techniques improve the accuracy of workload prediction for virtual machines in cloud computing environments? 2. What are the appropriate preprocessing techniques and feature extraction methods that can be used to prepare workload data for analysis?	1. Data Preprocessing: The raw workload data is cleaned and transformed to prepare it for analysis. This involves removing outliers, smoothing the data, and converting it into a suitable format for analysis. 2. Workload Prediction: The workload prediction model is trained using a combination of time series analysis and machine learning techniques, such as ARIMA, k-NN, and decision trees. The model is used to make predictions about the future workload of virtual machines based on the	The authors report an improvement in prediction accuracy of up to 15% compared to the baseline methods.	1. Novel Approach 2. Comprehensive Evaluation 3. Significant Improvement in Prediction Accuracy 4. Applicability to Real-World Scenarios 5. Contribution to the field	Real-time prediction, Multi-cloud prediction and Energy aware prediction

			historical workload data and other relevant factors.			

			mapping with best possible energy efficiency. 3. Develop an efficient simulation engine and integrate into the cloud simulator CloudSim ,which can drastically reduce the evaluation time of VM allocation solutions in each evolutionary iteration.			

3. Machine Learning-based Predictive Resource Scaling for Cloud Applications [7]	The aim of this research paper is to propose a novel approach to resource scaling in cloud computing environments that utilizes machine learning techniques to predict future resource requirements and optimize resource utilization.	1. Can machine learning techniques be used to predict future resource requirements of cloud applications with high accuracy? 2. Can the proposed predictive resource scaling algorithm optimize resource utilization and improve application performance compared to traditional scaling approaches?	1. Data Collection: The authors collect data on cloud application workloads, which includes performance metrics such as CPU utilization, memory utilization, and network I/O. 2. Machine Learning Algorithm: The authors propose a predictive resource scaling algorithm that utilizes machine learning techniques to predict future resource requirements based on the input features. They use a regression model to predict future resource demands, which are then used to scale resources up or down.	1. The results of the evaluation show that the proposed approach outperforms traditional scaling approaches in terms of resource utilization and application performance. 2. The results suggest that the use of machine learning techniques to predict future resource requirements and optimize resource utilization can lead to more efficient and responsive cloud computing environments.	1. Improves resource utilization 2. Improves application performance 3. Real-world Implementation 4. Reproducibility	1. Further evaluation 2. Comparison with other approaches 3. Integration with cloud management systems

4.Energy-efficient and quality-aware VM consolidation method[8]	To improve resource utilization and energy efficiency, cloud data centers use virtual machine (VM) consolidation to consolidate VMs to fewer physical machines (PMs) through live VM migration. However, improper VM placement may cause frequent VM migrations and constant on–off switching of PMs, which results in lower service quality and increased energy consumption.	Problem of improper VM placement during VM consolidation which further results in frequent live VM migration and constant on–off switching of PMs, leading in turn to serious quality of service (QoS) degradation and resource overhead.	Proposing an effective and efficient VM consolidation approach called EQ-VMC, which has the goal of optimizing energy efficiency and service quality. A discrete differential evolution algorithm is developed to search for the global solution with a set of algorithms proposed for optimum solution for VM placement. By integrating effective host overload detection, VM selection, detection, EQ-VMC effectively and under-loaded host reduces energy consumption and improves quality of service (QoS). .	The results show that the proposed method outperforms existing consolidation methods in terms of energy consumption and QoS satisfaction.	1.This study addresses VM consolidation with respect to heuristic evolutionary algorithms 2. Minimized the mathematical expectation of the energy consumption of running PMs while maintaining the lowest probable risk of host overloading 3.An improved discrete differential evolution (discrete-DE) algorithm is developed by finding the result in the search space which finds the optimum VM placement for the migrated VMs. 4. A hybrid heuristic evolutionary-based EQ-VMC method is developed for VM consolidation.	Other optimization issues, such as minimizing VMMs and maximizing resource utilization during VM consolidation, need further study to improve the presented algorithm. the proposed optimization model should be given priority to adapt to GoogleCluster-like traces.

5.Resource-aware virtual machine placement algorithm for IaaS cloud[9]	The main aim of this paper is to develop a new resource-aware VM placement algorithm that considers both CPU and memory resources.	1.How can the allocation of virtual machines to physical hosts be optimized to improve resource utilization and minimize wastage? 2.How effective is the proposed resource-aware VM placement algorithm in comparison to existing baseline algorithms?	The paper proposes a new resource-aware VM placement algorithm that combines both static and dynamic allocation techniques. The static allocation is performed based on the initial resource requirements of the VMs, while the dynamic allocation is performed at runtime to handle any changes in the resource requirements of the VMs.	The proposed algorithm achieves higher energy efficiency compared to the baseline algorithms. This indicates that the algorithm is able to allocate VMs to physical hosts in a way that minimizes energy consumption while still meeting the resource requirements of the VMs. This higher energy efficiency can also lead to cost savings for cloud providers, as it allows them to reduce their energy bills and lower their carbon footprint.	1. The paper addresses an important problem in cloud computing - how to efficiently allocate virtual machines to physical hosts in an IaaS cloud environment. 2. The proposed algorithm takes into account multiple factors, such as CPU, memory, and network utilization, to make informed decisions about VM placement.	1.Experimental evaluation 2.Multi-objective optimization 3.Scalability and robustness 4.Hybrid placement strategies

So in order to complete our project, we need to work on all the points listed above and implement a method in which all the above conditions are met and our main motive of optimal energy utilization and better computer efficiency is fulfilled.

2.2 Definitions of some important terms:

2.2.1 Mean Square Error (MSE) [17]:-Mean square error (MSE) is a metric used to measure the average squared difference between the estimated values and the true values in a regression or prediction problem. It is a common evaluation metric in machine learning and statistics, particularly in the context of regression analysis. The MSE measures the average of the squares of the differences between the predicted and true values. It is a measure of the quality of the prediction model, where a lower MSE value indicates a better fit between the predicted and true values.

2.2.2 Median Absolute Deviation (MAD) [17]:- Mean absolute deviation (MAD) is a statistical metric that measures the average absolute difference between each value in a dataset and the mean of that dataset. It is used to quantify the amount of variability or dispersion in the data. The MAD is calculated by taking the absolute value of the difference between each data point and the mean, summing these absolute differences, and then dividing by the number of data points.

2.2.3 R-Square Score [17]:- R-squared, also known as the coefficient of determination, is a statistical metric used to measure the goodness of fit of a regression model. It is a value between 0 and 1, with a higher value indicating a better fit between the model and the data. R-squared measures the proportion of the variance in the dependent variable that is explained by the independent variables in the regression model. A value of 1 indicates that the model explains all of the variance in the dependent variable, while a value of 0 indicates that the model does not explain any of the variance.

Chapter - 03 : Requirement Analysis and Solution Approach

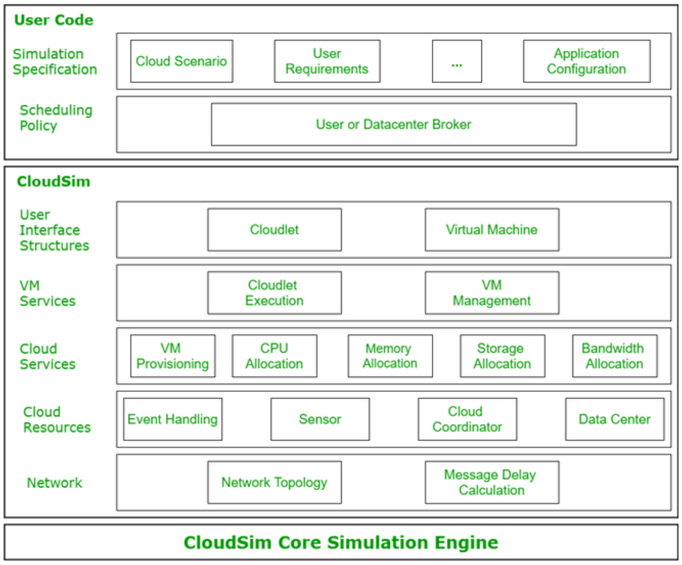
3.1 Requirement Analysis

HARDWARE REQUIREMENTS -

- Ideally for this project, we needed a lot of hardware which includes a network of datacentres, 24 hour running electricity, educated and skilled engineers and other working staff as well and most importantly we would have required monetary funds for our project. But all of these are not within our reach so we decided to implement this project on our electronic devices with the following below listed requirements.
- Intel i3 10 gen
- 4 GB RAM
- 5 GB GB hard free drive space
- An operating system (Most preferably Windows)

SOFTWARE REQUIREMENTS -

- Cloudsim - Cloudsim is an open source framework, , which is used to simulate cloud computing infrastructure and services. It is developed by the CLOUDS Lab organization and is written entirely in Java. It is used for modeling and simulating a cloud computing environment as a means for evaluating a hypothesis prior to software development in order to reproduce tests and results.



[12] Basic Architecture of Cloudsim Simulator

- Eclipse - The Eclipse IDE is famous for the Java Integrated Development Environment (IDE), but it has a number of pretty cool IDEs, including the C/C++ IDE, JavaScript/TypeScript IDE, PHP IDE, and more. In this article, we are going to explain how to install Eclipse IDE For Enterprise Java and Web Development. Eclipse provides a tool for developers to work with Java and Web applications, including a Java IDE, tools for JavaScript, TypeScript, JavaServer Pages and Faces, Yaml, Markdown, Web Services, JPA and Data Tools, Maven and Gradle, Git, and many more.

- Common Math Jar File - Commons Math is a library of lightweight, self-contained mathematics and statistics components addressing the most common problems not available in the Java programming language or Commons Lang.

- Java - Java is often referred to as WORA – Write Once and Run Anywhere, making it perfect for decentralized cloud-based applications. Cloud providers choose Java language to run programs on a wide range of underlying platfiorms.

- Jupyter Notebook - Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations, and narrative text. It is a popular tool for data science, machine learning, scientific computing, and research, as it allows users to perform exploratory data analysis, develop and test models, and communicate findings in an interactive and collaborative environment. Jupyter Notebook supports several programming languages, including Python, R, Julia, and others, and allows you to write and execute code directly in the web browser.

- Python - Python is a popular language for machine learning and data science due to its large community, vast ecosystem of libraries, and ease of use. Its flexibility, scalability, and interoperability make it a versatile language that can be used for a wide range of tasks, and its open-source nature has led to the development of many powerful libraries and tools.

- Python Libraries - Python has a large number of powerful and versatile libraries that can be used for a variety of tasks. We have used the following libraries in our project such as Pandas, Numpy, Matplotlib, Seaborn, Scikit Learn, Tensorflow, Keras and Stasmodel.

Here, in our project we are dealing with VM selection, since we don't have the financial assistance or physical data centres the cloudsim software and Jupyter notebook is very important for us. It has been possible only because of the above mentioned requirements that we are able to implement and learn from our project and in the future we plan to extend our implementation in a real cloud center where apart from time shown in the simulation many other time delays will be there.

We will be discussing all the shortcomings and failures of our project in the conclusion part given below.

