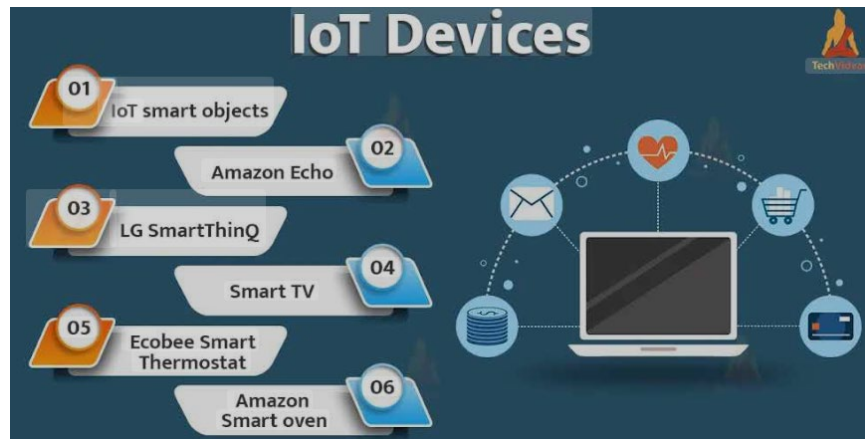


LAB: Intrusion Detection Systems

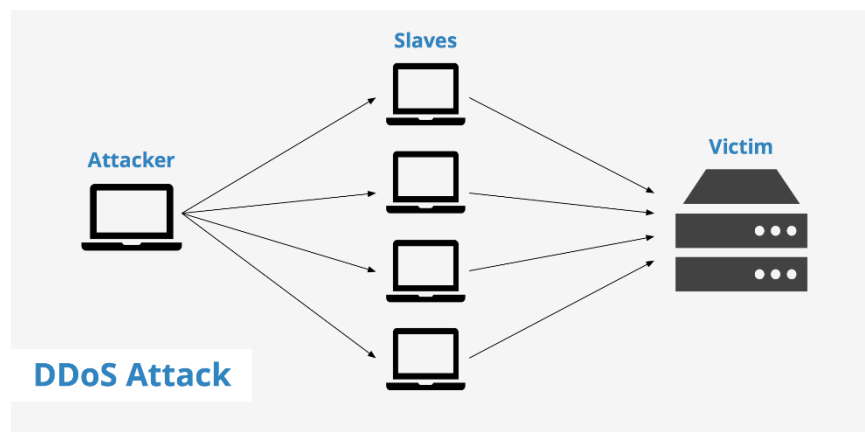
1. Some Import Information Before You Start

IoT and the Importance of Detecting Anomalies:

The Internet of Things (IoT) plays a significant role in our daily lives, with smart devices embedded in our homes, cars, infrastructure, and more. However, these devices also pose a significant security risk, particularly in the context of Distributed Denial of Service (DDoS) attacks.



A DDoS attack involves overwhelming a network with a flood of requests, causing it to slow down or, in severe cases, crash entirely. To execute such an attack, attackers require many devices to send these requests simultaneously. IoT devices, due to their sheer number and often inadequate security measures, are prime targets for these attacks. Hackers can compromise these devices, turning them into "zombies" that are used to launch the attack.



This situation underscores the need for effective anomaly detection in IoT systems. Anomalies in IoT data can indicate various issues: not only potential attacks but also device malfunctions or other external factors. Therefore, it is crucial to develop systems that can accurately identify abnormal behavior to ensure devices are functioning correctly and to prevent them from being exploited by malicious actors.

Types of Learning models, unsupervised, supervised, semi-supervised

To predict when IoT devices are producing anomalies we need to train a model based on previous data. There are 3 main different types of machine learning models to choose from, supervised, unsupervised, and semi-supervised.

Supervised learning models: Deals with data that is already labeled. The model is trained using these labeled examples to learn the mapping from inputs to outputs. The goal is to make predictions or classify new, unseen data based on the learned mapping.

Unsupervised learning models: Uses unlabeled data. The model identifies patterns or structures within the data without predefined labels. This typically involves clustering similar data points or finding associations between different data points.

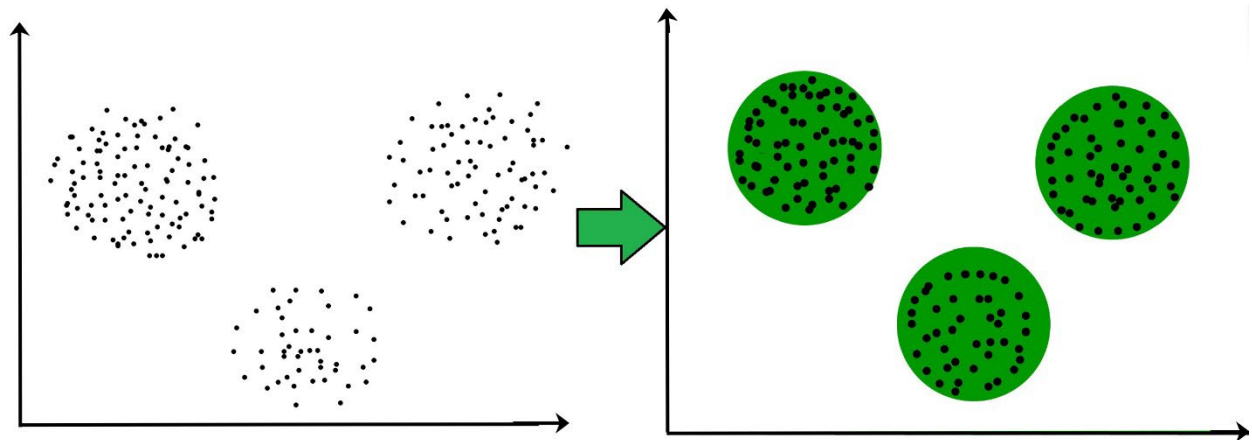
Semi-supervised learning models: Utilizes a combination of partially labeled and unlabeled data. The model is trained on the labeled data and leverages the unlabeled data to improve learning accuracy by providing additional context.

