A Project Report

On

**Leak Detection in Water Distribution Networks using Machine Learning and Deep learning algorithms**

BY

**KANDE VINAY HARSHA VARDHAN**

**2010B4A21027H**

Under the supervision of

**PROF. A VASAN**

**SUBMITTED IN PARTIAL FULLFILLMENT OF THE REQUIREMENTS OF**

**CE F377: DESIGN PROJECT**



**BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE PILANI (RAJASTHAN)**

**HYDERABAD CAMPUS**

**(March 2023**)

**ACKNOWLEDGMENTS**

I would like to thank the BIRLA INSTITUTE OF TECHNOLOGY & SCIENCES, for giving me this opportunity of working with this prestigious project. It is not possible to do a project without the assistance and encouragement of other people, this one is certainly no exception. I would like to personally express my gratitude to the following people for their help in the work leading to this report



**Birla Institute of Technology and Science-Pilani,**

**Hyderabad Campus**

**Certificate**

This is to certify that the project report entitled “Leak Detection in Water Distribution Networks using Machine Learning and Deep Learning algorithms” submitted by Mr. KANDE VINAY HARSHA VARDHAN (ID No. 2019B4A21027H) in partial fulfillment of the requirements of the course CE F377, Design Project Course, embodies the work done by him under my supervision and guidance.

**Date:22 March 2023 ( PROF. A.VASAN)**

BITS-Pilani, Hyderabad Campus

**ABSTRACT**

Water is essential to human survival, human health, and economic growth on a global scale. As the global population rises, so does the standard that residents and workers have constant access to clean water. So, it is our responsibility to keep the pressure up and make sure water is distributed safely and efficiently. In many regions of the world, water leaks have become a major problem. So, in this project we aim to increase the efficiency and lifespan of pipe systems by catching leaks early and fixing them before they do any harm with the help of Machine Learning and Deep Learning algorithms.

**CONTENTS**

Title page…………………………………………………………….1

Acknowledgements………………………………………………….2

Certificate……………………………………………………………3

Abstract………………………………………………………………4

1. Literature Review….....………………...……………….…………6

2. Introduction…………………………………………………………7

3. Objectives…..………………………………………………………8

4. Leak Detection………………………………………………………9

4.1 Methodology……………………………………………………9

4.1.1 Leak Detection ……………………..................................................9

4.1.2 Normalisation ………………………………………..……………..10

4.1.3 Test-train split………………………………………………………10

4.1.4 Random Forest………………………………………………………10

4.1.5 Confusion Matrix……………………………………………………10

4.1.6 Accuracy, Precision and Recall……………………………………..11

4.2 Results………………………………………………………………………11

5. Leak Localization……………………………………..……………13

5.1 Methodology………………………………………………13

5.1.1 Data Balancing…………………………………..13

5.1.2 Leak Localization………………………………13

5.2 Results……………………………………………………14

6.References…………………………………………………….……16

**Literature Review:**

Prediction leak detection using simulated data was first introduced by **Caputo and Pelagagge** (2002). They looked at three different structures and made comparisons: ML-Perceptron, PNN, and BRF are all types of neural networks (MLP). In regards to predictive ability and noise rejection, they discovered that MLP performed the best. By utilizing SVM, **Mounce** et al. (2011) were able to spot anomalies in flow and pressure data.

Acoustic noise data was used by Kang et al. (2017) to propose a water leakage detecting system. They used SVM in conjunction with a convolutional neural network (CNN) with only one dimension. The overall accuracy of the procedure was 99.3%.

The pressure transient wave approach was emphasized by Chen et al. (2004). The negative pressure waves in the curve were identified using SVM. Their approach outperformed competing transient wave-based methods.

In addition, Zhou et al. (2019) created a leak detection approach that uses machine learning to examine actual recorded transient pressure wave data. Deep neural networks (DNNs) were utilised to train the neural weights and generate a leak event notification after a convolutional neural network (CNN) was employed to extract the important texture-features of transient wave samples. When the signal-to-noise ratio is 0 dB, it has been stated that this approach has a failure rate of less than 6 104.

