Water Quality Data Analysis

University of Groningen

Distributed Systems Group

Document No.…………………………………….........V1.0

Date...............…......................…........................... February 15, 2022

Version...1.0

Presenter...…… Narges Norouzi

Table of Contents

1   Introduction

This document reports the results of descriptive and predictive analyses on the KU-MWQ dataset. These analyses have been implemented in two separate **Container-based and VM-based clustered environments.**

Note: The structure and information of the attached files are summarized in Table 1 and Fig. 1.
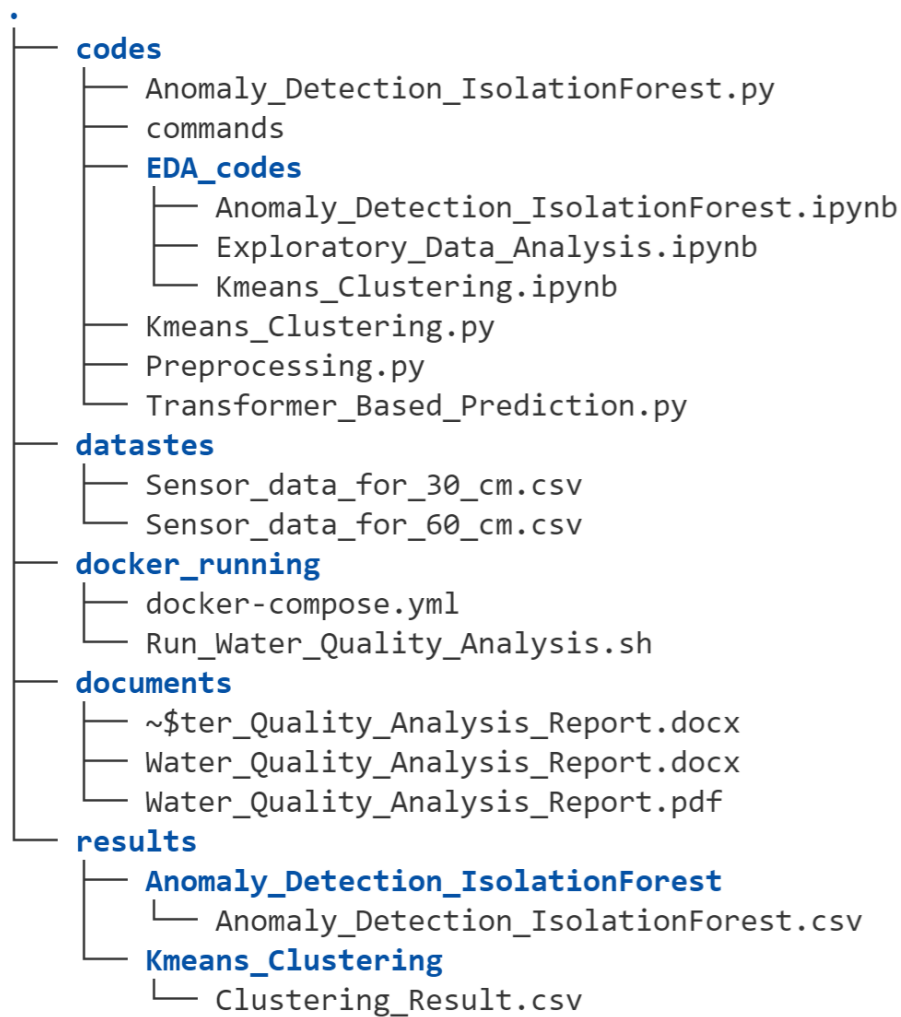
```
.
├── codes
│   ├── Anomaly_Detection_IsolationForest.py
│   ├── commands
│   ├── EDA_codes
│   │   ├── Anomaly_Detection_IsolationForest.ipynb
│   │   ├── Exploratory_Data_Analysis.ipynb
│   │   └── Kmeans_Clustering.ipynb
│   ├── Kmeans_Clustering.py
│   ├── Preprocessing.py
│   └── Transformer_Based_Prediction.py
├── datastes
│   ├── Sensor_data_for_30_cm.csv
│   └── Sensor_data_for_60_cm.csv
├── docker_running
│   ├── docker-compose.yml
│   └── Run_Water_Quality_Analysis.sh
├── documents
│   ├── ~$ter_Quality_Analysis_Report.docx
│   ├── Water_Quality_Analysis_Report.docx
│   └── Water_Quality_Analysis_Report.pdf
└── results
    ├── Anomaly_Detection_IsolationForest
    │   └── Anomaly_Detection_IsolationForest.csv
    └── Kmeans_Clustering
        └── Clustering_Result.csv
```

Fig.1. The structure of the attached files

Table 1: The information of the attached files

No.	Name	Description	
		Target	Type
1	Preprocessing.py	Data Preprocessing Steps	py
2	Kmeans_Clustring.py	Water Quality Clustering	py
3	Anomaly_Detection_IsolationForest.py	Parameter Anomaly Detection	py
4	Transformer_Based_Prediction.py	Parameters Prediction	py
5	Exploratory_Data_Analysis.ipynb	Descriptive Analysis	ipynb
6	Anomaly_Detection_IsolationForest.ipynb	Anomaly Visualization	ipynb
7	Kmeans_Clustering.ipynb	Clustering Visualization	ipynb
8	Water_Quality_Analysis_Report.pdf	Document	pdf
9	Anomaly_Detection_IsolationForest.csv	Anomaly Detection Results	csv
10	Clustering_Result.csv	Clustering Results	csv
11	run_water_quality_analysis.sh	Docker Running	sh
12	docker-compose.yml	Docker Config File	.yml

2   Implementation Infrastructure

The following two infrastructures have been used to implement the analyses.