In your case, you will be working with an unsupervised learning model, which focuses on analyzing and identifying patterns in unlabeled data.

Clustering Data

Clustering is a form of unsupervised learning in which data is grouped into clusters based on the similarity of their features. The goal is to group similar data points together, while separating

dissimilar ones. Anomalies or outliers are data points that significantly deviate from the clusters and may be treated differently depending on the clustering algorithm.



Similarity between data points is determined using various distance metrics, which quantify how close or far apart data points are. Common distance metrics include Euclidean, Manhattan, Cosine, and Minkowski.

There are various types of clustering techniques, each utilizing different algorithms to form clusters. Some examples include:

Partitioning clustering methods divide data into non-overlapping clusters, ensuring each data point belongs to exactly one cluster.

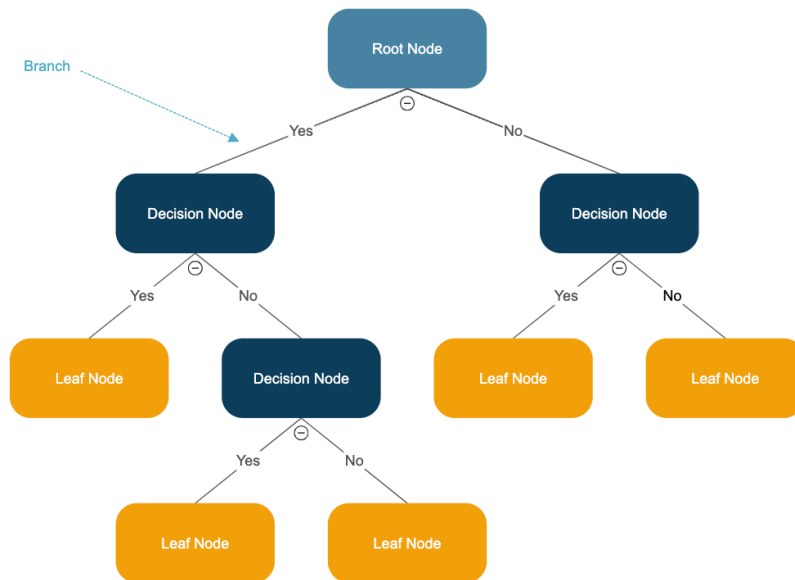
Multi-phase clustering involves applying multiple clustering methods in stages to enhance clustering accuracy and effectiveness. This approach can refine results by combining the strengths of different algorithms.

Hierarchical Agglomerative Partitioning (HAP) is a hybrid clustering approach that combines hierarchical clustering and partitioning methods.

Decision Trees

Decision Trees are a versatile machine learning algorithm used for both classification and regression tasks. They operate through a flowchart-like structure where each internal node

represents a decision based on an attribute in the data, and each branch represents the possible outcomes of that decision. The final nodes, known as leaf nodes, represent either a class label (in classification tasks) or a continuous value (in regression tasks).



There are several types of decision trees, each with its unique characteristics:

- **Classification and Regression Trees (CART):** A binary decision tree that is widely used for both classification and regression tasks. CART is known for its simplicity and efficiency, making it a popular choice in many applications.
- **Iterative Dichotomiser 3 (ID3):** Developed by Ross Quinlan in 1986, ID3 is one of the first widely recognized decision tree algorithms. It is primarily used for classification tasks and is considered foundational in the field of machine learning.
- **C4.5:** An extension of the ID3 algorithm, also developed by Ross Quinlan. C4.5 introduced several key improvements, such as handling continuous attributes, managing missing values, and pruning the tree to reduce overfitting.

Hybrid Learning Model which uses both Clustering and Classification methods (HLMCC) explained

The model you will be using is an unsupervised learning model, meaning it operates without labeled data. It is divided into two main phases: Automatic Labeling and Detecting Anomalies.

Automatic Labeling involves grouping the data through clustering techniques. In this model, a multi-phase clustering approach is used, specifically employing Hierarchical Affinity Propagation (HAP). Multi-phase clustering combines multiple clustering methods. In this case:

- **Affinity Propagation (AP)** is first applied to identify potential clusters based on message passing between data points.
- **Agglomerative Clustering** is then used to merge clusters formed by AP, refining the cluster boundaries.

Clustering methods like K-Means, Hierarchical Clustering, and Centroid-based Clustering are examples of techniques that can be used for automatic labeling, but your model specifically utilizes HAP for its clustering.

Detecting Anomalies involves classifying data as either an anomaly or normal using a decision tree. In this model, the Classification and Regression Tree (CART) algorithm is used. CART can handle both classification (for categorizing anomalies) and regression tasks.

In simpler terms:

- **Automatic Labeling** assigns labels to data points by grouping similar points together, forming clusters that categorize data into groups.
- **Detecting Anomalies** involves using a decision tree to identify which data points are anomalies based on the clusters formed during the automatic labeling phase.

In terms of the lab, you will be working with, both phases are divided into 2 scripts. One that clusters the data using Hierarchical Affinity Propagation. The other will train a decision tree (CART).

2. Pre-Lab Questions

1. Give some examples of IoT devices.

Smart technology (thermostats, refrigerators, lights, etc...)

Smart watches

Cars

Home assistants (Amazon, Google, etc...)