Ayadi et al. (2019) used pressure data to implement a combination kernelized leak detection method, with dimensionality reduction provided by Kernel Fisher Discriminant Analysis. Also, they analysed the performance of the One Class Support Vector Machine (OCSVM) and the KNN classifier. The outcomes showed that OCSVM is more productive than KNN.

Using the LR algorithm, Oliveira et al. (2018) suggested a classification technique for leak detection. The focus of the research was on identifying the cutoff point between abnormal and typical readings. They discovered that picking the right threshold requires striking a balance between the two goals of increasing accuracy and lowering false alarms.

Bjerke (2019) investigated two recurrent neural network architectures for leak detection utilising produced data from the LeakDB dataset. According to the selected circumstances, the results showed high precision. On the other hand, the study did not make use of the LeakDB dataset's primary data.

In addition, they disregarded early leakage that are difficult to spot.

Prior efforts focused on enhancing detection and classification efficiencies by refining algorithm parameters and features. Most of them didn't account for problems with the data itself, water's flow patterns, or fluctuations in consumer demand. Their water distribution networks had unusually well-defined structures and sizes, as did the sizes and types of injection leaks. They also used an unsupervised learning mode, which makes validating their suggested models much trickier. This paper provides a clear comparison of several techniques within supervised learning, with the help of the publicly available LeakDB dataset serving as the standard. Comparing the effectiveness and efficiency of intelligent approaches and assessing their behaviour through application to hydraulic data is achieved through an experimental implementation utilising the same dataset and the same types of networks and leaks.

**INTRODUCTION:**

Finding the source and location of a leak in a water distribution network requires a method known as leakage detection and localization. The quantity of water lost from the system may be reduced if leaks were detected sooner. There may be further advantages as well. For instance, it may lessen the effects on the environment by cutting down on water waste, lessen the blow to customers during scheduled supply outages, and save money by cutting down on pumping and losses directly related to water waste. When a leak is discovered, it's important to pinpoint its exact position so that it can be fixed.

There are three stages involved in locating the source of a leak. The initial step is to locate the leak to a specific area, or zone. Step two involves tracing the pipes in a district where the leaks have been reported. The next step is to zero in on the exact spot where the leak is occurring, reducing the potential damage zone to a diameter of only two to three feet. A majority of the time, hardware-based approaches are used in the last stage of leakage detection and localisation, known as pinpointing.

To create a machine learning or deep learning model, this report draws its information from the free and publicly available LeakDB database. The Hanoi Network's time-series data was utilised for this study. It includes the requirements for the pipe network's pressure, flow, and demands. It includes a whopping one thousand different "Scenarios," each of which represents a unique demand scenario. Each Scenario consists of 32 nodes and 34 links. The Hanoi Connection is seen on Fig. 1.

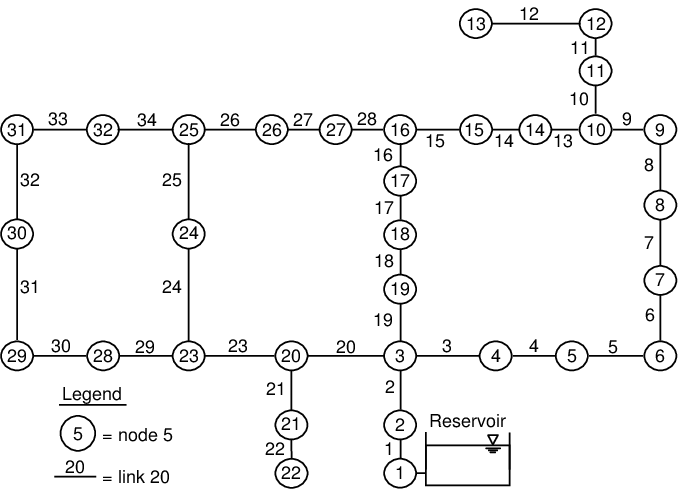


Figure-1

The pressure and demand at each node and flow in each link for every half an hour is provided for a year in a csv file for each scenario. The structure of the LeakDB dataset and an example data file is shown in figure 2.