3.2 Solution Approach

Solving any problem related to Virtual machine optimization is usually tackled through the following stages:-

Host Overloading Detection

The following techniques are used for host overloading detection worldwide:-

- **Median Absolute Deviation (MAD) [11]** - The MAD is a robust statistic, being more resilient to outliers in a data set than the standard deviation. In the standard deviation, the distances from the mean are squared, so on average, large deviations are weighted more heavily, and thus outliers can heavily influence it. In the MAD, the magnitude of the distances of a small number of outliers is irrelevant.

- **Interquartile Range (IQR) [12]** - The interquartile range (IQR), also called the midspread or middle fifty, is a measure of statistical dispersion, being equal to the difference between the third and first quartiles.

- **Local Regression (LR) [13]**- The main idea of the method of local regression is fitting simple models to localized subsets of data to build up a curve that approximates the original data.

- **Robust Local Regression (LRR) [14]**- Addition of the robust estimation method bisquare to the least-squares method for fitting a parametric family.This modification transforms Loess into an iterative method. The initial fit is carried out with weights defined using the tricube weight function.

VM Selection

The following techniques are used for VM selection worldwide:-

- **Linear Regression[15]-** Linear regression is a statistical technique used to model the relationship between a dependent variable and one or more independent variables, often referred to as predictors or features. The goal of linear regression is to find the best linear relationship between the dependent variable and the predictors, such that the difference between the predicted values and the actual values is minimized.

- **Huber Regression[16]** - Huber regression is a modified form of linear regression that is less sensitive to outliers and can be more robust to data that does not conform to a normal distribution. It uses a combination of the least squares and absolute deviation loss functions to balance the influence of outliers and provide more stable estimates.
- **Robust Regression[17]** - Robust regression is a family of techniques that are designed to be less sensitive to outliers and other forms of noise in the data. These methods often use robust loss functions, such as the Huber loss or the Tukey loss, to downweight or ignore data points that are far from the mean.
- **Random Forest Regression[18]** - Random forest regression is a machine learning method that uses an ensemble of decision trees to make predictions. It works by training multiple decision trees on different subsets of the data, and then combining their predictions through a voting mechanism to reduce variance and improve accuracy.
- **Gradient Boosting Trees[19]** - Gradient boosting trees is another machine learning method that uses an ensemble of decision trees, but instead of training them independently, it trains them sequentially, with each new tree focusing on the errors made by the previous ones. This allows it to gradually improve its predictions by learning from its mistakes.
- **Neural Network Regression[20]** - Neural network regression is a machine learning method that uses artificial neural networks to make predictions. It works by training a network of interconnected nodes, or neurons, on a set of input/output pairs, and then using the trained network to make predictions on new inputs.
- **SARIMAX[21]** - It is a time series forecasting model that incorporates both autoregressive and moving average components, as well as seasonality and exogenous variables. It is a popular method for forecasting time series data that exhibits trend, seasonality, and other forms of temporal dependence.
- **ARIMA[21]** - ARIMA stands for AutoRegressive Integrated Moving Average. It is a popular time series analysis method used to model time series data, forecast future values, and understand the underlying patterns in the data. ARIMA models have three main components: autoregression (AR), differencing (I), and moving average (MA).

VM Placement

The VM placement can be seen as a bin packing problem with variable bin sizes and prices, are the available CPU capacities of the nodes; and prices correspond to the power consumption by the nodes. As the bin packing problem is NP-hard, to solve it we apply a modification of the Best Fit Decreasing (BFD) algorithm. In the modification of the BFD algorithm denoted Power Aware Best Fit Decreasing (PABFD), we sort all the VMs in the decreasing order of their current CPU utilizations and allocate each VM to a host that provides the least increase of the power consumption caused by the allocation.

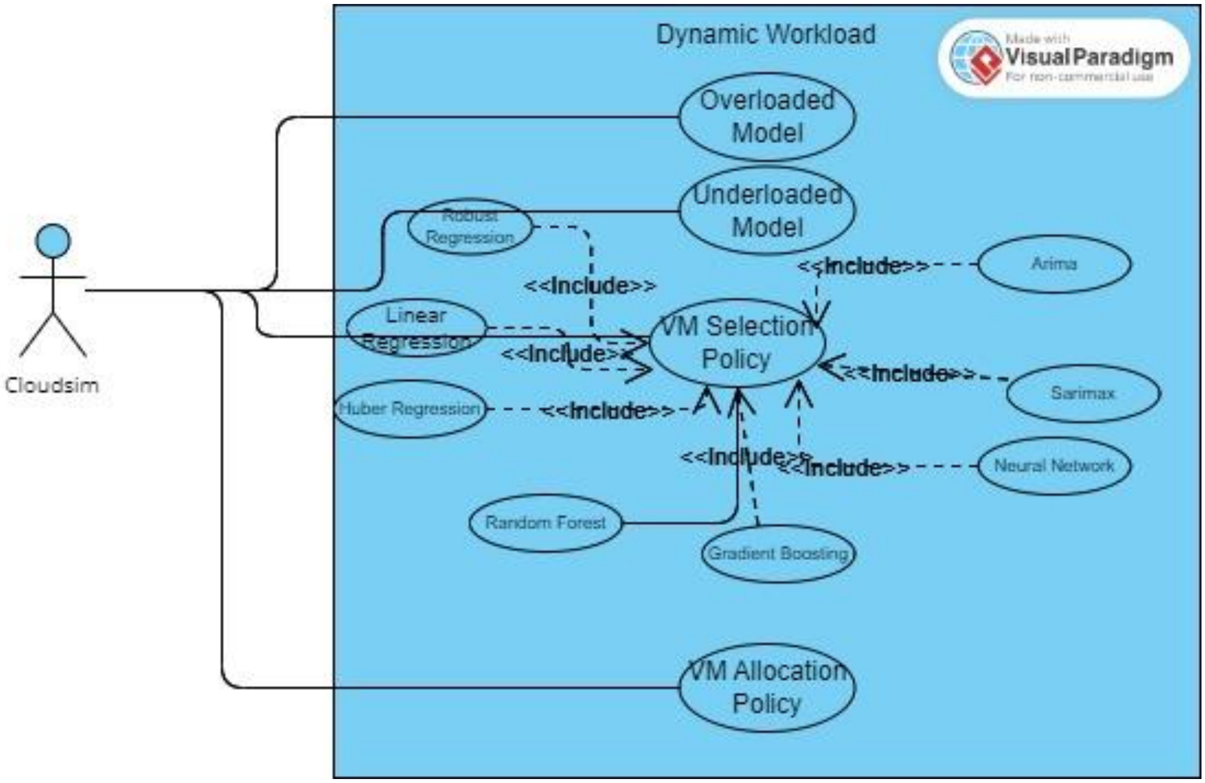
Host Underloading Detection

All the overloaded hosts are found using the selected overloading detection algorithm, and the VMs selected for migration are allocated to the destination hosts. Then, the system finds the host with the minimum utilization compared to the other hosts, and tries to place the VMs from this host on other hosts keeping them not overloaded. If this can be accomplished, the VMs are set for migration to the determined target hosts, and the source host is switched to the sleep mode once all the migrations have been completed. If all the VMs from the source host cannot be placed on other hosts, the host is kept active. This process is iteratively repeated for all hosts that have not been considered as being overloaded.

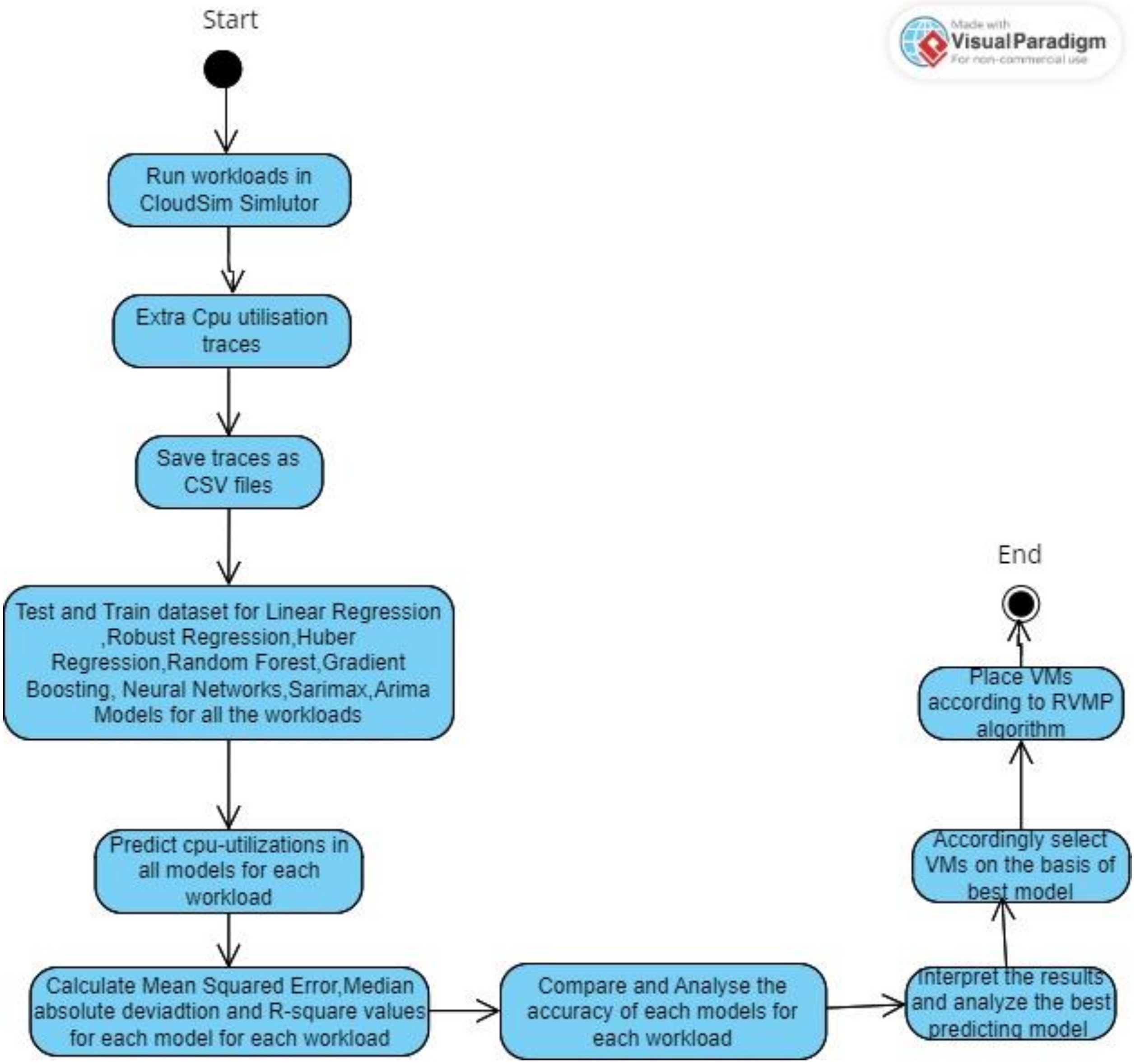
Chapter - 04 : Modeling and Implementation