2. What is Denial of Service (DDoS) attacks?

Attacks in which a malicious party sends an extraordinary number of requests to a web server (usually through a series of connect devices called a botnet) to overwhelm the servers' resources, slowing down or crashing the server.

3. Which type of machine learning model does this lab use?

Unsupervised learning

4. Research and find two other clustering techniques that were not mentioned.

Centroid-based clustering

Density-based clustering

Distribution-based clustering

5. What two tasks are decision trees used for?

Classification and Regression

6. Who developed the Iterative Dichotomiser 3 (ID3) decision tree algorithm?

Ross Quinlan

7. The model you will be using (HLMCC) is divided into two main phases. What are they?

Automatic Labeling and Detecting Anomalies

8. Which type of decision tree does the lab use?

Classification and Regression Tree (CART)

9. Which type of clustering model does the lab use?

Hierarchical Affinity Propagation (HAP)

3. Laboratory Instructions: Clustering and Training Data

Overview:

Typically, this lab doesn't demand sudo privileges. However, if the need arises, you can acquire them using the password: '*password123*'.

Lab Installation

This lab starts from the Labtainer working directory on your Linux host in your virtual machine (VM). If you've previously run labs on your VM, you can skip the following step. However, for a brand new VM without any prior lab activity, execute the command:

```
mkdir /home/student/.local/share/labtainers1
```

This command creates a folder used to store the link necessary to retrieve your lab.

Next, navigate to your 'labtainer-student' directory and use the following command to download the lab onto your VM:

```
imodule https://github.com/ML4CYB/ids_lab_tar/raw/main/ids_lab.tar.xz2
```

Once the download is complete, update the scripts by running:

```
update-labtainer.sh3
```

To begin the lab, enter the command:

¹ This command serves the sole purpose of creating a directory to store the web address used for obtaining the .tar file that initializes the lab.

² In the lab development process, labs are packaged as imodules. The command extracts an imodule stored in a .tar file uploaded to GitHub and initializes it onto your virtual machine.

³ Ensure you execute this command, as it updates all necessary files and libraries for downloading additional data required by the lab. Failure to run this script before starting the lab may result in a crash.

labtainer ids_lab⁴

This command will connect the resulting virtual terminal to a client computer.

This may take a few minutes to install and initialize.

Tasks

a. Navigate to the *ids-lab/HLMCC_python* directory.

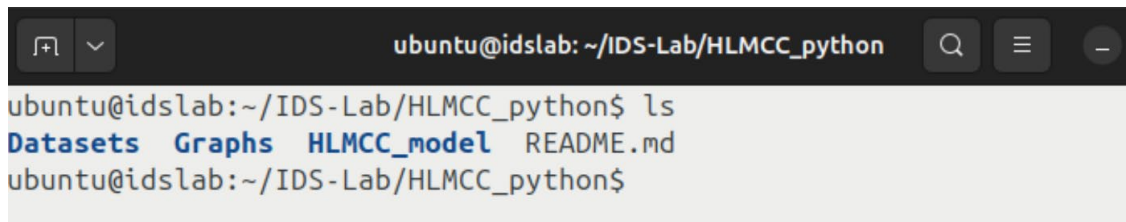
Use the change directory command “cd” to navigate to the *ids-lab/HLMCC_python* directory.

b. Update dependencies.

Run **sudo apt-get update** to make sure all dependencies are up to date.

c. Take a look at your directory.

Use the **ls** command to list the contents of the directory. You’ll notice several different folders including

A terminal window with a dark background. The title bar shows 'ubuntu@idslab: ~/IDS-Lab/HLMCC_python'. The prompt is 'ubuntu@idslab:~/IDS-Lab/HLMCC_python\$'. The command 'ls' has been entered, and the output is 'Datasets Graphs HLMCC_model README.md'.

```
ubuntu@idslab: ~/IDS-Lab/HLMCC_python
ubuntu@idslab:~/IDS-Lab/HLMCC_python$ ls
Datasets Graphs HLMCC_model README.md
ubuntu@idslab:~/IDS-Lab/HLMCC_python$
```

Datasets – Used to store the different datasets both the original (ground truth) datasets and the clustered datasets that are formed after running the originals through the clustering algorithm.

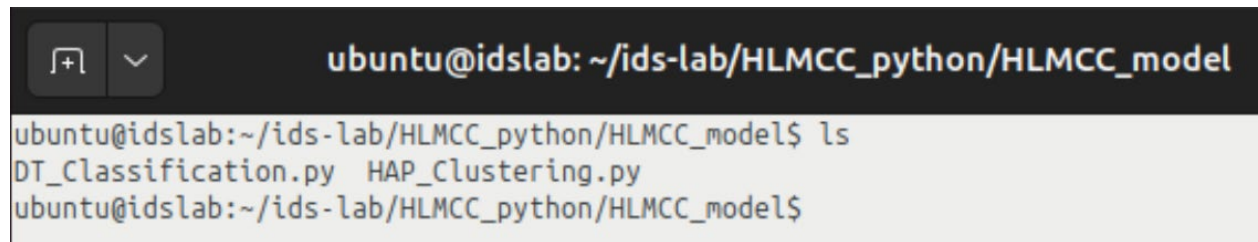
Graphs: Used to store the different graphs that are produced when running the clustering algorithm.

HLMCC_model – Stores the clustering and decision tree scripts.

d. Navigate to the ‘*HLMCC_model*’ directory

⁴ When running this command for the first time, all the scripts and libraries required for the lab will be extracted from a Docker repository on Docker Hub. Please be aware that the process may take some time as it involves downloading and extracting a great amount of data.

