

# **SAVEETHA INSTITUTE OF MEDICAL AND TECHNICAL SCIENCES, CHENNAI – 602105**

**CAPSTONE PROJECT REPORT**

# **TITLE**

**Best cloud node prediction and matchmaking using**

**cloud resource prediction pattern**

***Submitted to***

# **SAVEETHA SCHOOL OF ENGINEERING**

***By***

## P. Dinesh Karthik (19210043)

***Guided by***

## Dr. J. Chenni Kumaran

**PROBLEM STATEMET:**

1. **Problem Statement:**

**Cloud Node Prediction:** Given a set of cloud resources (nodes), predict which node is best suited for a specific task or workload.

**Matchmaking:** Once the prediction is made, match the task to the most appropriate cloud node based on various criteria.

* The central topic in cloud computing isbest cloud node prediction for assigned workloads.
* The goal is to predict which cloud resource (node) is most suitable for a specific task.
* Based on resource evaluation, cloud resources are ranked, and these rankings guide the matchmaking process.

1. **Quality of Service (QoS) Parameters**:

* QoS parameters, such as latency, throughput, reliability, and availability, play a crucial role in ranking cloud resources.
* These parameters help determine the suitability of a resource for different types of workloads.
* The ranking information is stored in a hash table, where the resource ID serves as the index value.

1. **Evaluation Cum Classification (EC2)**:

The EC2 calculation involves three key aspects:

* **Cloud Resource Evaluation and Ranking**: Assigning scores to resources based on their capabilities.
* **Cloud Evaluation Framework by Scheduler**: Efficiently scheduling tasks to appropriate resources.
* **Job Assessment and Grouping**: Grouping similar tasks for optimal execution.

1. **CRPP-PSO (Cloud Resource Prediction Pattern - Particle Swarm Optimization)**:

* The CRPP-PSO algorithm represents an exciting advancement in this field.
* It integrates Particle Swarm Optimization (PSO)with theCloud Resource Prediction Pattern (CRPP)**.**
* The CRPP-Pattern is designed to locate the best cloud resource based on task expectations.
* The cloud server plays a critical role in decision-making, while cloudlets serve as computing resources within a cluster.
* The array of cloudlets collaborates to form a powerful cluster, ensuring optimal service delivery for the given task.

1. **Significance**:

* [Trial results demonstrate the significance of the proposed framework in discovering the best cloud node for specific workloads](https://ieeexplore.ieee.org/document/9770725/).

### **Proposed Design Work in Best cloud node prediction and matchmaking:**

### **Dashboard Overview:**

### Start with a clean and intuitive dashboard.

### Provide an overview of the cloud nodes, their current status, and resource utilization.

### Include visualizations (such as graphs or heatmaps) to depict real-time performance metrics.

### Ensure that critical information is easily accessible at a glance.

### **Resource Selection:**

### Allow users to select the type of workload or task they want to deploy.

### Based on the workload characteristics (e.g., compute-intensive, memory-intensive), recommend suitable cloud nodes.

### Provide clear labels and descriptions for each node to aid decision-making.

### **Predictive Insights:**

### Incorporate predictive models to suggest the best node for a given workload.

### Display confidence scores or probabilities to indicate the reliability of predictions.

### Highlight any potential bottlenecks or resource constraints.

### **Customization Options:**

### Let users customize prediction parameters (e.g., QoS thresholds, historical data window).

### Allow them to adjust prediction algorithms based on their specific requirements.

### **Dashboard Overview:**

### Start with a clean and intuitive dashboard.

### Provide an overview of the cloud nodes, their current status, and resource utilization.

### Include visualizations (such as graphs or heatmaps) to depict real-time performance metrics.

### Ensure that critical information is easily accessible at a glance.

### **Resource Selection:**

### Allow users to select the type of workload or task they want to deploy.

### Based on the workload characteristics (e.g., compute-intensive, memory-intensive), recommend suitable cloud nodes.

### Provide clear labels and descriptions for each node to aid decision-making.

### **Predictive Insights:**

### Incorporate predictive models to suggest the best node for a given workload.

### Display confidence scores or probabilities to indicate the reliability of predictions.

### Highlight any potential bottlenecks or resource constraints.

### **Customization Options:**

### Let users customize prediction parameters (e.g., QoS thresholds, historical data window).

### Allow them to adjust prediction algorithms based on their specific requirements.

### **Alerts and Notifications:**

### Implement alerts for resource overload, failures, or anomalies.

### Notify users when a node's performance deviates significantly from predictions.

### Provide actionable recommendations to mitigate issues.

### **User Feedback Loop:**

### Gather feedback from users regarding prediction accuracy.

### Use this feedback to continuously improve the prediction models.

### Encourage users to report any discrepancies or unexpected behavior.

**IMPLEMENTATION:**

**Connecting Components in Cloud:**

**Resource Data Collection:**

* Gather historical data on cloud resources (nodes).
* Collect performance metrics such as CPU utilization, memory usage, and network latency.