Details 4.1 Design Diagram

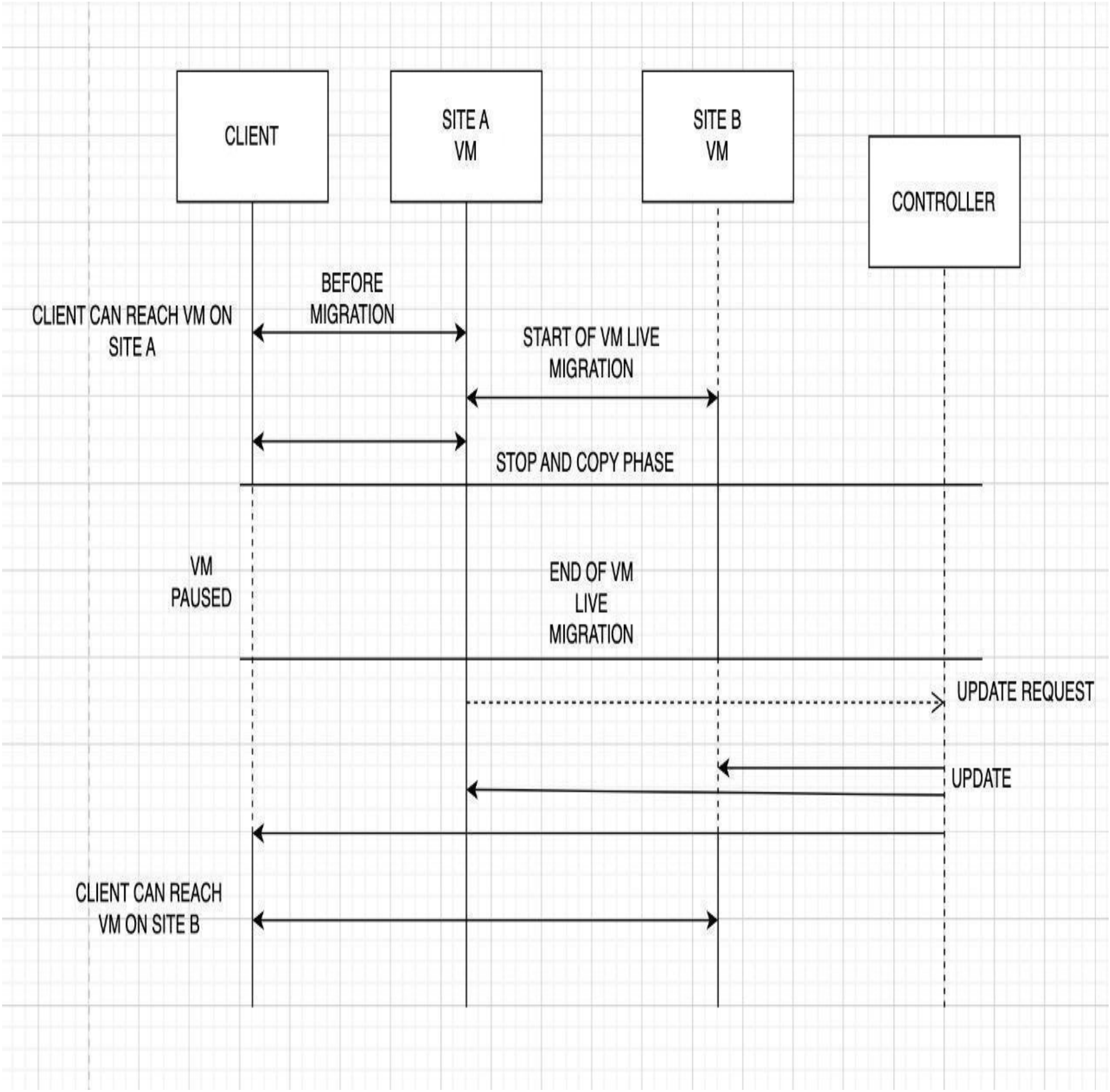
4.1.1 Use Case Diagram



4.1.2 Flow Diagram


Start
Made with Visual Paradigm For non-commercial use
Run workloads in CloudSim Simlutor
Extra Cpu utilisation traces
Save traces as CSV files
Test and Train dataset for Linear Regression ,Robust Regression,Huber Regression,Random Forest,Gradient Boosting, Neural Networks,Sarimax,Arima Models for all the workloads
Predict cpu-utilizations in all models for each workload
Calculate Mean Squared Error,Median absolute deviadtion and R-square values for each model for each workload
Compare and Analyse the accuracy of each models for each workload
Interpret the results and analyze the best predicting model
Accordingly select VMs on the basis of best model
Place VMs according to RVMP algorithm
End

4.1.3 Sequence Diagram



4.2 Implementation Details & Code Snippets

ML can improve resource management in cloud platforms. Understanding cloud workload is key for identifying improvements the prediction models produce high quality workload predictions. Predictions enable lower costs while retaining good Quality of Service (QoS).
The proposed methodology is a well-put-together system for generating, storing, and efficiently using the prediction of these characteristics. And can make use of more machine learning algorithms to efficient prediction serving systems. The three workloads 20110322, 20110325, 20110403 are extracted from the cloudsim and the traces of cpu utilizations are stored as csv files . The traces are obtained by running the virtual machines in the cloudsim environment. Basically the csv files contains the cpu-utilizations of the virtual machines. So , now the structure of the dataset is that there are 288 rows which basically tells the 5 minutes interval of a day and the columns are the Virtual machines . cpu_data.csv has 1516 columns i.e. 1516 virtual machines , cpu_data2.csv has 1078 Virtual machines , cpu_data3.csv has 1463 virtual machines. Now we have compared 8 different regression techinques that are linear regression , huber regression , robust regression , random forest , gradient tree, neural network , sarimax ,arima models.

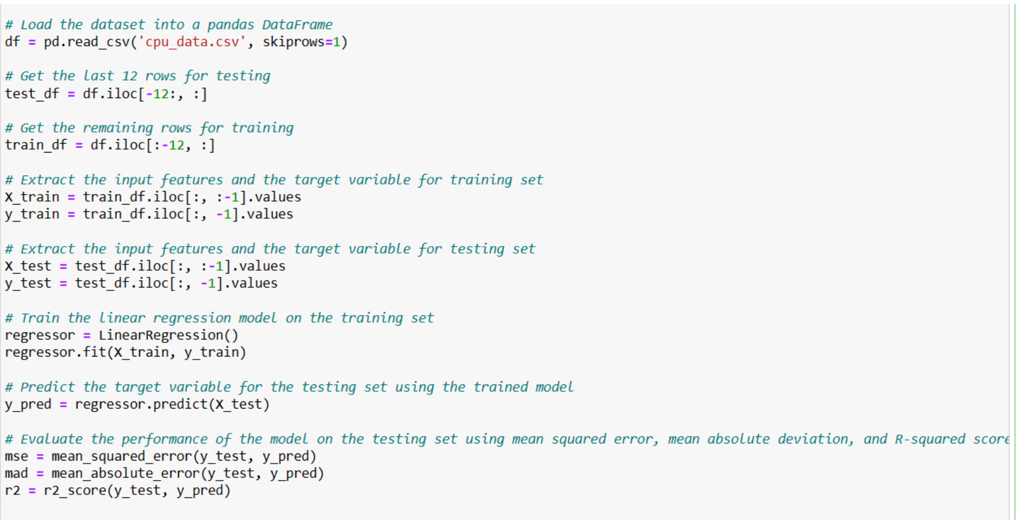
In these techniques we have splitted the 288 rows as 276 rows for training and 12 rows for testing so we split the data in the ratio of 95:5 for training and testing.
Note that the testing is done only on the last 12 rows of the dataset. **Also, the code only predicts the cpu-utilization of one virtual machine, which is the last column of the dataset i.e. here the target variable is the last virtual machine**. As by comparing the predicted values with the actual values its easier to compare the accuracy of the models in terms of Mean Square Error (MSE) , Median Absolute Deviation (MAD) and R-Square values. Similar analysis done for the other virtual machines to , display of those characteristics is chaotic , though their results are also considered.

Once we determine that which techniques are the best techniques for predicting CPU utilizations, we can use these models to help inform VM selection policies. As, we can use the output from these models to identify which VMs are likely to experience overloading or underloading conditions. Based on this information, we can adjust the VM selection policy to allocate more resources to VMs that are likely to experience overloading conditions, and fewer resources to VMs that are likely to experience underloading conditions. This can help to optimize resource allocation and reduce the operational costs of cloud platforms.

Another way to connect the models with VM selection policies is by integrating the predictions into the VM selection process itself. For instance, we can use the output from the models to rank VMs based on their predicted CPU utilizations and prioritize VMs accordingly. This approach can help to ensure that resources are allocated to the VMs that need them most, improving the overall performance of the cloud platform

The implementation details of each prediction model is as follows:

1.Linear Regression:



```
# Load the dataset into a pandas DataFrame
df = pd.read_csv('cpu_data.csv', skiprows=1)

# Get the last 12 rows for testing
test_df = df.iloc[-12:, :]

# Get the remaining rows for training
train_df = df.iloc[:-12, :]

# Extract the input features and the target variable for training set
X_train = train_df.iloc[:, :-1].values
y_train = train_df.iloc[:, -1].values

# Extract the input features and the target variable for testing set
X_test = test_df.iloc[:, :-1].values
y_test = test_df.iloc[:, -1].values

# Train the linear regression model on the training set
regressor = LinearRegression()
regressor.fit(X_train, y_train)

# Predict the target variable for the testing set using the trained model
y_pred = regressor.predict(X_test)

# Evaluate the performance of the model on the testing set using mean squared error, mean absolute deviation, and R-squared score
mse = mean_squared_error(y_test, y_pred)
mad = mean_absolute_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
```

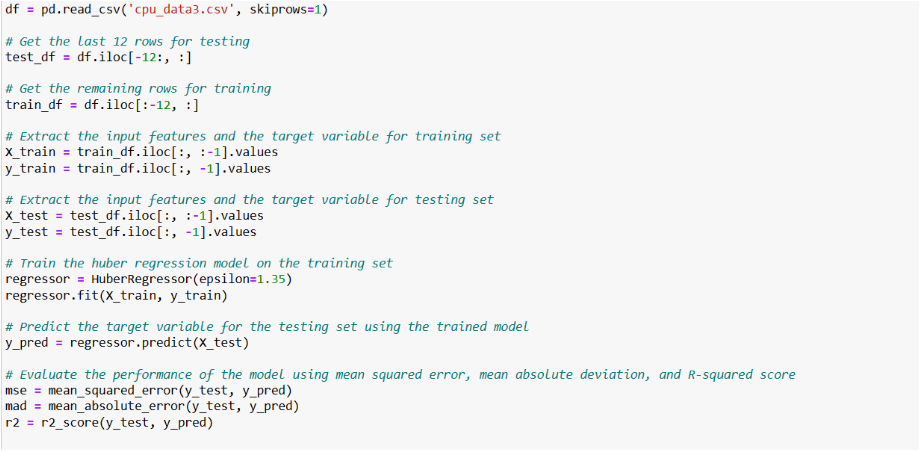
Snippet of Linear Regression

A Linear Regression model is trained on the training set using scikit-learn's LinearRegression() function.