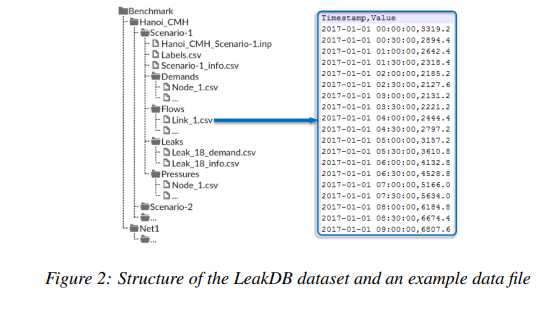
****

Figure-2

In this experiment, the diameter of the leaking holes was randomised to be anywhere from 2 centimetres to 20 centimetres, resulting in a wide range of leakage intensities.

The LeakDB dataset organises each network's simulated situations into its own directory, for assessments of algorithms tailored to the specifics of each network. There are hundreds of nsc scenarios in each Network Directory, each with a different amount of leakage occurrences (or no events).

The node requirements, node pressures, link flows, and leak flow data for each case are all stored in separate "CSV" files. A "Timestamp" column with the format "YYYY-MM-DD hh:mm:ss" and a "Value" column with the same format are included in each data file (real numbers).

**OBJECTIVES**

1. To test one of the available machine learning methodologies to detect the leak in water distribution network
   1. Data pulling
   2. Data preprocessing
   3. Data analysis
   4. Evaluating the performance
2. To improve the accuracy in detecting leaks.
3. Early Leak detection.
4. Leak localization

**LEAK DETECTION**

**Methodology**

Preparation of data, including consolidation of all pressure, flow, demand, and leak data into a single data frame.

The available historical data is culled from several sources, but one that stands out is LeakDB, which has a whopping one thousand scenarios, each of which involves a water distribution network with 32 nodes. From 1 JAN 2017 to 12 DEC 2017 at half-hour intervals, LeakDB offers us with demand data, flow data, pressure data, and leak with its duration and kind for each of 32 nodes.

To begin training our model, we chose 10 out of 1000 situations and used Python to compile all of the available data for pressure, demand, and flow along with their associated timestamps.

After merging all the data to a single dataframe, the dataframe of one of the scenarios is shown in figure 3. In the same way we merge all the 1000 scenarios to single dataframe and that dataframe consists of 17520000 rows.

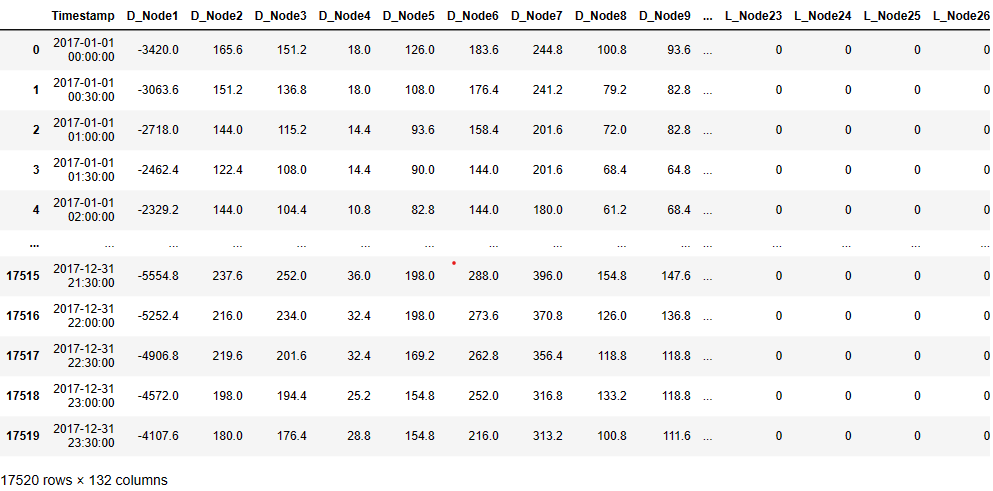


Figure-3

**Leak Detection:** The process of detecting leaks entails the creation of a new binary column labelled "Leak" within a designated dataframe. This column is generated by computing the aggregate sum of values across all Leak nodes. Initially, the procedure identifies the columns that encompass data pertaining to leak nodes. Subsequently, it calculates the summation of the values present in these columns for every row. Finally, a new column named "Leak" is generated, which is initialised with zeroes. The procedure mentioned above subsequently modifies the "Leak" column to a value of 1 in instances where the summation of leak nodes exceeds 0. This process detects if there is any leak in the whole network.