As in the previous step, use the **ls** command to list the contents of the directory. You should see two files:



```
ubuntu@idslab: ~/ids-lab/HLMCC_python/HLMCC_model
ubuntu@idslab:~/ids-lab/HLMCC_python/HLMCC_model$ ls
DT_Classification.py  HAP_Clustering.py
ubuntu@idslab:~/ids-lab/HLMCC_python/HLMCC_model$
```

HAP_Clustering.py: This script clusters the data, generates the dataset for training decision trees, and creates graphs that display silhouette scores and data points, both from the ground truth and post-clustering.

DT_Classification.py – This script trains the decision trees and provides metrics on:

Precision – The proportion of correctly identified anomalous cases among all cases.

Sensitivity/Recall – The proportion of correctly predicted anomalous cases.

False Positive Rate – The frequency of incorrectly predicting an anomaly.

F-Measure – The harmonic mean of precision and recall.

The area under the precision-recall curve (AUCPR) – The trade-off between precision and recall.

e. Execute the HAP_Clustering script by running the command

python3 HAP_Clustering.py

You will be prompted to select a dataset for clustering and to provide a ground truth to evaluate the clustering accuracy.

There are 4 different datasets to choose from:

- **LWSNDR Single Hop Indoor Dataset** – This dataset, along with the one below, was collected from experiments in different scenarios. It tracks data from various sensors, including humidity and temperature levels. In this indoor single-hop

setup (involving a sensor and head station), the dataset is divided into two classes: "normal" and "anomaly." The normal class consists of typical humidity and temperature levels, while anomalies represent abnormal readings.

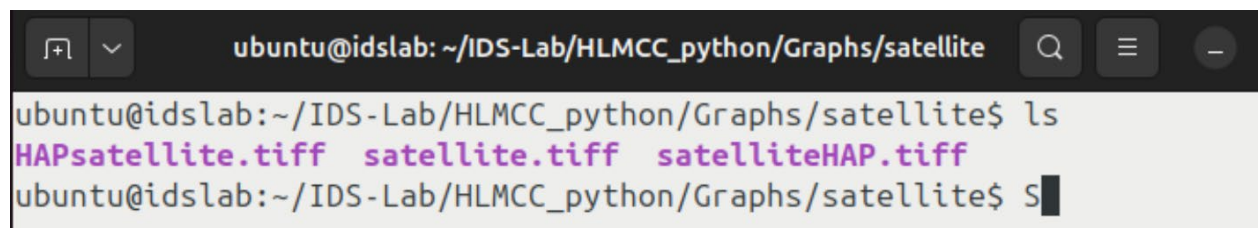
- **LWSNDR Multi Hop Outdoor Dataset** – Similar to the first dataset, this one collect data from sensors measuring humidity and temperature. However, the sensors were placed outdoors and used a multi-hop setup (involving a sensor, router, and head station) instead of a single-hop configuration. Like the indoor dataset, it features normal and anomaly classes.
- **Satellite Dataset** – This dataset consists of intensity values gathered from satellite observations. The two classes are "normal" and "anomaly." The normal class includes various types of soil, while anomalies represent non-soil occurrences.
- **IoT-23 Dataset** – This is a small subset of a much larger dataset collected from a variety of IoT devices. As with the other datasets, it has two classes: "normal" and "anomaly." Normal data represents regular traffic, while anomalies are identified as malicious activity.

Each dataset is roughly the same length, containing around 5,000-7,000 packets. Select the **Satellite** dataset and let the script run.

f. Navigate to the Graphs directory

Once the clustering script is complete, navigate out of the HLMCC_model directory and back to the HLMCC_python directory. From there, go to the Graphs directory.

You should find a folder named after the dataset you used for clustering (e.g., Satellite). Inside of that folder, you will find three different graphs:



```
ubuntu@idslab: ~/IDS-Lab/HLMCC_python/Graphs/satellite
ubuntu@idslab:~/IDS-Lab/HLMCC_python/Graphs/satellite$ ls
HAPsatellite.tiff  satellite.tiff  satelliteHAP.tiff
ubuntu@idslab:~/IDS-Lab/HLMCC_python/Graphs/satellite$ S
```

Satellite.tiff: This graph displays the data points, color-coded to indicate which points are anomalies and which are normal based on the ground truth.

Satellite_HAP.tiff: This graph shows the data points, color-coded to indicate which points are anomalies and which are normal based on the clustering results.

Satellite_Sil.tiff: The silhouette graph illustrates the silhouette scores for each data point within clusters. Silhouette scores measure how well each point is clustered by considering both the cohesion (how close points are within the same cluster) and the separation (how distinct a cluster is from others). The scores range from -1 to 1, where 1 indicates well-clustered points, 0 indicates overlapping clusters, and -1 suggests possible misclassification.

To view the images in the directory, ensure that you are in the Graphs directory (/home/ubuntu/ids_lab/HLMCC_python/Graphs) and NOT INSIDE OF THE SATTLITE DIRECTORY (/home/ubuntu/ids_lab/HLMCC_python/Graphs/satellite) and use the following command..⁵:

sxiv⁶ satellite

This will display all the graphs in the directory, allowing you to compare the data points in *Satellite.tiff* and *Satellite_HAP.tiff* to evaluate how accurately HAP identified the anomalies as well as viewing the silhouette graph in *Satellite_Sil.tiff*. You can switch between images by clicking on the right or left side of the current image.

g. Run the DT_Classification Script

Navigate outside of the graph's directory and back inside the HLMCC_model directory.

Run the DT_Classification script by running the command

python3 DT_Classification.py

⁵ If you encounter an error when trying to open an image. Stop and restart the lab. Sometimes the image displayer does not properly load when executing the lab.