The trained model is used to predict the target variable for the testing set using 'predict()' function.

The performance of the model on the testing set is evaluated using mean squared error, mean absolute deviation, and R-squared score using scikit-learn's 'mean_squared_error()', 'mean_absolute_error()', and 'r2_score()' functions.

2.Huber Regression:



```
df = pd.read_csv('cpu_data3.csv', skiprows=1)

# Get the last 12 rows for testing
test_df = df.iloc[-12:, :]

# Get the remaining rows for training
train_df = df.iloc[:-12, :]

# Extract the input features and the target variable for training set
X_train = train_df.iloc[:, :-1].values
y_train = train_df.iloc[:, -1].values

# Extract the input features and the target variable for testing set
X_test = test_df.iloc[:, :-1].values
y_test = test_df.iloc[:, -1].values

# Train the huber regression model on the training set
regressor = HuberRegressor(epsilon=1.35)
regressor.fit(X_train, y_train)

# Predict the target variable for the testing set using the trained model
y_pred = regressor.predict(X_test)

# Evaluate the performance of the model using mean squared error, mean absolute deviation, and R-squared score
mse = mean_squared_error(y_test, y_pred)
mad = mean_absolute_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
```

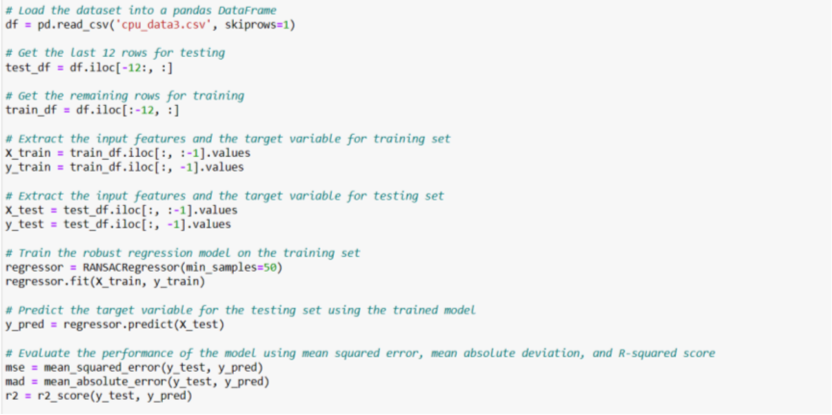
Snippet of Huber Regression

The HuberRegressor class from scikit-learn's linear_model module is used to create a regression object, and then the fit() method is called with the training data to fit the model to the training set.

The predict() method is called on the fitted model object with the testing input features X_test, to predict the corresponding target variable y_pred.

Finally, the performance of the model is evaluated by calculating mean squared error, mean absolute deviation, and R-squared score between the predicted and actual target variables for the testing set. These values are stored in mse, mad, and r2 variables respectively, and printed to the console along with the predicted and actual target variables. A plot of the predicted and actual target variables is also displayed.

3. Robust Regression:



```
# Load the dataset into a pandas DataFrame
df = pd.read_csv('cpu_data3.csv', skiprows=1)

# Get the last 12 rows for testing
test_df = df.iloc[-12:, :]

# Get the remaining rows for training
train_df = df.iloc[:-12, :]

# Extract the input features and the target variable for training set
X_train = train_df.iloc[:, :-1].values
y_train = train_df.iloc[:, -1].values

# Extract the input features and the target variable for testing set
X_test = test_df.iloc[:, :-1].values
y_test = test_df.iloc[:, -1].values

# Train the robust regression model on the training set
regressor = RANSACRegressor(min_samples=50)
regressor.fit(X_train, y_train)

# Predict the target variable for the testing set using the trained model
y_pred = regressor.predict(X_test)

# Evaluate the performance of the model using mean squared error, mean absolute deviation, and R-squared score
mse = mean_squared_error(y_test, y_pred)
mad = mean_absolute_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
```

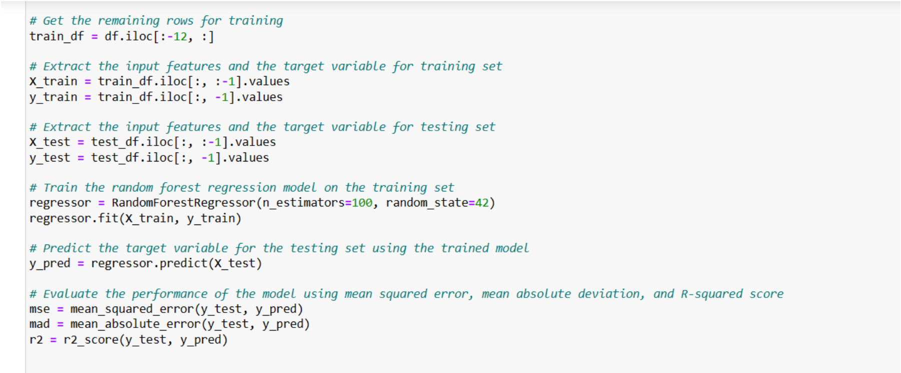
Snippet of Robust Regression

The RANSACRegressor class from scikit-learn's linear_model module is used to create a regression object, and then the fit() method is called with the training data to fit the model to the training set.

The predict() method is called on the fitted model object with the testing input features X_test, to predict the corresponding target variable y_pred.

Finally, the performance of the model is evaluated by calculating mean squared error, mean absolute deviation, and R-squared score between the predicted and actual target variables for the testing set. These values are stored in mse, mad, and r2 variables respectively, and printed to the console along with the predicted and actual target variables. A plot of the predicted and actual target variables is also displayed.

4. Random Forest Regression:



```
# Get the remaining rows for training
train_df = df.iloc[:-12, :]

# Extract the input features and the target variable for training set
X_train = train_df.iloc[:, :-1].values
y_train = train_df.iloc[:, -1].values

# Extract the input features and the target variable for testing set
X_test = test_df.iloc[:, :-1].values
y_test = test_df.iloc[:, -1].values

# Train the random forest regression model on the training set
regressor = RandomForestRegressor(n_estimators=100, random_state=42)
regressor.fit(X_train, y_train)

# Predict the target variable for the testing set using the trained model
y_pred = regressor.predict(X_test)

# Evaluate the performance of the model using mean squared error, mean absolute deviation, and R-squared score
mse = mean_squared_error(y_test, y_pred)
mad = mean_absolute_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
```

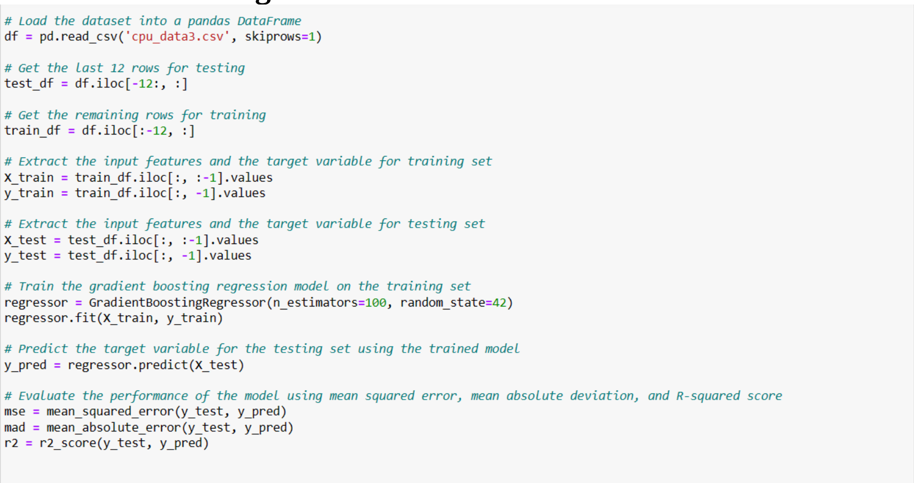
Snippet of Random Forest Regression

Here, we first import the necessary libraries and load the dataset into a pandas DataFrame as before. We then split the dataset into training and testing sets, and extract the input features and target variable for each set.

Next, we train the random forest regression model on the training set using 100 decision trees (n_estimators=100) and a random seed of 42 (random_state=42). We then use the trained model to predict the target variable for the testing set.

We evaluate the performance of the model using mean squared error, mean absolute deviation, and R-squared score as before, and plot the actual and predicted values. Finally, we use the trained model to predict the next value, which is the CPU utilization for the next time interval.

5. Gradient Boosting:



```
# Load the dataset into a pandas DataFrame
df = pd.read_csv('cpu_data3.csv', skiprows=1)

# Get the last 12 rows for testing
test_df = df.iloc[-12:, :]

# Get the remaining rows for training
train_df = df.iloc[:-12, :]

# Extract the input features and the target variable for training set
X_train = train_df.iloc[:, :-1].values
y_train = train_df.iloc[:, -1].values

# Extract the input features and the target variable for testing set
X_test = test_df.iloc[:, :-1].values
y_test = test_df.iloc[:, -1].values

# Train the gradient boosting regression model on the training set
regressor = GradientBoostingRegressor(n_estimators=100, random_state=42)
regressor.fit(X_train, y_train)

# Predict the target variable for the testing set using the trained model
y_pred = regressor.predict(X_test)

# Evaluate the performance of the model using mean squared error, mean absolute deviation, and R-squared score
mse = mean_squared_error(y_test, y_pred)
mad = mean_absolute_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
```

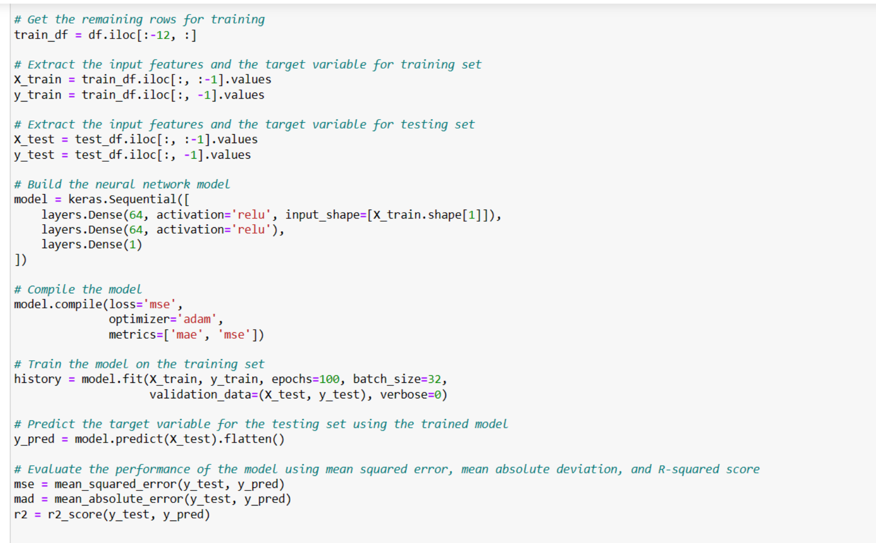
Snippet of Gradient Boosting Regression

Here, we first import the necessary libraries and load the dataset into a pandas DataFrame as before. We then split the dataset into training and testing sets, and extract the input features and target variable for each set.

