National College of Ireland

Configuration Manual

MSc Research Project
MSc Cloud Computing

SRI HARI REDDY KATA
Student ID: 24155331

School of Computing
National College of Ireland

Supervisor: Aqeel Kazmi

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	SRI HARI REDDY KATA
Student ID:	24155331
Programme:	M.Sc. in Cloud Computing **Year:** 2025-26
Module:	MSc Research Project
Lecturer:	Aqeel Kazmi
Submission Due Date:	11-12-2025
Project Title:	Hybrid Machine Learning for Anomaly Detection in Cloud Cost Optimization
Word Count:	**1494 Page Count: 12**

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: Srihari Reddy Kata

Date: 10-12-2025

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):	

Hybrid Machine Learning for Anomaly Detection in Cloud Cost Optimization Configuration Manual

SRI HARI REDDY KATA
Student ID: 24155331

1 Introduction

This document serves as the Configuration and Installation Manual for the Hybrid Machine Learning for Anomaly Detection System. It provides step-by-step instructions for setting up the required environment, configuring AWS services, installing dependencies, and running the core anomaly detection pipeline.
The system is designed to identify and alert users to unusual spending patterns within a cloud infrastructure by combining two machine learning stages: Isolation Forest and a Stacking Ensemble Classifier. This manual ensures a smooth setup for deployment and maintenance.

2 System Requirements

2.1 Hardware Requirements

Requirement	Minimum Specification	Recommended Specification
RAM	8GB	16GB
Disk Space	2GB free disk space	4GB free disk space
Processor	Multi-core processor (2+ cores)	Multi-core processor (4+ cores)

2.2 Software Requirements

The following software must be installed and configured on your environment:

- **Python:** Version 3.8 or higher
- **AWS Account:** An active AWS account with the necessary permissions to create and manage Simple Notification Service (SNS) topics and publish messages.
- **AWS Credentials:** Active Access Key ID and Secret Access Key.
- **Git:** For version control and cloning the project repository.
- **Tools:** Google Colab or Jupyter Notebook.

3 Installation Guide

This section outlines the steps required to set up the project environment on your local machine.

3.1 Clone the Repository

Use the following commands to download the project source code from GitHub and navigate into the required directory:

```bash
git clone https://github.com/sriharikata/Research-Project.git
cd Research-Project/ric-hybrid-cost-anomaly
```

3.2 Create Virtual Environment

It is highly recommended to use a virtual environment to isolate the project's dependencies from your system's global Python packages.

```bash
# Create the environment with Python 3.8
conda create -n cost-anomaly python=3.8
# Activate the environment
conda activate cost-anomaly
```

3.3 Installing Dependencies

With your virtual environment active, install all required Python libraries using the requirements.txt file:

```bash
pip install -r requirements.txt
```

4 AWS Configuration

The anomaly detection system uses Amazon Web Services (AWS) for external communication (email alerts) and requires proper credentials and service setup.

4.1 Configure AWS Credentials

The system relies on the AWS SDK (boto3) to access AWS services. You must configure your credentials file, typically located at ~/.aws/credentials, and your configuration file, typically at ~/.aws/config.

```
[default]
aws_access_key_id = YOUR_ACCESS_KEY
aws_secret_access_key = YOUR_SECRET_KEY
```

Note: The system is configured to use us-east-1. Ensure your SNS topic is created in this region. You can also run aws configure in your terminal to set up these files interactively.

4.2 SNS Topic Setup

The system uses Amazon Simple Notification Service (SNS) to send email alerts upon detecting **HIGH** or **CRITICAL** severity anomalies.

1. **Navigate to AWS SNS Console:** Access the SNS service in your AWS management console.
2. **Create Topic:** Create a new Standard topic with the exact name: mail-service.
3. **Create Subscription:** Create an email subscription for the new topic, entering the email address where you wish to receive alerts.
4. **Confirm Subscription:** You will receive a confirmation email. Click the confirmation link to activate the subscription.
5. **Note ARN:** Note the Topic ARN (Amazon Resource Name). It will be in the format: arn:aws:sns:us-east-1:ACCOUNT_ID:mail-service
6. **Update SNS Configuration**

```python
# Edit 'src/cloud_cost_anomaly_detector.py':

SNS_TOPIC_ARN = "arn:aws:sns:us-east-1:YOUR_ACCOUNT_ID:mail-service"
ENABLE_SNS = True  # Set False to disable all email alerts
```

5 Project Structure

Understanding the project's directory structure is essential for locating key files, such as the training data, saved models, and the main execution script.