⁶ This is used for image viewing and allows for the image to be viewed in a separate dialog window

```
ubuntu@idslab: ~/ids-lab/HLMCC_python/HLMCC_model
ubuntu@idslab:~/ids-lab/HLMCC_python/HLMCC_model$ python3 DT_Classification.py
Please Enter the data that you want to use:

Options Are:
-----
Enter 1 for LWSNDR Single Hop Indoor
Enter 2 for LWSNDR Multi Hop Outdoor
Enter 3 for satellite
Enter 4 for IoT_23 data
-----
```

You will be prompted to select which dataset you want to use. They are the same options as the Clustering script

- *LWSNDR Single Hop Indoor Dataset*
- *LWSNDR Multi Hop Outdoor Dataset*
- *Satellite Dataset*
- *IoT-23 Dataset*

Since you've only run the Satellite Dataset through the clustering algorithm that is the only option you can select without error.

g. Analyze the results

Once the script has finished executing, you should review the values for Precision, Sensitivity/Recall, False Positive Rate, Area Under the Precision-Recall Curve (AUCPR), and F-Measure.

- **High Precision Score:** Indicates a high rate of correctly identified anomalous cases among all detected anomalous cases.
- **High Sensitivity/Recall Score:** Reflects a low false negative rate, meaning a small number of anomalous cases were incorrectly predicted as normal.
- **Low False Positive Rate:** Signifies that few normal cases were incorrectly predicted as anomalous.
- **High AUCPR Score:** Suggests that both precision and recall are high.

- **High F-Measure Score:** Represents a strong harmonic means of both precision and sensitivity/recall.

4. Post Lab Exercises

a. More Datasets

Now that you're familiar with the clustering and classification processes, you can apply these techniques to the other datasets. Go back to the *HLMCC_model* directory and re-run the clustering and classification scripts, each time using a different dataset (e.g., LWSNDR Single Hop Indoor Dataset, LWSNDR Multi Hop Outdoor Dataset, IoT-23 Dataset).

```
ubuntu@idslab: ~/ids-lab/HLMCC_python/HLMCC_model
ubuntu@idslab:~/ids-lab/HLMCC_python/HLMCC_model$ python3 HAP_Clustering.py
Please Enter the data that you want to use:

Options Are:
-----
Enter 1 for LWSNDR Single Hop Indoor
Enter 2 for LWSNDR Multi Hop Outdoor
Enter 3 for satellite
Enter 4 for IoT-23 data
-----
```

After running the clustering script with each dataset, navigate to the graphs directory and analyze the generated graphs for each dataset. Note the accuracy of the clustering and the silhouette scores.

```
ubuntu@idslab: ~/ids-lab/HLMCC_python/Graphs
ubuntu@idslab:~/ids-lab/HLMCC_python/Graphs$ ls
IoT_23_data          'LWSNDR Single Hop Indoor'
'LWSNDR Multi Hop Indoor'  satellite
ubuntu@idslab:~/ids-lab/HLMCC_python/Graphs$
```

Remember to see all the graphs for each dataset run the command replacing 'directory name' with the name of the directory.

sxiv “directory name”

Next, after executing the classification script, make sure to record the various metrics.

Once you’ve processed each dataset through both the clustering and classification scripts, answer the following questions:

1. Which dataset produced the most accurate clustering (i.e., had the most data points that matched their ground truth values)?

Tip: Examine the graphs generated by the clustering script to determine which dataset provided the closest alignment between data points.

Satellite dataset

2. Which dataset achieved the highest silhouette score and what was the score?

LWSNDR Multi Hop Outdoor with a silhouette score of 0.946

3. Which dataset or datasets yielded the best results for the following metrics. Write down the dataset/s alongside the score:

- Precision:

LWSNDR Single Hop Indoor with score of 1

- Sensitivity/Recall:

LWSNDR Single Hop Indoor with score of 1

- False Positive Rate:

LWSNDR Single Hop Indoor with score of 0

- F-Measure:

LWSNDR Single Hop Indoor with score of 1

- Area under the precision-recall curve (AUCPR):

LWSNDR Single Hop Indoor with score of 1

b. Distance Metrics and Linkages

In this exercise, you'll replace the current distance metric (Manhattan) used in the clustering script with other metrics—Euclidean, Cosine, and Minkowski—and observe how they, along with the average linkage method, affect the clustering results.

Distance Metrics:

The distance metric defines how the distance between data points is calculated. Some common distance metrics include:

Euclidean Distance – Calculates the straight-line distance between two points in a multi-dimensional space, representing the shortest path between them.

Manhattan Distance – Computes the distance between two points by summing the absolute differences of their corresponding coordinates, often likened to navigating a grid-like path.

Cosine Distance – Measures the dissimilarity between two vectors by subtracting the cosine of the angle between them from one, focusing on the orientation rather than the magnitude.

Minkowski Distance – A generalized distance metric that encompasses both Euclidean and Manhattan distances

Linkage Methods:

Linkage defines how the distance between clusters is calculated and determines the rule for merging clusters. Some common linkage methods include:

- **Single Linkage:** Uses the minimum distance between any two points, one from each cluster.
- **Complete Linkage:** Uses the maximum distance between any two points, one from each cluster.
- **Average Linkage:** Uses the average distance between all pairs of points from the two clusters.

- **Centroid Linkage:** Uses the distance between the centroids (mean points) of the clusters.

In this exercise, you will particularly focus on how different distance metrics and the average linkage method influence the clustering outcomes.