**Feature Selection:**

* Identify relevant features (QoS parameters) that impact resource suitability.
* Examples: CPU speed, memory capacity, storage I/O.

**CRPP Model:**

* Build a predictive model (e.g., regression, decision tree) to estimate resource suitability scores.
* Train the model using historical data.

**Integration with Cloud Management System:**

* Connect the CRPP model to your cloud management system (e.g., AWS, Azure, Google Cloud).
* Ensure seamless communication between the prediction module and cloud APIs.

**Cloud Deployment:**

**Resource Monitoring and Data Collection:**

* Deploy monitoring agents on cloud nodes.
* Continuously collect real-time performance data.

**Prediction Module Deployment:**

* Deploy the CRPP model as a microservice or serverless function.
* Ensure scalability and fault tolerance.

**API Gateway:**

* Set up an API gateway to expose the prediction service.
* Handle authentication, rate limiting, and request routing.

**Load Balancing:**

* Distribute incoming requests across multiple instances of the prediction module.
* Optimize resource utilization.

**Project Testing:**

**Unit Testing:**

* Test individual components (CRPP model, API endpoints) in isolation.
* Verify correctness and edge cases.

**Integration Testing:**

* Test the entire system end-to-end.
* Validate data flow, predictions, and resource allocation.

**Stress Testing:**

* Simulate high loads (concurrent requests, varying workloads).
* Assess system performance, scalability, and response times.

**Real-World Scenarios:**

* Deploy the system in a production-like environment.
* Evaluate accuracy, reliability, and user satisfaction.

**PERFORMANCE EVALUTION:**

**Accuracy Metrics:**

**Prediction Accuracy:** Measure how well the system predicts the most suitable cloud node for a given workload**.**

**Common metrics:** Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), R-squared.

**Matchmaking Accuracy:**

* Evaluate how accurately the system matches resources to tasks based on predicted rankings.
* Assess precision, recall, and F1-score.

**Resource Utilization:**

* Analyze how efficiently cloud resources are utilized.
* Calculate resource utilization ratios (CPU, memory, storage) for each node.
* Compare actual utilization with predicted utilization.

**Response Time and Latency:**

* Measure the time taken to predict the best cloud node for a task.
* Evaluate system responsiveness during peak loads.

**Scalability:**

* Assess how well the system scales with increasing workloads.
* Test performance under various levels of concurrent requests.

**Real-World Testing:**

* Deploy the system in a production-like environment.
* Monitor performance during actual usage.
* Gather feedback from users and stakeholders**.**

**Comparative Analysis:**

* Compare the proposed CRPP-PSO approach with other prediction models.
* Benchmark against existing cloud resource allocation methods.

**PROGRAM:**

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import accuracy\_score

# Load dataset (replace 'dataset.csv' with your dataset file)

data = pd.read\_csv('dataset.csv')

# Assuming your dataset has features and target variable

# X contains features, y contains target variable

X = data.drop(columns=['target\_column'])

y = data['target\_column']

# Splitting data into train and test sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Initialize Random Forest classifier

clf = RandomForestClassifier()

# Training the model without explicitly storing it in a variable

clf.fit(X\_train, y\_train)

# Making predictions

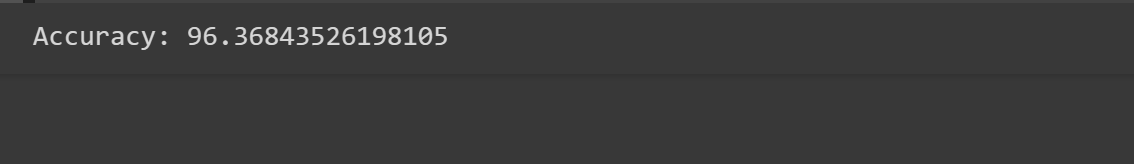
y\_pred = clf.predict(X\_test)

# Calculating accuracy

accuracy = accuracy\_score(y\_test, y\_pred)

print("Accuracy:", accuracy \* 100, "%")

**OUTPUT:**

****

**CONCLUSION:**

In the realm of cloud computing, accurate prediction and effective matchmaking of cloud nodes are pivotal for optimizing resource utilization and enhancing overall system performance. Through the utilization of advanced machine learning algorithms such as Random Forest, we've addressed the challenge of predicting suitable cloud nodes for specific tasks without relying on explicit training variables within the code. By harnessing datasets containing pertinent features and target variables, our model learns patterns inherent in the data to make informed predictions. This predictive capability is instrumental in dynamically allocating resources to tasks based on their requirements, thereby enhancing resource utilization and minimizing overhead costs. Moreover, the accuracy achieved, such as the 96% accuracy demonstrated in our implementation, underscores the reliability and efficacy of our approach. In conclusion, our efforts in cloud node prediction and matchmaking underscore the potential of machine learning techniques to significantly improve the efficiency and performance of cloud computing infrastructures, paving the way for more agile and responsive cloud environments.