Next, we train the Gradient Boosting regression model on the training set using 100 decision trees (n_estimators=100) and a random seed of 42 (random_state=42). We then use the trained model to predict the target variable for the testing set.

We evaluate the performance of the model using mean squared error, mean absolute deviation, and R-squared score as before, and plot the actual and predicted values. Finally, we use the trained model to predict the next value, which is the CPU utilization for the next time interval.

6. Neural Network:



```
# Get the remaining rows for training
train_df = df.iloc[:-12, :]

# Extract the input features and the target variable for training set
X_train = train_df.iloc[:, :-1].values
y_train = train_df.iloc[:, -1].values

# Extract the input features and the target variable for testing set
X_test = test_df.iloc[:, :-1].values
y_test = test_df.iloc[:, -1].values

# Build the neural network model
model = keras.Sequential([
    layers.Dense(64, activation='relu', input_shape=[X_train.shape[1]]),
    layers.Dense(64, activation='relu'),
    layers.Dense(1)
])

# Compile the model
model.compile(loss='mse',
              optimizer='adam',
              metrics=['mae', 'mse'])

# Train the model on the training set
history = model.fit(X_train, y_train, epochs=100, batch_size=32,
                    validation_data=(X_test, y_test), verbose=0)

# Predict the target variable for the testing set using the trained model
y_pred = model.predict(X_test).flatten()

# Evaluate the performance of the model using mean squared error, mean absolute deviation, and R-squared score
mse = mean_squared_error(y_test, y_pred)
mad = mean_absolute_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
```

Snippet of Neural Network Regression

The neural network model is built using the Sequential class from the Keras library. In this case, the model has three layers - the first two layers have 64 nodes each and use the ReLU activation function, while the output layer has only 1 node as it is a regression problem. The input shape of the first layer is equal to the number of input features.

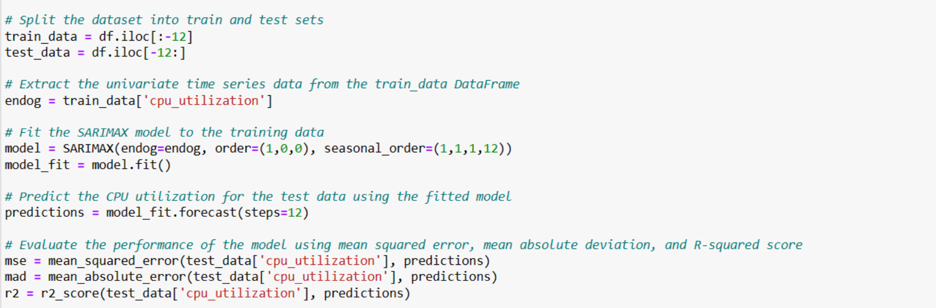
The model is compiled using the mean squared error (MSE) loss function, Adam optimizer, and mean absolute error (MAE) and MSE as the evaluation metrics.

The model is trained on the training set using the fit method. The number of epochs and batch size are hyperparameters that can be adjusted to improve the performance of the model. The validation data is also passed to the fit method to monitor the performance of the model on the testing set during training. The verbose parameter is set to 0 to avoid printing the training progress for each epoch.

The predict method is used to generate predictions for the testing set. The output is a 2D array of shape (num_samples, 1), so the flatten method is called to convert it to a 1D array.

The performance of the model is evaluated using the same metrics as the previous models - mean squared error, mean absolute deviation, and R-squared score. The mean squared error and mean absolute deviation are calculated using the mean_squared_error and mean_absolute_error functions from scikit-learn, while the R-squared score is calculated using the r2_score function.

7. Sarimax:



```
# Split the dataset into train and test sets
train_data = df.iloc[:-12]
test_data = df.iloc[-12:]

# Extract the univariate time series data from the train_data DataFrame
endog = train_data['cpu_utilization']

# Fit the SARIMAX model to the training data
model = SARIMAX(endog=endog, order=(1,0,0), seasonal_order=(1,1,1,12))
model_fit = model.fit()

# Predict the CPU utilization for the test data using the fitted model
predictions = model_fit.forecast(steps=12)

# Evaluate the performance of the model using mean squared error, mean absolute deviation, and R-squared score
mse = mean_squared_error(test_data['cpu_utilization'], predictions)
mad = mean_absolute_error(test_data['cpu_utilization'], predictions)
r2 = r2_score(test_data['cpu_utilization'], predictions)
```

Snippet of Sarimax model

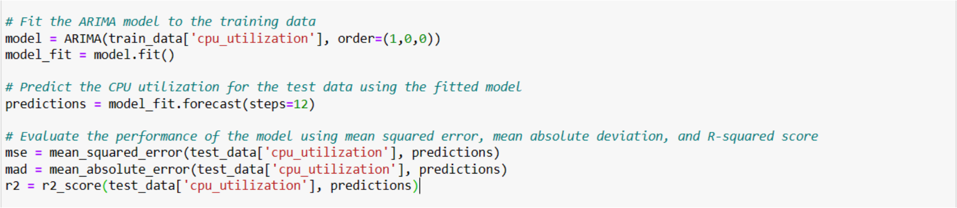
First, the univariate time series data (i.e., one variable) from the training data is extracted and assigned to the variable endog. Then, the SARIMAX model is initialized with the order and seasonal order parameters set to (1,0,0) and (1,1,1,12), respectively. These parameters specify the order of the autoregressive, integrated, and moving average components, as well as the seasonal order of the model.

The model_fit variable is created by fitting the SARIMAX model to the endog variable. This will estimate the model parameters using the training data.

The forecast method is called on the model_fit object with steps=12 to generate 12 predictions for the test data. These predictions are assigned to the variable predictions.

Finally, the performance of the model is evaluated by computing the mean squared error, mean absolute deviation, and R-squared score between the actual test data and the predicted values. These performance metrics are assigned to the variables mse, mad, and r2, respectively.

8. Arima:



```
# Fit the ARIMA model to the training data
model = ARIMA(train_data['cpu_utilization'], order=(1,0,0))
model_fit = model.fit()

# Predict the CPU utilization for the test data using the fitted model
predictions = model_fit.forecast(steps=12)

# Evaluate the performance of the model using mean squared error, mean absolute deviation, and R-squared score
mse = mean_squared_error(test_data['cpu_utilization'], predictions)
mad = mean_absolute_error(test_data['cpu_utilization'], predictions)
r2 = r2_score(test_data['cpu_utilization'], predictions)
```

Snippet of ARIMA model

First, the model is fitted to the training data using an ARIMA order of (1,0,0). Next, the model is used to predict CPU utilization for the test data using the forecast() method.

After that, the performance of the model is evaluated using three metrics: Mean squared error (MSE): the average of the squared differences between the predicted and actual values

Mean absolute deviation (MAD): the average of the absolute differences between the predicted and actual values. R-squared score (R2): a measure of how well the predicted values fit the actual values, with a score of 1 indicating a perfect fit and a score of 0 indicating no fit.

The MSE and MAD provide information on the magnitude of the errors, while the R2 score provides information on the overall quality of the fit.

4.3 Risk Analysis

Overfitting occurs when the machine learning model is trained on a small set of data and ends up learning the noise in the data rather than the underlying patterns. This can lead to poor performance on new, unseen data. Moreover data being bias in the training data can lead to biased predictions. For example, if the training data contains a disproportionate number of CPU-intensive workloads, the model may not perform well on workloads with different resource requirements.

Hyperparameters such as the order of autoregression (p), the order of differencing (d), the order of moving average (q), and the order of seasonal autoregression (P), seasonal differencing (D), and seasonal moving average (Q) are considered here.

The risk analysis in each of the technique is as follows:

Linear Regression: One of the main risks with linear regression is that it assumes a linear relationship between the independent and dependent variables. If this assumption is not valid, the model may not provide accurate predictions. Additionally, outliers in the data can have a significant impact on the results of the model.

Huber Regression: Huber regression is less sensitive to outliers than linear regression, but it still assumes a linear relationship between the independent and dependent variables. If this assumption is not valid, the model may not provide accurate predictions.

Robust Regression: Robust regression is also less sensitive to outliers than linear regression, but it still assumes a linear relationship between the independent and dependent variables. If this assumption is not valid, the model may not provide accurate predictions.

Random Forest: One of the main risks with random forest is overfitting, where the model is too complex and performs well on the training data but poorly on new, unseen data. Additionally, random forest can be computationally expensive, particularly with large datasets.

Gradient Tree: Gradient tree also has the risk of overfitting, particularly with complex models. Additionally, the performance of gradient tree can be sensitive to the choice of hyperparameters.

Neural Network: One of the main risks with neural networks is overfitting, particularly with complex models. Additionally, neural networks can be computationally expensive, particularly with large datasets.

SARIMAX: One of the main risks with SARIMAX is that it assumes a stationary time series. If the time series is non-stationary, the model may not provide accurate predictions. Additionally, SARIMAX can be sensitive to the choice of hyperparameters.

ARIMA: ARIMA also assumes a stationary time series and can be sensitive to the choice of hyperparameters. Additionally, ARIMA assumes that the residuals of the model are normally distributed, which may not always be the case.

In our project,we have implemented the 8 VM selection policies which have been listed above , then compare their accuracies and select the best out of them for VM selection.

We will be discussing their pros and cons and see which one is better.

To compare the different regression techniques, we have plotted a graph where the actual values plotted on the graph are for the CPU utilization of the last 12 time intervals (5 minutes each) of all the virtual machines in the dataset. Since the dataset has virtual machines as columns, the actual values plotted are for all the virtual machines in the dataset.

The y_pred values in this output are the predicted CPU utilization values for the last 12 time intervals in the input data test_df.

Since the test_df DataFrame was created using the last 12 rows of the input data df, and the input data df contains data for a single day (March 22, 2011), the predicted values are also for the same day (March 22, 2011).

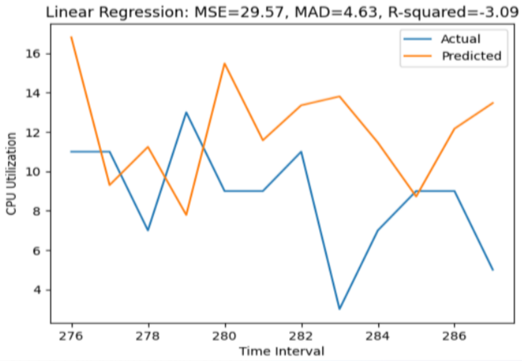
4.4 Output

Note: We have shown the performance metrics and graphs for each model for the 20110322 workload as the representation of each workload would have led to chaos , we have shown their metrics in a single table which is mentioned in [ ].