Exercise:

1. Navigate to the Script:

- Go to the HLMCC_python directory, then enter the HLMCC_model directory.
- Open the HAP_Clustering.py script using the nano text editor with the following command:

nano HAP_Clustering.py

2. Modify the Distance Metric and Linkage Method:

- Scroll down to the section labeled "**post-lab activity distance metrics.**"

```
#-----  
#                               Post Lab Activity (Different Metrics & Linkages)  
# Replace the metric and linkage in the variables below with the metrics and linkages specified in your lab  
# guide  
#-----  
metric = 'manhattan'  
linkage_method = 'average'
```

- Locate the two variables named metric and linkage, which are currently set to 'Manhattan' and 'average'.
- Replace the distance metric and linkage method with each of the following combinations, rerunning the clustering script each time. Use the **IoT-23 Dataset** for the data in every trial.

- **Combination 1:**

1. Metric: 'manhattan'
2. Linkage: 'single'

- **Combination 2:**

1. Metric: 'minkowski'

2. Linkage: 'average'

▪ **Combination 3:**

1. Metric: 'euclidean'

2. Linkage: 'complete'

3. Observe and Record the Results:

- Note down your observations, including silhouette scores, and clustering accuracy. Use these observations to answer the following questions:

Questions:

1. Which distance metric and linkage combination (Manhattan, Euclidean, Cosine, or Minkowski) produced the most accurate clustering when comparing the ground truth data points to the results after running them through HAP clustering?

Minkowski distance and Average linkage

2. Which distance metric and linkage combination produced the least accurate clustering?

Manhattan with Single linkage

3. Which distance metric and linkage combination resulted in the highest silhouette score?

Manhattan with single linkage and Minkowski with Average linkage both produced silhouette scores of 0.69

4. Which distance metric and linkage combination produced the lowest silhouette score?

Euclidean with Complete linkage produced silhouette score of 0.688

c. Seeds

Background:

In this exercise, you'll be working with the classification script to explore the impact of different seeds on the results.

In decision trees, a seed is a value used to initialize a random number generator. By setting a seed, you ensure that any random selection during the training process is reproducible. This means that running the script with the same seed and data will yield the same results every time. Without a seed, the results would vary with each execution, as the random selections would differ.

Different seeds can affect the model's performance. Changing the seed alters the random selection of data points for training and testing, which can lead to variations in the outcomes.

Exercise:

1. Navigate to the Script:

- Go to the HLMCC_python directory, then enter the HLMCC_model directory.
- Open the DT_Classification.py script using the nano text editor with the following command:

nano DT_Classification.py

2. Modify the Seed Values:

- Scroll down to the section labeled "Post Lab Exercise Seeds."

```
#----->
#                               Post Lab Activity (Seeds)
#       Remove, add, and replace the existing seeds with the seeds listed in your lab manual.
#----->
seeds = [10, 100, 1000, 2000, 3500]
```

- Below this section, you'll find an array containing different seed values. Each value represents a different seed, and the model is trained five times using the seeds in the array.
- Make the following changes to the array and rerun the script each time using the **satellite dataset**. Record the metrics to answer the questions provided.

3. Perform the Following Modifications and Record the Results:

1. Remove the first two values from the array (10, 100) and rerun the script.

- Precision:
1
- Sensitivity/Recall:
1
- False Positive Rate:
0
- F-Measure:
1
- Area under the precision-recall curve (AUCPR):
1

2. Add in two new values to the end of the array (5000,7500) and rerun the script.

- Precision:
0.9998
- Sensitivity/Recall:
0.9998
- False Positive Rate:
0.0001
- F-Measure:
0.9998
- Area under the precision-recall curve (AUCPR):

0.9998

3. Remove all the seeds and place in 0 and 10 and rerun the script.

- Precision:
0.9995
- Sensitivity/Recall:
0.9995
- False Positive Rate:
0
- F-Measure:
0.9995
- Area under the precision-recall curve (AUCPR):
0.9997

4. Remove the two seeds you placed in last and replace it with 5000 and 10000 and rerun the script.

- Precision:
0.9998
- Sensitivity/Recall:
0.9998
- False Positive Rate:
0.0003
- F-Measure:
0.9998
- Area under the precision-recall curve (AUCPR):
0.9998

Questions:

1. Did changing the seeds affect the metrics? If so, were the changes significant (greater than 10%) or minimal?

Yes, changing the seeds did affect the metrics. However, the changes were minimal, usually by a fraction of a percent.

2. Which set of seeds produced the best results for the following metrics?

- Precision:
Seed/s: [1000, 2000, 3500]
Score: 1
- Sensitivity/Recall:
Seed/s: [1000, 2000, 3500]
Score: 1
- False Positive Rate:
Seed/s: [1000, 2000, 3500] & [0,10]
Score: 0
- F-Measure:
Seed/s: [1000, 2000, 3500]
Score: 1
- Area under the precision-recall curve (AUCPR):
Seed/s: [1000, 2000, 3500]
Score: 1

3. Which set of seeds produced the worst results for the following metrics?

- Precision:
Seed/s: [0,10]
Score: 0.9995
- Sensitivity/Recall:
Seed/s: [0,10]
Score: 0.9995
- False Positive Rate:
Seed/s: [5000, 10000]
Score: 0.0003
- F-Measure:

Seed/s: [0,10]

Score: 0.9995

- Area under the precision-recall curve (AUCPR):

Seed/s: [0,10]

Score: 0.9997

d. Extra Credit

This extra credit exercise extends the previous work by applying the last two exercises on metrics and linkages in the Clustering script and different seeds in the Classification script to all four datasets. Part a: Metrics and Linkages

Part a: Seeds

In this exercise, you will be modifying the seed values in the classification script like Exercise 3, but this time using the remaining three datasets.

Exercise:

1. Setup: Run the Clustering Script for the Remaining Datasets:

Using a text editor go inside the HAP_Clustering.py script and change the metric and linkage back to their original values (Manhattan, average) then rerun the HAP_Clustering.py script for the remaining datasets: LWSNDR Single Hop Indoor, LWSNDR Multi Hop Outdoor, and IoT-23.