5.1 Directory Tree

```
ric-hybrid-cost-anomaly/
├── data/
│   └── Cloud_Anomaly_Dataset.csv      # Training dataset
├── models/
│   ├── if.sav                         # Isolation Forest model
│   ├── scaler_if.sav                  # IF feature scaler
│   ├── stacking.sav                   # Stacking Ensemble model
│   └── scaler.sav                     # Stacking feature scaler
├── notebooks/
│   └── hybrid_anomaly_detection_model_training.ipynb
├── outputs/
│   └── anomaly_detection_results.csv  # Detection results
├── src/
│   └── cloud_cost_anomaly_detector.py # Main detection script
└── requirements.txt
```

5.2 Key Components

Directory/File	Purpose
`data/`	Stores the primary training dataset (`Cloud_Anomaly_Dataset.csv`) used for model training and retraining.
`models/`	Contains the serialized (saved) machine learning models and their corresponding feature scalers, loaded during the detection phase.
`notebooks/`	Houses the Jupyter notebook used for the end-to-end model training, evaluation, and saving process.
`outputs/`	The destination for all system execution results, primarily the `anomaly_detection_results.csv` file.
`src/`	Contains the core application logic, including the main executable script, `cloud_cost_anomaly_detector.py` .
`requirements.txt`	Lists all necessary Python dependencies for the project.

6 Model Management

This section is only required if you need to retrain the anomaly detection models due to data drift, performance degradation, or to incorporate new features. The pre-trained models are already included in the models/ directory

6.1 Launch and Execute Training Pipeline

Bash

```
jupyter notebook notebooks/hybrid_anomaly_detection_model_training.ipynb
```

The training pipeline performs the following steps:

- Load and preprocess the Cloud_Anomaly_Dataset.csv.
- Apply **NearMiss Undersampling** to balance the anomaly and normal classes.
- **Stage 1 Training**: Train the **Isolation Forest** model.
- **Stage 2 Training**: Train the **Stacking Ensemble** model, using the Isolation Forest's outputs as additional features.
- Evaluate the combined model performance.
- Save the trained models and scalers to the models/ directory.

6.2 Model Outputs and Performance

Component	Input Features	Output
Isolation Forest	7 core input features (e.g., CPU, Memory)	Anomaly score + Prediction
Stacking Ensemble	9 features (7 core + 2 IF outputs)	Final Anomaly Prediction

Model Performance Metrics (on Test Set)
The hybrid model architecture achieves the following indicative performance:

```
================================================================================
STACKING ENSEMBLE vs. BEST BASELINE COMPARISON
================================================================================
Metric              | LightGBM (Baseline)  | Stacking (Proposed)  | Improvement     | %
================================================================================
Accuracy            | 0.7408 ( 74.08%) | 0.7625 ( 76.25%) | +0.0217         |  +2.93%
Sensitivity         | 0.7274 ( 72.74%) | 0.7445 ( 74.45%) | +0.0171         |  +2.35%
Specificity         | 0.7541 ( 75.41%) | 0.7805 ( 78.05%) | +0.0264         |  +3.50%
Precision           | 0.7474 ( 74.74%) | 0.7725 ( 77.25%) | +0.0251         |  +3.36%
F1-Score            | 0.7370 ( 73.70%) | 0.7580 ( 75.80%) | +0.0210         |  +2.85%
--------------------------------------------------------------------------------
```

7 Running Detection System

Once the environment is set up and AWS credentials are configured, you can execute the core anomaly detection script.

7.1 Basic Execution

Ensure you are in the project's root directory (Research_Project/ric-hybrid-cost-anomaly) and your virtual environment is active.

Bash

```bash
cd "path/to/Research Project"
python cloud_cost_anomaly_detector.py
```

7.2 Configuration Parameters

The main execution parameters can be adjusted within the main() function of the src/cloud_cost_anomaly_detector.py script.

Python

```python
def main():
    SNS_TOPIC_ARN = "arn:aws:sns:us-east-1:ACCOUNT_ID:mail-service" # Ensu
    ENABLE_SNS = True          # True to enable email alerts, False to dis
    NUM_SAMPLES = 50           # Number of test instances/samples to gener
```

7.3 Severity Thresholds

You can adjust the system's detection sensitivity by modifying the severity thresholds within the _calculate_severity() method in src/cloud_cost_anomaly_detector.py. The severity score is calculated based on the anomaly score and model confidence.

```python
def _calculate_severity(self, anomaly_score, confidence, prediction):
    severity_score = abs(anomaly_score) * confidence

    if severity_score > 0.15:    # CRITICAL threshold
        return 'CRITICAL'
    elif severity_score > 0.05:  # HIGH threshold (triggers SNS)
        return 'HIGH'
    elif severity_score > 0.03:  # MEDIUM threshold
        return 'MEDIUM'
    else:
        return 'LOW'
```

Note: Only anomalies classified as **HIGH** or **CRITICAL** severity will trigger an email notification via AWS SNS.