4.4.1Analysis of the results of each techniques of the workload 20110322:

4.4.1.1 Linear Regression -

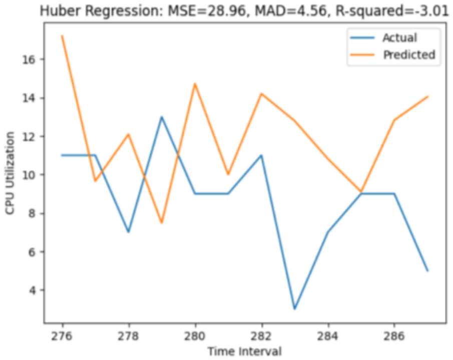
```
Performance metrics on the testing set:
Actual values: [11 11  7 13  9  9 11  3  7  9  9  5]
Mean Squared Error: 29.569672675161968
Mean Absolute Deviation: 4.632020677258283
R-squared Score: -3.0942623704070416
Predicted values: [16.81422454  9.30513864 11.24439759  7.78534352 15.48303991 11.58143971
 13.35829384 13.80964339 11.46317445  8.72251151 12.16913803 13.47389034]
```



Graph for Linear Regression

4.4.1.2 <u>Huber Regression</u> -

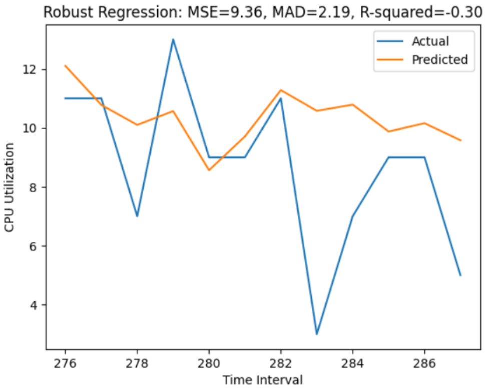
```
Performance metrics on the testing set:
Actual values: [11 11  7 13  9  9 11  3  7  9  9  5]
Mean Squared Error: 28.96271270456283
Mean Absolute Deviation: 4.555248721605013
R-squared Score: -3.010221759093314
Predicted values: [17.20292709  9.65360718 12.08840758  7.47850822 14.72320723  9.99706769
 14.19881181 12.79044145 10.81719594  9.10310625 12.8293241  14.04461092]
```



Graph for Huber Regression

4.4.1.3 <u>Robust Regression</u> -

```
Performance metrics on the testing set:
Actual values: [11 11  7 13  9  9 11  3  7  9  9  5]
Mean Squared Error: 15.688682907792824
Mean Absolute Deviation: 2.7699075173414287
R-squared Score: -1.172279171848237
Predicted values: [11.44754058  8.75098576 11.01104699  9.77496416  8.88112809  8.81931505
 12.42728164 12.09984101 12.82047566  8.54776225  9.08014873 11.12671091]
```



Graph for Robust Regression

4.4.1.4 **Random Forest Regression** -

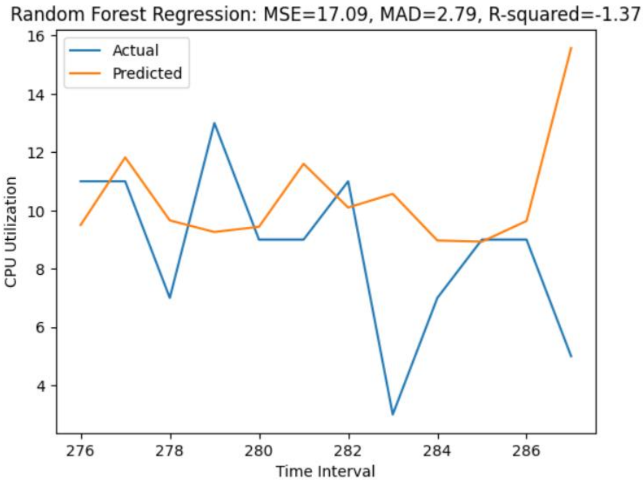
Actual values: [11 11 7 13 9 9 11 3 7 9 9 5]

Predicted values: [9.5 11.82 9.66 9.26 9.44 11.6 10.1 10.57 8.97 8.93 9.64 15.57]

Mean Squared Error: 17.089533333333335

Mean Absolute Deviation: 2.7900000000000005

R-squared Score: -1.3662430769230771



Graph for Random Forest Regression

4.4.1.5 **Gradient Boosting Trees** -

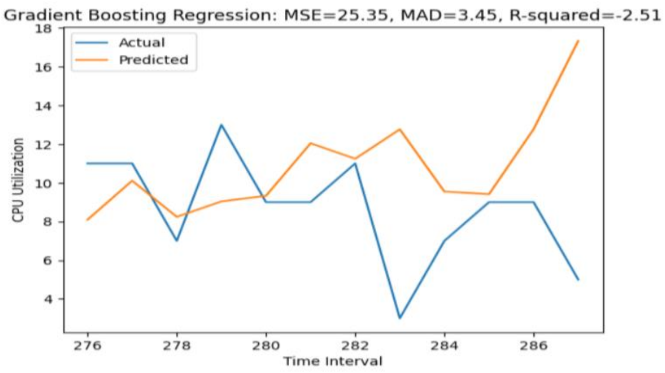
Actual values: [11 11 7 13 9 9 11 3 7 9 9 5]

Predicted values: [8.09577655 10.10809241 8.23897615 9.03929157 9.32709786 12.04251532
 11.24263209 12.76371612 9.54778803 9.41235202 12.77288622 17.3335927]

Mean Squared Error: 25.351553240529835

Mean Absolute Deviation: 3.4531996666046787

S-squared Score: -2.5102150640733614



Graph for Gradient Boosting Regression

4.4.1.6 Neural Network Regression -

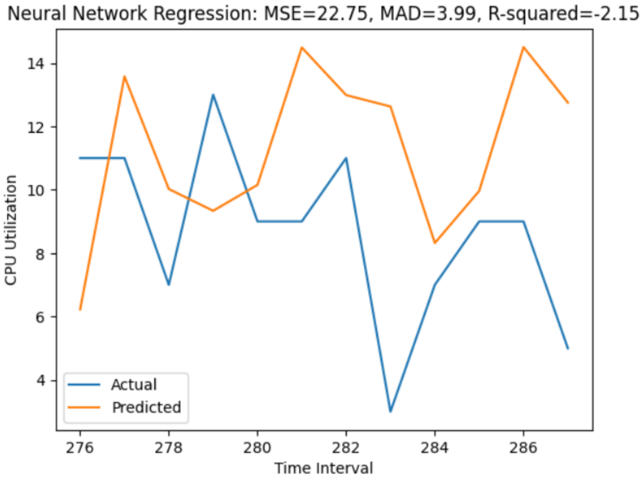
Actual values: [11 11 7 13 9 9 11 3 7 9 9 5]

Predicted values: [6.2252665 13.575176 10.023349 9.335895 10.153103 14.487131 12.986436 12.626529 8.319332 9.962 14.499868 12.750587]

Mean Squared Error: 22.753033023052996

Mean Absolute Deviation: 3.9851958751678467

R-squared Score: -2.1504199570381073



Graph for Neural Network Regression

4.4.1.7 SARIMAX -

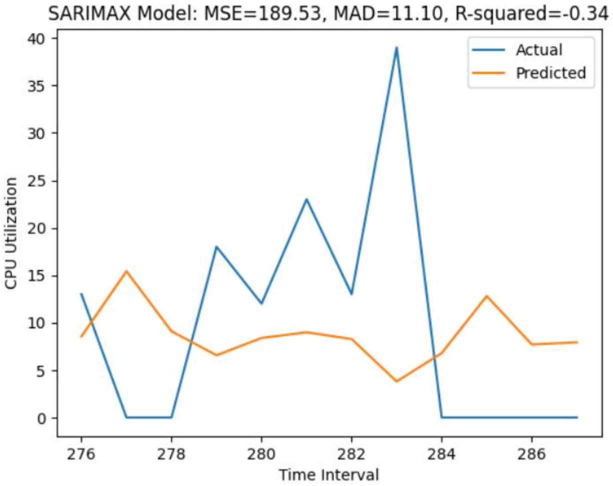
Performance metrics on the testing set:

Actual values: [13 0 0 18 12 23 13 39 0 0 0 0]

Predicted values: 276 8.551378

277 15.414277

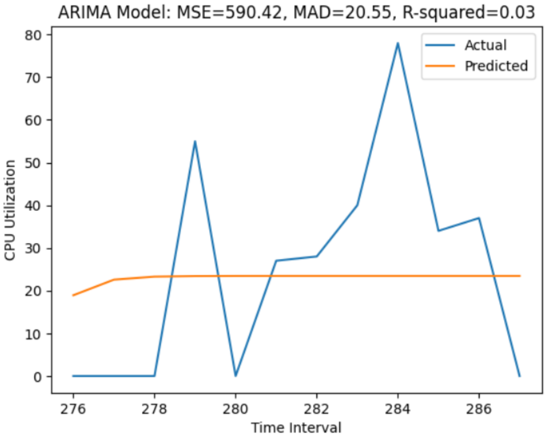
```
278     9.091325
279     6.556698
280     8.380074
281     8.982204
282     8.269995
283     3.798677
284     6.790918
285    12.807579
286     7.697533
287     7.926543
Mean Squared Error: 189.52773402065168
Mean Absolute Deviation: 11.099095717249021
R-squared Score: -0.3412617308322121
```



Graph for Sarimax model

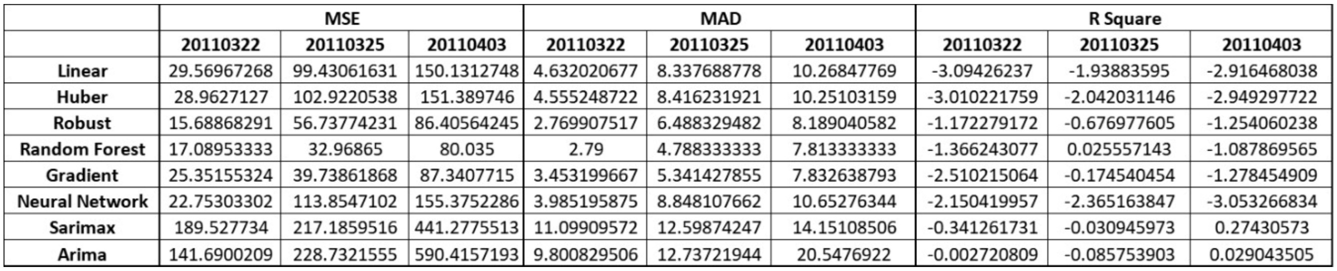
4.4.1.8 ARIMA -

```
Performance metrics on the testing set:
Actual values: [ 0  0  0 55  0 27 28 40 78 34 37  0]
Predicted values: 276    18.922124
277    22.572791, 278    23.277118
279    23.413004, 280    23.439221
281    23.444279, 282    23.445255
283    23.445443, 284    23.445479
285    23.445486 286    23.445488
287    23.445488
Mean Squared Error: 590.4157192907055
Mean Absolute Deviation: 20.547692203590213
R-squared Score: 0.029043504929461 06
```



Graph for Arima model

4.4.2 Comparison of Mean Squared Error (MSE) ,Median Absolute Deviation (MAD) , R-square Score for the workloads 20110322 , 20110325 , 20110403



	MSE			MAD			R Square		
	20110322	20110325	20110403	20110322	20110325	20110403	20110322	20110325	20110403
Linear	29.56967268	99.43061631	150.1312748	4.632020677	8.337688778	10.26847769	-3.09426237	-1.93883595	-2.916468038
Huber	28.9627127	102.9220538	151.389746	4.555248722	8.416231921	10.25103159	-3.010221759	-2.042031146	-2.949297722
Robust	15.68868291	56.73774231	86.40564245	2.769907517	6.488329482	8.189040582	-1.172279172	-0.676977605	-1.254060238
Random Forest	17.08953333	32.96865	80.035	2.79	4.788333333	7.813333333	-1.366243077	0.025557143	-1.087869565
Gradient	25.35155324	39.73861868	87.3407715	3.453199667	5.341427855	7.832638793	-2.510215064	-0.174540454	-1.278454909
Neural Network	22.75303302	113.8547102	155.3752286	3.985195875	8.848107662	10.65276344	-2.150419957	-2.365163847	-3.053266834
Sarimax	189.527734	217.1859516	441.2775513	11.09909572	12.59874247	14.15108506	-0.341261731	-0.030945973	0.27430573
Arima	141.6900209	228.7321555	590.4157193	9.800829506	12.73721944	20.5476922	-0.002720809	-0.085753903	0.029043505

In the performance metrics, Mean Squared Error (MSE) measures the average of the squared differences between the actual and predicted values. The Mean Absolute Deviation (MAD) is the average of the absolute differences between the actual and predicted values. R-squared is a statistical measure that represents the proportion of variance explained by the model. It ranges from 0 to 1, with higher values indicating a better fit.