**Normalisation:** The process of normalisation involves defining a function called ‘normalise’ that accepts a dataframe ‘df’ as an argument and produces a normalised version of the input. A new dataframe named `result` is generated within the function. The function proceeds to iterate through each feature or column present in the dataframe and calculates the maximum and minimum values associated with said feature. Subsequently, the characteristics' values are converted to floating-point values. Subsequently, the values of the feature are subjected to normalisation through the utilisation of the formula ‘(x - min\_value) / (max\_value - min\_value)’. The resultant normalised values are then stored in the designated `result` dataframe. The resultant normalised dataframe is returned by the function. In general, this function serves the purpose of standardising the data within a dataframe, a prevalent approach employed in the preprocessing and analysis of data.

**Test train split:** In order to divide a dataset into training and testing sets, we import the ‘train\_test\_split’ function from the ‘sklearn.model\_selection’ module. The input dataset is broken down into two arrays, ‘X’ and ‘y,’ with ‘X’ including the features (here, pressure and flow columns) and ‘y’ containing the target (here, ‘Leak’). The input dataset is randomly divided into training and testing sets by the 'train\_test\_split' function; the 'test\_size' parameter indicates the percentage of the dataset to be allocated for testing (33% in this case). The function returns four arrays, ‘X\_train’ for the training set's features, ‘X\_test’ for the testing set's features, ‘y\_train’ for the training set's target values, and ‘y\_test’ for the testing set's target values. The dataset is divided in this fashion to avoid overfitting the model and to permit testing of the model's performance on new data.

**Random Forest:** The machine learning method known as random forest has seen extensive usage in a variety of academic settings. It's an ensemble technique for building a more powerful and precise prediction model through the use of several decision trees. In order to construct decision trees, random forest first selects features and data points at random. Each decision tree in the forest contributes to the final forecast, which is then aggregated. The high dimensional features and noise in the data are no match for random forest. It has been put to use in a number of different contexts, including the very precise resolution of classification and regression issues in healthcare, finance, and bioinformatics.

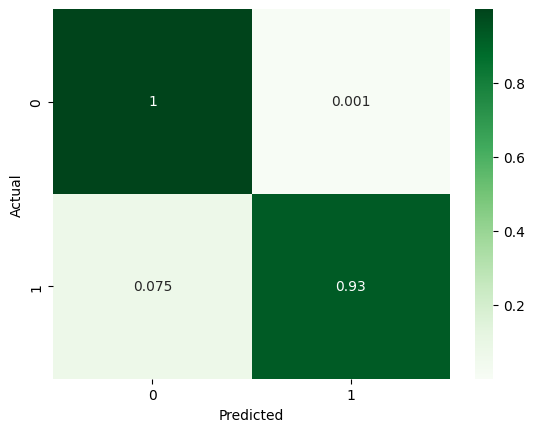
**Confusion Matrix:** We bring in the ‘confusion\_matrix’ function from the ‘sklearn.metrics’ module to construct the confusion matrix for a two-class issue. Predicted values (‘predictions’) and true target values (‘y\_test’) are used to calculate the confusion matrix for the test dataset. In this scenario, the 'labels' option determines which labels come first in the confusion matrix (rows) and which come second (columns), with 0 and 1 representing the true and predicted labels, respectively. The resultant confusion matrix ‘cfmat’ is a 2x2 array that includes the total counts of positive, negative, and invalid responses. A confusion matrix can be used to assess the efficacy of a binary classifier; TP stands for the number of correctly identified positive instances, FP for the number of incorrectly identified negative instances, TN for the number of correctly identified negative instances, and FN for the number of correctly identified positive instances that were misclassified as negative. Accuracy, precision, recall, and the F1 score are only few of the assessment metrics that may be calculated using the confusion matrix.