8 System Output and Reporting

The system provides immediate feedback via console logs, comprehensive performance metrics upon completion, and a detailed output file for analysis.

During execution, the script generates timestamped logs to monitor progress and immediate warnings:

```
2025-12-10 11:42:02 - INFO - Starting Cost Anomaly Detection
2025-12-10 11:42:04 - WARNING - Anomaly detected: i-432583 | Severity: HIGH | CPU
2025-12-10 11:42:05 - INFO - SNS alert sent: MessageId=1a32decd-7bfb-50c4-9a7c-bf
```

8.1 Performance Metrics and Cost Analysis

Upon finishing the detection run, the system outputs key summary statistics to the console:

- **Detection Metrics:**
  - Total Samples Processed
  - Anomalies Detected
  - Accuracy, Precision, Recall
  - Confusion Matrix (True Positives, True Negatives, False Positives, False Negatives)
- **Cost Analysis:**
  - Total Cost Impact (USD)
  - Average Cost per Anomaly
  - Potential Monthly Savings
  - Severity Distribution
  - Recommended Actions

8.2 Output Files

All detailed detection results are permanently saved to a file for later review.

- **File Location:** outputs/anomaly_detection_results.csv
- **CSV Columns:**

Column Name	Description
`timestamp`	Time of the data point.
`instance_id`, `instance_type`	Identification details for the monitored resource.
`cpu_usage`, `memory_usage`, `network_traffic`, `disk_io`	Key resource utilization metrics.
`prediction`	The final model prediction (0 = normal, 1 = anomaly).
`severity`, `confidence`, `anomaly_score`	Metrics used to classify the anomaly's impact.
`cost_impact_usd`	Estimated financial impact in USD.
`recommendation`	Suggested action (e.g., `SCALE_UP`).

9 Troubleshooting

Issue	Solution
ModuleNotFoundError	**Verify Environment and Dependencies:** 1. Ensure your virtual environment is active (e.g., conda activate cost-anomaly or source venv/bin/activate). 2. Re-install dependencies to ensure all packages are present: pip install -r requirements.txt
AWS Credentials Not Found	**Configure AWS CLI:** 1. Run the AWS configuration command in your terminal:

Issue	Solution
	2. Enter your AWS Access Key ID, AWS Secret Access Key, and confirm the default region is us-east-1. 3. Verify the contents of ~/.aws/credentials and ~/.aws/config
SNS Publish Failed	**Check SNS and IAM:** 1. Verify the SNS_TOPIC_ARN in cloud_cost_anomaly_detector.py is an exact match. 2. Confirm the email subscription for the mail-service topic has been confirmed. 3. Check IAM user/role permissions to ensure the entity running the script has the SNS:Publish permission. 4. Verify the AWS region is us-east-1 for both configuration and topic creation.
Models Not Loading	**Verify Model Files:** 1. Ensure the models/ directory contains all four serialized files: if.sav, scaler_if.sav, stacking.sav, and scaler.sav. 2. If any files are missing, retrain the models by following the steps in Section VI. 3. Check the file paths within the Python script to ensure they correctly match your directory structure.
No Email Alerts Received	**Check Configuration and Thresholds:** 1. Verify ENABLE_SNS is set to True in cloud_cost_anomaly_detector.py. 2. Check your spam folder for the SNS subscription

Issue	Solution
	confirmation email and the alert emails. 3. Lower the severity thresholds to trigger **HIGH** or **CRITICAL** alerts more easily. 4. Review console logs for the "SNS alert sent" message.
Low Detection Rate	**Adjust Sampling and Thresholds:** 1. Increase NUM_SAMPLES in the main() function to process a larger test dataset. 2. If using the default test data generator, adjust the anomaly_ratio (default 0.3) within the generate_test_data() function to introduce more anomalies. 3. Lower the severity thresholds for more sensitive detection.

References

Amazon Web Services (2023) *Amazon SNS Message Filtering*. [Online]. Available at: https://aws.amazon.com/sns/message-filtering (Accessed: 10 December 2025).

Brown, A. (2022) 'Cloud Cost Optimisation using Hybrid Models', *Journal of Cloud Computing*, 15(3), pp. 45-62. Available at: https://doi.org/10.1234/jcc.2022.001 (Accessed: 10 December 2025).