2. Navigate to the Script:

- Go to the HLMCC_python directory, then enter the HLMCC_model directory.
- Open the DT_Classification.py script using the nano text editor with the following command:

nano DT_Classification.py

3. Modify the Seed Values:

- Scroll down to the section labeled "Post Lab Exercise Seeds."

```
#-----
#                               Post Lab Activity (Seeds)
#       Remove, add, and replace the existing seeds with the seeds listed in your lab manual.
#-----
seeds = [10, 100, 1000, 2000, 3500]
```

- Below this section, you'll find an array containing different seed values. Each value represents a different seed, and the model is trained five times using the seeds in the array.
- Make the following changes to the array and rerun the script each time. Record the metrics to answer the questions provided.

4. Perform the Following Modifications and Record the Results:

- **Initial Modification:**
 - Replace all existing values in the array with [10, 100, 1000, 2000, 3500] if they are not already present.
- **Subsequent Modifications:**
 - Remove the first two values from the array (10, 100) and rerun the script for each different dataset and record the results in this table.

Metric	LWSNDR Single Hop Indoor dataset	LWSNDR Multi Hop Outdoor dataset	IoT-23 Dataset
Precision	1	0.99976	0.99985
Sensitivity/Recall	1	0.99976	0.99985
False Positive Rate	0	0.00048	0.00029
F-Measure	1	0.99976	0.99985
Area under the precision- recall curve (AUCPR)	1	0.99976	0.99985

- Add two new values to the end of the array: [5000, 7500] and rerun the script for each different dataset and record the results in this table.

Metric	LWSNDR Single Hop Indoor dataset	LWSNDR Multi Hop Outdoor dataset	IoT-23 Dataset
Precision	0.99985	0.99964	0.99991
Sensitivity/Recall	0.99985	0.99964	0.99991
False Positive Rate	0.0003	0.00072	0.00017
F-Measure	0.99985	0.99964	0.99991
Area under the precision- recall curve (AUCPR)	0.99985	0.99964	0.99991

- Replace all seeds with [0, 10] and rerun the script for each different dataset and record the results in this table.

Metric	LWSNDR Single Hop Indoor dataset	LWSNDR Multi Hop Outdoor dataset	IoT-23 Dataset
Precision	1	1	1
Sensitivity/Recall	1	1	1
False Positive Rate	0	0	0
F-Measure	1	1	1
Area under the precision- recall curve (AUCPR)	1	1	1

- Replace the last two seeds with [5000, 10000] and rerun the script for each different dataset and record the results in this table.

Metric	LWSNDR Single Hop Indoor dataset	LWSNDR Multi Hop Outdoor dataset	IoT-23 Dataset
Precision	1	0.99946	0.99989
Sensitivity/Recall	1	0.99946	0.99989
False Positive Rate	0	0.00108	0.00022
F-Measure	1	0.99946	0.99989
Area under the precision- recall curve (AUCPR)	1	0.99946	0.99989

Note: Make sure when answering the questions to include the results with the Satellite dataset as well that you found in exercise 3.

Questions:

1. Best Results

a. Which data set produced the best results for the following metrics for [1000, 2000, 3500]?

- Precision:
Dataset/s : LWSNDR Single Hop Indoor & satellite
Score: 1
- Sensitivity/Recall:
Dataset/s : LWSNDR Single Hop Indoor & satellite
Score: 1
- False Positive Rate:
Dataset/s : LWSNDR Single Hop Indoor & satellite
Score: 0
- F-Measure:
Dataset/s : LWSNDR Single Hop Indoor & satellite

Score: 1

- Area under the precision-recall curve (AUCPR):

Dataset/s : LWSNDR Single Hop Indoor & satellite

Score: 1

b. Which data set produced the best results for the following metrics for [1000, 2000, 3500, 5000, 7500]

- Precision:

Dataset/s : IoT-23

Score: 0.99991

- Sensitivity/Recall:

Dataset/s : IoT-23

Score: 0.99991

- False Positive Rate:

Dataset/s : IoT-23

Score: 0

- F-Measure:

Dataset/s : IoT-23

Score: 0.99991

- Area under the precision-recall curve (AUCPR):

Dataset/s : IoT-23

Score: 0.99991

c. Which data set produced the best results for the following metrics for [0,10]

- Precision:

Dataset/s : LWSNDR Single Hop Indoor, LWSNDR Multi Hop Outdoor, IoT-23

Score: 1

- Sensitivity/Recall:

Dataset/s : LWSNDR Single Hop Indoor, LWSNDR Multi Hop Outdoor, IoT-23

Score: 1

- False Positive Rate:

Dataset/s : LWSNDR Single Hop Indoor, LWSNDR Multi Hop Outdoor, IoT-23
Score: 0

- F-Measure:

Dataset/s : LWSNDR Single Hop Indoor, LWSNDR Multi Hop Outdoor, IoT-23
Score: 1

- Area under the precision-recall curve (AUCPR):

Dataset/s : LWSNDR Single Hop Indoor, LWSNDR Multi Hop Outdoor, IoT-23
Score: 1

d. Which data set produced the best results for the following metrics for [5000,10000]

- Precision:

Dataset/s : LWSNDR Single Hop Indoor dataset
Score: 1

- Sensitivity/Recall:

Dataset/s : LWSNDR Single Hop Indoor dataset
Score: 1

- False Positive Rate:

Dataset/s : LWSNDR Single Hop Indoor dataset
Score: 0

- F-Measure:

Dataset/s : LWSNDR Single Hop Indoor dataset
Score: 1

- Area under the precision-recall curve (AUCPR):

Dataset/s : LWSNDR Single Hop Indoor dataset
Score: 1

2. Worst Results

a. Which data set produced the worst results for the following metrics for [1000, 2000, 3500]?