**Accuracy, Precision and Recall:** To evaluate a binary classifier, we utilise the accuracy\_score, precision\_score, and recall\_score methods included in the sklearn.metrics package. The ‘accuracy\_score’ function determines how well the model performs by contrasting the falsely predicted values ('predictions') with the actual target values (‘y\_test’). The ‘precision\_score’ function determines the accuracy of the model by calculating the proportion of correct predictions to all correct ones. Model recall, defined as the proportion of correct classifications relative to the total number of positive occurrences, is calculated by the ‘recall\_score’ function. Both ‘y\_test’ and ‘predictions’ are required as inputs to each of these procedures. Each function returns a scalar value that acts as a proxy for its own metric of evaluation. It is possible to evaluate a binary classification model's efficacy and compare models using these measures.

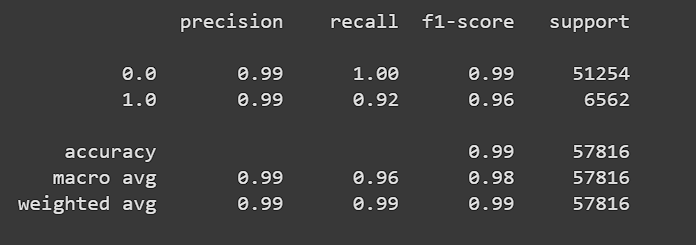
**Results:**

1. **Both pressure and flow as input**

Confusion Matrix

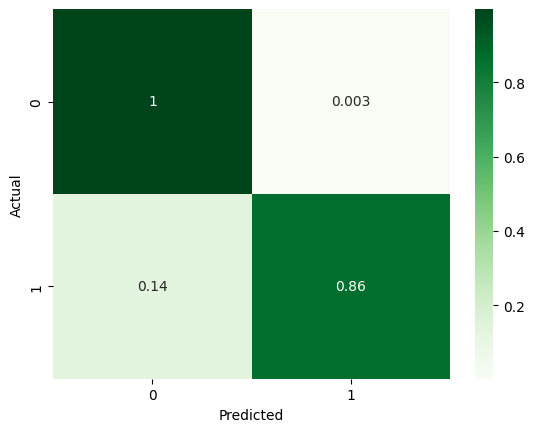
****

Classification report

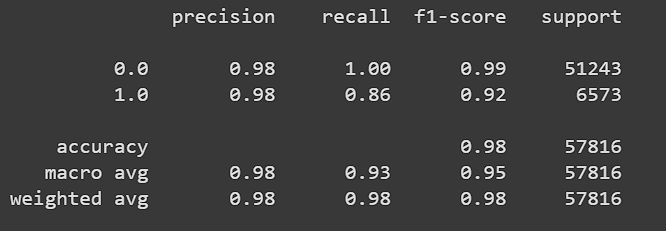


1. **Only Pressure as input**

Confusion Matrix

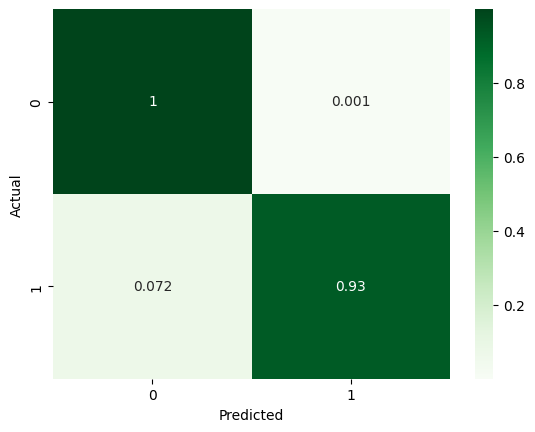
****

Classification report

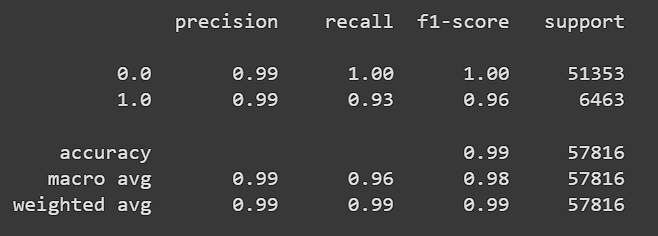


1. **Only Flow as input**

Confusion Matrix



Classification report



|  |  |  |  |
| --- | --- | --- | --- |
|  | Accuracy | Precision | Recall |
| Pressure and Flow as input | 0.9906 | 0.9919 | 0.9248 |
| Only Pressure as input | 0.9819 | 0.9772 | 0.8616 |
| Only Flow as input | 0.9913 | 0.9932 | 0.9284 |

**LEAK LOCALIZATION**

**Methodology:**

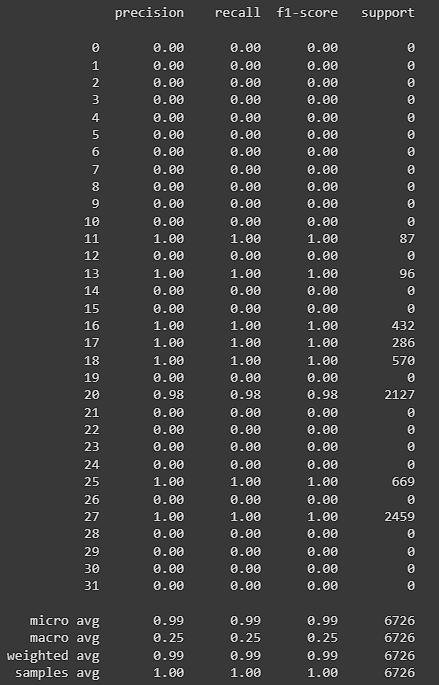
**Balancing the data:** The procedure of data balancing entails the resolution of the predicament of imbalanced data in a given dataset, wherein the frequency distribution of values in one or more columns is significantly skewed towards a specific value (in this case, 0, signifying the absence of a leak). In the domain of leak localization, the matter at hand can pose a significant challenge if the preponderance