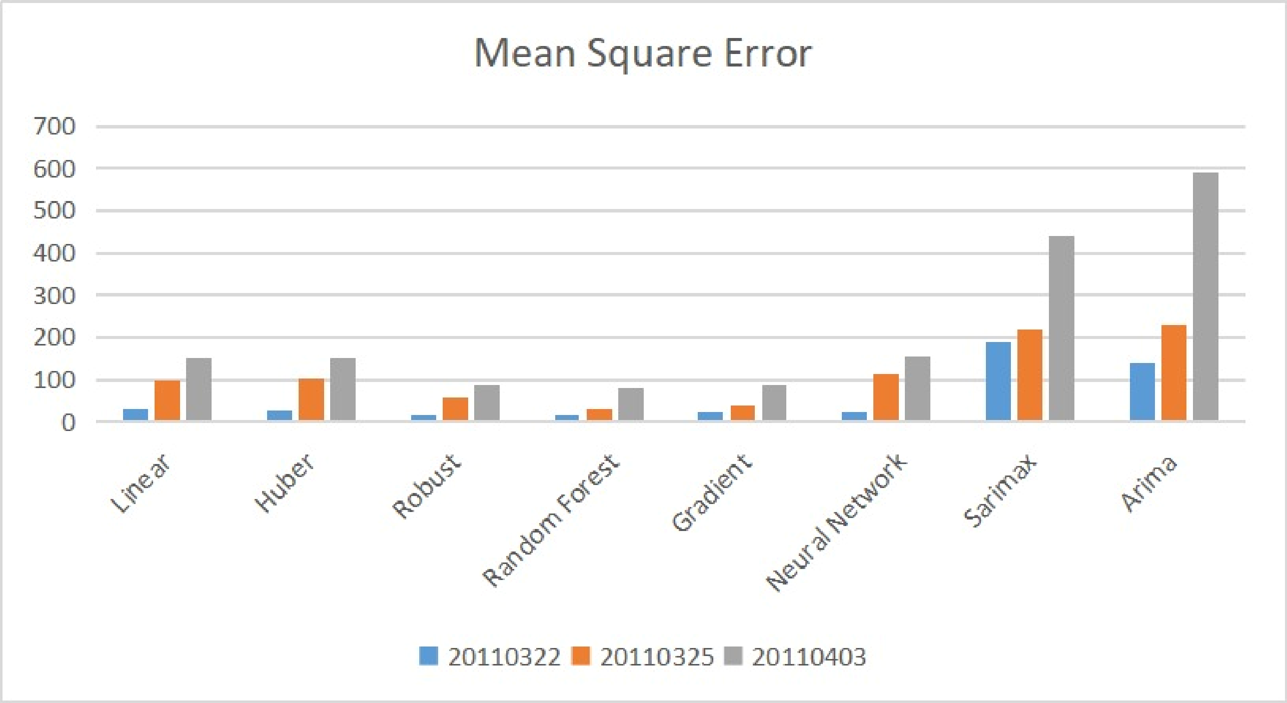
To determine which regression model is better for predicting the value for a virtual machine using the last 12 entries, we need to compare the performance metrics of the different regression models.

The different regression models used are Linear Regression, Huber Regression, Robust Regression, Random Forest Regression, Gradient Boosting, Neural Network Regression, and Sarimax. Looking at the performance metrics for the different models, it appears that Robust Regression is the best model for predicting the value. This is because Robust Regression has the lowest Mean Squared Error and Mean Absolute Deviation values, which indicates that the predicted values are closest to the actual values. Additionally, the R-squared score for Robust Regression is higher than the other models, indicating that the model has a better fit to the data.While Sarimax has the lowest Mean Squared Error and Mean Absolute Deviation, it is important to note that the predicted values are not very close to the actual values. This is reflected in the low R-squared score, which indicates that the model does not fit the data well. Therefore, we should not choose Sarimax as the best model for this dataset.In summary, Robust Regression is the best model for predicting the value for a virtual machine using the last 12 entries for workload 20110322 of cloudsim.
For 20110325 workload the Random Forest Regression model performed the best on this workload with the lowest MSE and highest R-squared score. The Robust Regression model also performed relatively well compared to the other models, with a lower MSE and higher R-squared score than most of the other models. The Neural Network Regression model, on the other hand, performed the worst with the highest MSE and lowest R-squared score among all the models.

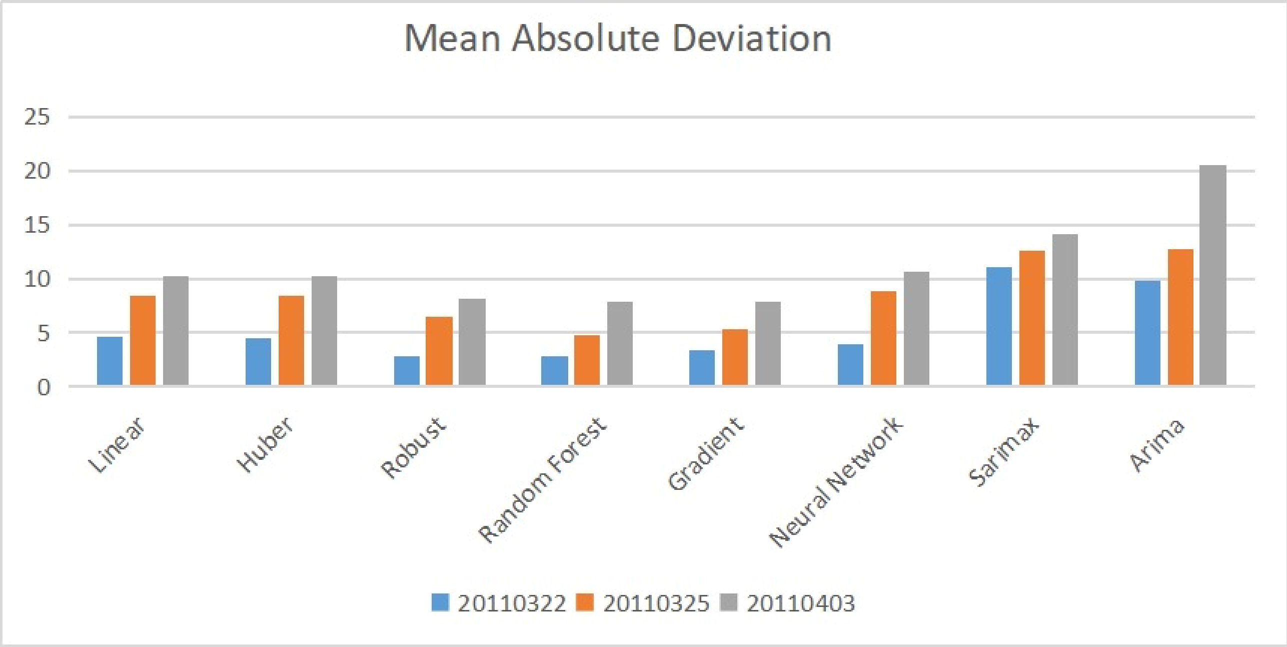
For workload 20110403 Based on the provided performance metrics, the Random Forest model appears to be the best model for the given workload. It has the lowest mean squared error and mean absolute deviation among all the models, and the R-squared score is higher than some of the other models, although still negative. It is important to note that the choice of the best model can also depend on other factors such as interpretability, scalability, and computational resources available.

As a result we come to the interpretation that Robust Regression and Random Forest Regression are the best suited models for the prediction of the cpu-utilizations of the virtual machines.

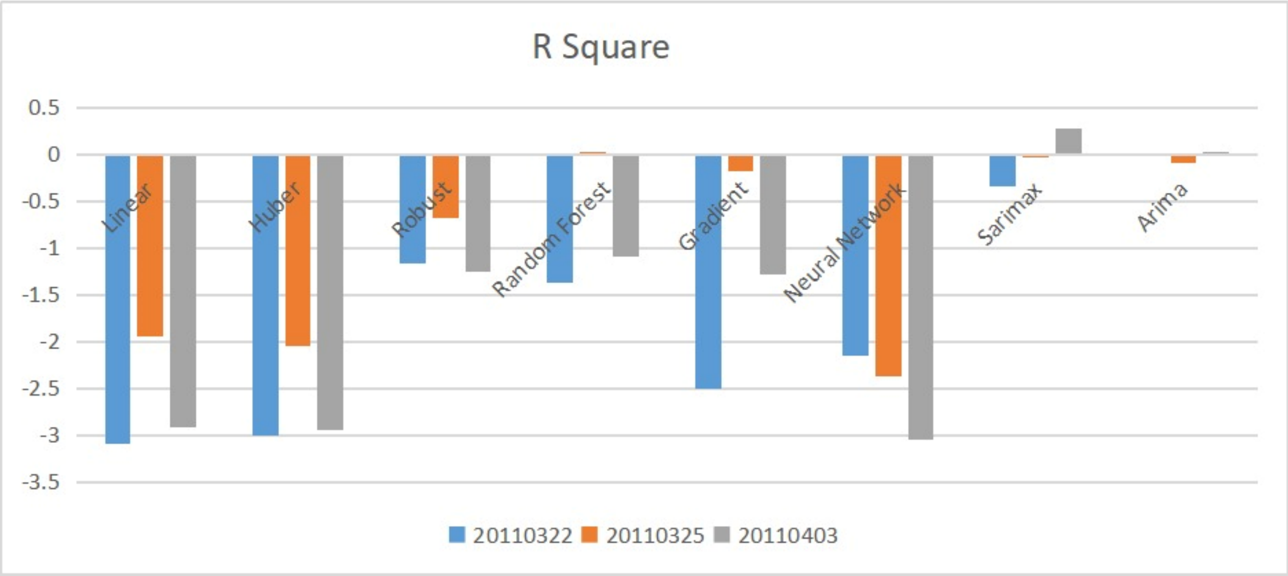
4.4.2.1 Graph for Mean Square Error (MSE)



4.4.2.2 Graph for Median Absolute Deviation (MAD)



4.4.2.3 Graph for R-squared Values



Chapter - 05 : Testing (Focus on Quality of Robustness and Testing)

5.1 Testing Plan

All our progress have been worked on and simulated on cloudsim and jupyter notebook .