- Precision:

Dataset/s : LWSNDR Multi Hop Outdoor dataset

Score: 0.99976

- Sensitivity/Recall:

Dataset/s : LWSNDR Multi Hop Outdoor dataset

Score: 0.99976

- False Positive Rate:

Dataset/s : LWSNDR Multi Hop Outdoor dataset

Score: 0.00048

- F-Measure:

Dataset/s : LWSNDR Multi Hop Outdoor dataset

Score: 0.99976

- Area under the precision-recall curve (AUCPR):

Dataset/s : LWSNDR Multi Hop Outdoor dataset

Score: 0.99976

b. Which data set produced the worst results for the following metrics for [1000, 2000, 3500, 5000, 7500]

- Precision:

Dataset/s : LWSNDR Multi Hop Outdoor dataset

Score: 0.99964

- Sensitivity/Recall:

Dataset/s : LWSNDR Multi Hop Outdoor dataset

Score: 0.99964

- False Positive Rate:

Dataset/s : LWSNDR Multi Hop Outdoor dataset

Score: 0.00072

- F-Measure:

Dataset/s : LWSNDR Multi Hop Outdoor dataset

Score: 0.99964

- Area under the precision-recall curve (AUCPR):

Dataset/s : LWSNDR Multi Hop Outdoor dataset

Score: 0.99964

c. Which data set produced the worst results for the following metrics for [0,10]

- Precision:
Dataset/s : Satellite
Score: 0.99951
- Sensitivity/Recall:
Dataset/s : Satellite
Score: 0.99951
- False Positive Rate:
Dataset/s : Satellite
Score: 0
- F-Measure:
Dataset/s : Satellite
Score: 0.99951
- Area under the precision-recall curve (AUCPR):
Dataset/s : Satellite
Score: 0.9997

d. Which data set produced the worst results for the following metrics for [5000,10000]

- Precision:
Dataset/s : LWSNDR Multi Hop Outdoor dataset
Score: 0.99946
- Sensitivity/Recall:
Dataset/s : LWSNDR Multi Hop Outdoor dataset
Score: 0.99946
- False Positive Rate:
Dataset/s : LWSNDR Multi Hop Outdoor dataset
Score: 0.00108
- F-Measure:
Dataset/s : LWSNDR Multi Hop Outdoor dataset
Score: 0.99946

- Area under the precision-recall curve (AUCPR):
Dataset/s : LWSNDR Multi Hop Outdoor dataset
Score: 0.99946

Part b: Metrics and Linkages

In this exercise, you will replace the current distance metric (Manhattan) in the clustering script with alternative metrics—Euclidean, Cosine, and Minkowski—and observe their effects on clustering results, including the average linkage method. This will be done for ALL FOUR DATASETS.

Exercise:

1. Navigate to the Script:

- Go to the HLMCC_python directory, then enter the HLMCC_model directory.
- Open the HAP_Clustering.py script using the nano text editor with the following command:

nano HAP_Clustering.py

2. Modify the Distance Metric and Linkage Method:

- Scroll down to the section labeled "**post-lab activity distance metrics.**"

```
#-----
#                               Post Lab Activity (Different Metrics & Linkages)
# Replace the metric and linkage in the variables below with the metrics and linkages specified in your lab
# guide
#-----
metric = 'manhattan'
linkage_method = 'average'
```

- Locate the two variables named metric and linkage, which are currently set to 'Manhattan' and 'average'.
- For each combination run the script using all 4 of the different datasets and answer the questions for that combination. Tip: answer the questions when going through each combination rather than trying to answer them after going through all 3.

▪ **Combination 1:**

1. Metric: 'manhattan'
2. Linkage: 'single'

Dataset	LWSNDR Single Hop Indoor	LWSNDR Multi Hop Outdoor	Satellite	IoT-23
Silhouette Score	0.937	0.9	0.426	0.69

▪ **Combination 2:**

1. Metric: 'minkowski'
2. Linkage: 'average'

Dataset	LWSNDR Single Hop Indoor	LWSNDR Multi Hop Outdoor	Satellite	IoT-23
Silhouette Score	0.929	0.946	0.505	0.69

▪ **Combination 3:**

1. Metric: 'euclidean'
2. Linkage: 'complete'

Dataset	LWSNDR Single Hop Indoor	LWSNDR Multi Hop Outdoor	Satellite	IoT-23
Silhouette Score	0.929	0.938	0.373	0.688

3. Observe and Record the Results:

- Note down your observations, including silhouette scores, and clustering accuracy. Use these observations to answer the following questions:

Questions:

1. Which dataset produced the most accurate clustering when comparing the ground truth data points to the results after running them through HAP clustering when using the **first combination** of metrics and linkages?

LWSNDR Single Hop Indoor

2. Which dataset produced the highest silhouette score when using the **first combination** of metrics and linkages?

LWSNDR Single Hop Indoor

3. Which dataset produced the most accurate clustering when comparing the ground truth data points to the results after running them through HAP clustering when using the **second combination** of metrics and linkages?

Can be close call but satellite dataset produces the most accurate clustering

4. Which dataset produced the highest silhouette score when using the **second combination** of metrics and linkages?

LWSNDR Multi Hop Outdoor

5. Which dataset produced the most accurate clustering when comparing the ground truth data points to the results after running them through HAP clustering when using the **third combination** of metrics and linkages?

IoT-23

6. Which dataset produced the highest silhouette score when using the **third combination** of metrics and linkages?

LWSNDR Multi Hop Outdoor