of data entries denotes the non-existence of a leak within the system, potentially leading to a prejudiced or imprecise model. In order to tackle this matter, a revised dataframe is generated, comprising of records where the "Leak" column exhibits a value of 1, denoting the existence of a leak, by doing so, the target vector of size 32 contains sufficient 0s and 1s to ensure a more balanced distribution of values. Through this approach, the resultant target vector, which possesses a dimensionality of 32, is endowed with an adequate proportion of binary values, thereby promoting a more equitable dispersion of data points. This, in turn, has the potential to enhance the precision and dependability of the leak localization algorithm. The aforementioned procedure holds significance as it guarantees the model's capability to precisely detect the existence and positioning of leaks within a designated system, and to prevent plausible safety risks or ecological harm that could arise from unobserved leaks.

**Leak Localization:** In order to precisely identify the particular node or component of a system where a leak has occurred, the process of leak localization requires the use of a multi-output model. This model accepts as an input a target feature set that consists of 32 binary features. Each of these features corresponds to a node in the network and signals whether or not a leak is present at that node. When the results of this model are analysed, it is possible to pinpoint the exact position of the water leak with high levels of accuracy. previous leak detection models are limited in that they are only able to determine whether or not a leak is present in the network as a whole; they are unable to pinpoint the specific site of the leak. The technique of leak localization represents a significant development over traditional leak detection methods. Because of this, locating the source of the leak and making the necessary repairs may be far more difficult and expensive. Leak localization, on the other hand, offers a method that is both more efficient and successful in locating leaks in industrial systems and resolving them.