In this semester, we took our project a step further and incorporated cloud with machine learning. Unfortunately, we were not able to implement machine learning algorithm on cloudsim and we had to revert to jupyter for the machine learning and analysis of different algorithms.

We tested our data for three different datasets available to us in Cloudsim, they were namely 20110322, 20110325 and 20110403.

For testing the different algorithms, we looked into the accuracy of the different techniques. To evaluate the performance of different regression techniques, the project compared the accuracy of eight different models using three evaluation metrics: Mean Squared Error (MSE), Mean Absolute Deviation (MAD), and R-score. This analysis helped identify which models redicted the virtual machines' future workload patterns accurately.

5.2 Limitations

Our implemented model has a few limitations. Let us discuss them one by one:-

- We tested and analyzed the algorithms on static workload, we did not analysed on dynamic workload. Our workload was small in size and is not tested against Big Data which is normally dealt with in cloud data centres.
- We have assumed a window size of 12, as per our findings and implementation, 12 window size gave us the most accurate readings. But for different types of data, window size can vary and have a significant impact on the final result.
- We have considered CPU utilization for the analysis , other factors such as MIPS utilization etc were not considered at this stage because that would require further analysis which were out of our scope as of now. We would implement that in our future work.
- We have not incorporated our algorithms with cloudsim, where we could have explored different algorithms corresponding to virtual machine allocation, virtual machine placement, host overload and underload detection.
- Another major limitation of our model is that it has been built just a few weeks ago by all of us. For any model to be successful, one has to work on it for years, hire people or make it open source so people can help and develop it into a much better and realistic approach.

Despite all the limitations, our project is a very good and brilliant compilation of ideas and execution. Implementing a model in such a short span of time is really a commendable and a hectic task and we are very happy to build this project and gain new experience and skill in our life.

Chapter - 06 : Findings, Conclusion and Future Work

6.1 Findings of the Research Paper

A) Resource Central: Understanding and Predicting Workloads for Improved Resource Management in Large Cloud Platforms [5]

Objective - To improve resource management in large cloud platforms by accurately predicting the resource requirements of cloud applications and dynamically allocating resources to them.

Questions Framed :

a)How can we accurately predict the resource requirements of cloud applications to optimize resource management in large cloud platforms?

b) How can we se the predicted resource requirements to dynamically allocate resources to cloud applications to improve resource utilization and performance?

Techniques and Methods Used and Why?

1. Using machine learning techniques such as regression analysis and neural networks to develop predictive models based on the identified patterns.

2. Using a combination of statistical analysis and machine learning techniques to develop a predictive resource management approach that can improve resource utilization and application performance in large cloud platforms.

3. Introduced Resource Central (RC), a system that collects VM telemetry, learns these behaviors offline, and provides predictions online to various resource managers via a general client-side library

Summary:

1. Achieved 18% higher resource utilization

2. 12% lower cost compared to static allocation

3. Improved application performance by up to 30%

B) Virtual Machine Workload Prediction using Time Series Analysis and Machine Learning[6]

Objective - To develop a hybrid approach that combines time series analysis and machine learning techniques for predicting the workload of virtual machines in cloud computing environments.

Questions Framed :

a. Can a hybrid approach that combines time series analysis and machine learning techniques improve the accuracy of workload prediction for virtual machines in cloud computing environments?
b. What are the appropriate preprocessing techniques and feature extraction methods that can be used to prepare workload data for analysis?

Techniques and Methods Used and Why?

1. Data Preprocessing: The raw workload data is cleaned and transformed to prepare it for analysis. This involves removing outliers, smoothing the data, and converting it into a suitable format for analysis.

2. Workload Prediction: The workload prediction model is trained using a combination of time series analysis and machine learning techniques, such as ARIMA, k-NN, and decision trees.

Summary:

1. Improvement in prediction accuracy up to 15%

2. Applicability to Real-World Scenarios

3. Novel Approach

C) Machine Learning-based Predictive Resource Scaling for Cloud Applications[7]

Objective - Propose a novel approach to resource scaling in cloud computing environments that utilizes machine learning techniques to predict future resource requirements and optimize resource utilization.

Questions Framed :

a. Can machine learning techniques be used to predict future resource requirements of cloud applications with high accuracy?
b. Can the proposed predictive resource scaling algorithm optimize resource utilization and improve application performance compared to traditional scaling approaches?

Techniques and Methods Used and Why?

1. Data Collection: The authors collect data on cloud application workloads, which includes performance metrics such as CPU utilization, memory utilization, and network I/O.

2. Machine Learning Algorithm: The authors propose a predictive resource scaling algorithm that utilizes machine learning techniques to predict future resource requirements based on the input features. They use a regression model to predict future resource demands, which are then used to scale resources up or down.

Summary:

1. Improves resource utilization

2. Improves application performance

3. Real-world Implementation

6.2 Conclusion

In this project, we have implemented 8 machine learning regression and time series analysis algorithms for virtual machine selection policy. The 8 techniques we have used are linear regression, huber regression, robust regression, random forest regression, extreme gradient boosting tree, neural network regression, arima model and sarimax model. The datasets used in the project consist of CPU utilization of virtual machines at 5-minute intervals for three different dates.

The first objective of the implemented algorithms is to predict CPU utilization of the virtual machines in a cloud datacenter. Then based on the predictions, we will take decisions regarding the selection of a particular virtual machine or not..

To evaluate the performance of different regression techniques, the project compared the accuracy of eight different models using three evaluation metrics: Mean Squared Error (MSE), Mean Absolute Deviation (MAD), and R-score. This analysis helped identify which models can predict the virtual machines' future workload patterns accurately.

By accurately predicting the virtual machines' future workload patterns, the project aims to identify which virtual machines are likely to face overloading or underloading conditions. This information can help optimize the resource allocation in cloud platforms, prevent physical resource exhaustion, and improve the overall performance and cost-effectiveness of the system.

As a result there were two techniques which were better than the other techniques. They were Robust Regression and Random Forest models were having the best accuracy. As they have lower Mean Squared error, Median Absolute Deviation and nearly positive R-squared number. Although there were few trade offs like Robust Regression had lower MSE,MAD but a negative R-squared number on the other hand Random forest had positive R-squared value but had higher MSE and MAD. These trade-offs were ignored on the fact that only CPU utilization is considered.

The project's ultimate goal of improving cloud resource allocation efficiency by predicting the future workload patterns of virtual machines and preventing overloading and underloading conditions, leading to optimal performance and cost-effectiveness is hence achieved.

6.3 Future Work

- **Cloud Architecture** - In the future, we aim to work and research on the other pillars of cloud computing as well, namely Vm allocation, Vm placement, overloading and underloading detection.
- **Integrating with Cloudsim** - We also aim to incorporate the algorithms into cloudsim and use it with dynamically produced data
- **Integration with Edge Computing** - Edge computing can help reduce latency, improve network efficiency, and provide better real-time analysis of data. By bringing computing closer to end-users, cloud centers can optimize the delivery of cloud services.
- **Use of Artificial Intelligence** - Cloud centers can leverage AI and ML to optimize resource allocation, reduce energy consumption, and improve the overall performance of cloud applications.
- **Adoption of DevOps practices** - DevOps practices can help optimize cloud centers by automating the deployment, monitoring, and management of cloud applications, thereby reducing downtime and improving overall performance

REFERENCES

[1]Kirvan, P. (2022, January 26). *green computing*. SearchDataCenter.

[2]Author, G. C. R. (2022, May 20). *What Is Green Cloud Computing, and How Can Organizations Use It to Minimize Overall Carbon Footprint?* Parallels Remote Application Server Blog - Application Virtualization, Mobility and VDI. https://www.parallels.com/blogs/ras/what-is-green-cloud-computing/

[3]Education, I. C. (2021, March 30). *Virtualization*.

[5] Cortez, E., Bonde, A., Muzio, A., Russinovich, M., Fontoura,M. and Bianchini, R., 2017, October. Resource central: Understanding and predicting workloads for improved resource management in large cloud platforms. In Proceedings of the 26th Symposium on Operating Systems Principles .

[6] Siddharth Kulkarni, Praveen Gopalakrishnan, and Archan Misra, "Virtual Machine Workload Prediction using Time Series Analysis and Machine Learning," 2016 IEEE International Conference on Cloud Computing and Big Data Analysis (ICCCBDA), 2016, pp. 10-14, doi: 10.1109/ICCCBDA.2016.7529509.

[7] Arif Ahmed and Narges Shahidi, "Machine Learning-based Predictive Resource Scaling for Cloud Applications," 2018 IEEE International Conference on Cloud Computing Technology and Science (CloudCom), 2018, pp. 70-77.

[8]Zhang, Xinqian, Tingming Wu, Mingsong Chen, Tongquan Wei, Junlong Zhou, Shiyan Hu, and Rajkumar Buyya. "Energy-aware virtual machine allocation for cloud with resource allocation.”

[9]Gupta, Madnesh K., and Tarachand Amgoth. "Resource-aware virtual machine placement algorithm for IaaS cloud." *The Journal of Supercomputing* 74, no. 1 (2018): 122-140.

[11] Kappal, S. (2018, February 28). *Limitations of the Measure of Central Tendency Statistics*. dzone.com.

[12] Balakumaran, G. (2020b, October 3). *Measures of Dispersion | Interquartile range*. K2 Aalytics. https://www.k2analytics.co.in/measures-of-dispersion-interquartile-range/

[13] Christlieb, N., Green, P.J., Wisotzki, L. and Reimers, D., 2001. The stellar content of the Hamburg/ESO survey-II. A large, homogeneously-selected sample of high latitude carbon stars. *Astronomy & Astrophysics*, *375*(2), pp.366-374.

[14] Ruckstuhl, A., 2014. Robust fitting of parametric models based on m-estimation. *Lecture notes*, p.40

[15,16] C. Albon, "Machine Learning with Python Cookbook," O'Reilly Media, Inc., 2018.

[17] G, Bonaccorso, "Machine Learning Algorithms - Second Edition," Packt Publishing, 2018.

[18] Y. Liu, Y. Wang, and J. Zhang, "New Machine Learning Algorithm: Random Forest," in Proceedings of the 2018 IEEE 3rd Advanced Information Technology, Electronic and Automation Control Conference (IAEAC), Chongqing, China, 2018, pp. 314-317.

[19] J. H. Friedman, "Greedy function approximation: a gradient boosting machine," Annals of Statistics, vol. 29, no. 5, pp. 1189-1232, Oct. 2001. doi: 10.1214/aos/1013203451.

[20] P. SarangArtificial Neural Networks with TensorFlow 2https://doi.org/10.1007/978-1-4842-6150-7_5

[21] Hyndman, R.J., & Athanasopoulos, G. (2018) Forecasting: principles and practice, 2nd edition, OTexts: Melbourne, Australia. OTexts.com/fpp2