**Results:**

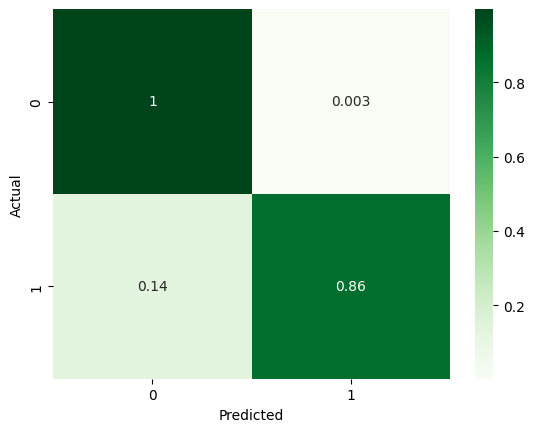
1. **Both pressure and flow as input**

Classification report

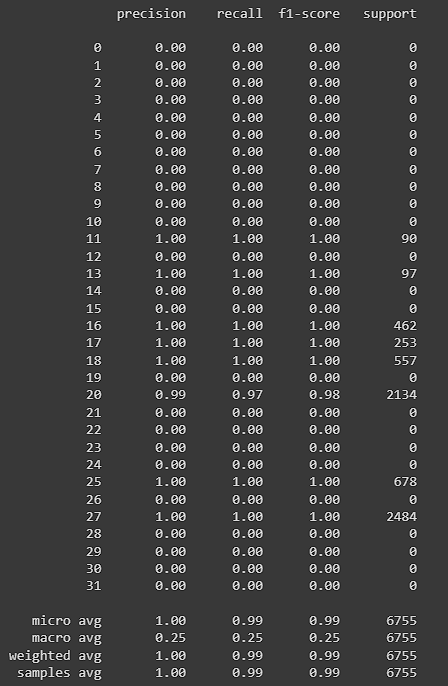


1. **Only Pressure as input**

Confusion Matrix

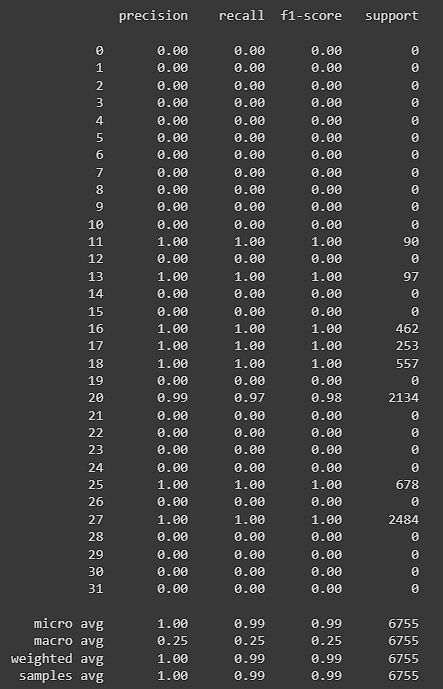
****

Classification report



1. **Only Flow as input**

Classification report



|  |  |  |  |
| --- | --- | --- | --- |
|  | Accuracy | Precision | Recall |
| Pressure and Flow as input | 0.9863 | 0.9946 | 0.9919 |
| Only Pressure as input | 0.9742 | 0.9950 | 0.9899 |
| Only Flow as input | 0.9849 | 0.9951 | 0.9899 |

**REFRENCES**

1. Review of Current Technologies and Proposed Intelligent Methodologies for Water Distributed Network Leakage Detection.
2. Literature Review of Data Analytics for Leak Detection in Water Distribution Networks: A Focus on Pressure and Flow Smart Sensor.
3. LeakDB: A benchmark dataset for leakage diagnosis in water distribution networks.
4. Experiments based comparative evaluations of machine learning techniques for leak detection in water distribution systems.
5. <https://github.com/KIOS-Research/LeakDB>
6. Maryam Kammoun1, Amina Kammoun1, Mohamed Abid1.LSTM‐AE‐WLDL: Unsupervised LSTM Auto‐Encoders for Leak Detection and Location in Water Distribution Networks
7. Stelios G. Varchimis, Mario S. Kyriakou, Demetrios, Marios M. Polycarpou. LeakDB: A benchmark dataset for leakage diagnosis in water distribution networks
8. Maryam Kammoun1, Amina Kammoun1, Mohamed Abid1. Experiments based comparative evaluations of machine learning techniques for leak detection in water distribution systems.
9. Martin Bjerke. Leak Detection in Water Distribution Networks using Gated Recurrent Neural Networks