2-1   Three Nodes Container-Based Cluster Using Docker Compose

In this section, a three-node cluster has been set up using Docker Compose, whose main purpose is to execute Jupyter notebooks containing the result visualization.

Actually, I have created these docker images (2 Spark nodes and 1 Jupyter node) so that .ipynb files can be executed by you on spark and you can see the results. How to set up this environment and execute the codes is described in section 3.

2-2   Five Nodes VM-Based Cluster Using Hortonwork Platform

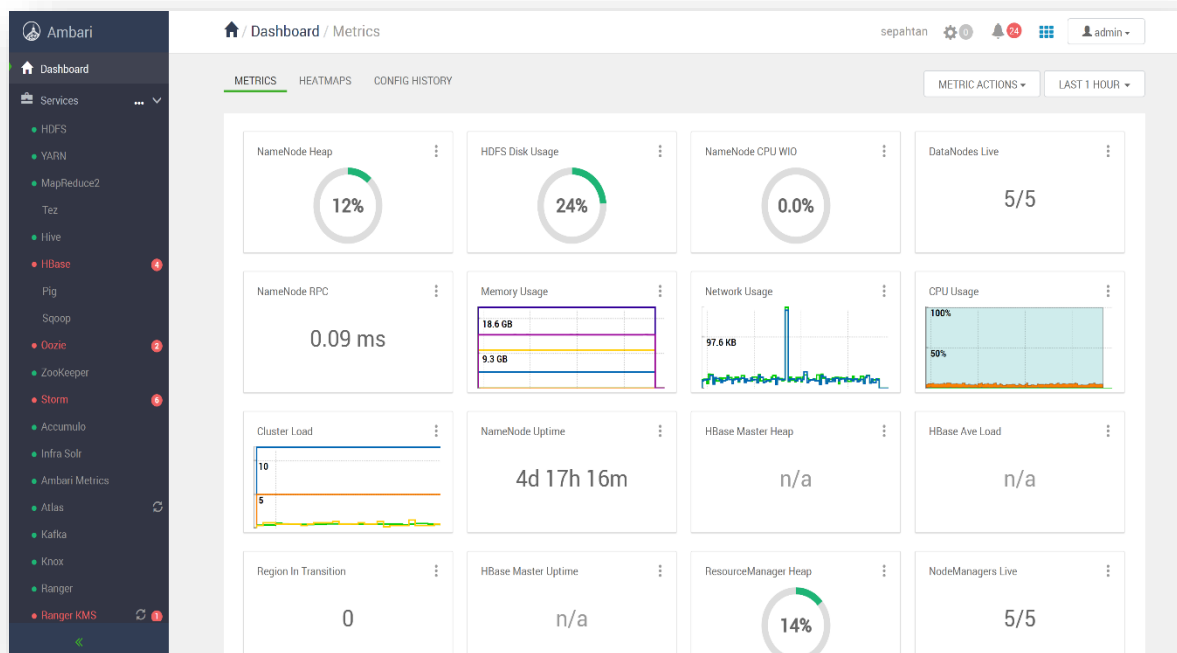This infrastructure, as shown in Fig. 2., has been used to implement machine learning based analyses.

Fig. 2. : Five Nodes VM-Based Cluster Using Hortonwork Platform

 I have set up a five-node cluster using the Hortonwork Data platform which is an open source tool for distributed storage and processing of large, multi-source data. Services such as Apache Storm, Apache Kafka, Apache Solr, and Apache Druid are also installed on it, which

I have used Spark 2.3.0 to implement and execute codes, according to the requirements of the task.

3   Exploratory Data Analysis (EDA: Descriptive Analysis)

Exploratory Data Analysis refers to the critical process, of performing initial investigation on raw data to maximize insight, uncover underlying structures and extract important features, which make it easier for us to develop machine learning models to discover patterns, spot anomalies, and forecasting. Hence, in this section, descriptive analyses have been performed in two modes with and without time considerations, before using machine learning algorithms to discover main hidden patterns from data.

Note: Information about the codes belonging to this section is shown in the fifth row of Table 1.

3-1   Code Running Instructions

For running the codes of this section which contains Jupyter notebooks, you can use one of the following methods:

- **Use My Docker Compose**

| **Hardware Requirements:** |
| Please consider you need a minimum of 5GB free space and 4GB of RAM |

| 1 | Install docker and docker-compose packages: |
| | Sudo apt install docker.io docker-compose |

2	Go to the docker_running directory and run the script
	Sudo chmod +x run_water_quality_analysis.sh
	Sudo ./run_water_quality_analysis.sh start

| 3 | Simply open the output URL in a web browser to start Jupyter |
| | **Run the notebooks** |

| 4 | To stop the containers just run the following command: |
| | **Sudo ./run_water_quality_analysis.sh start stop** |

- **Use PySpark in Jupyter**

  There are two ways to get PySpark available in a Jupyter Notebook:

  - **Configure PySpark driver to use Jupyter Notebook: running pyspark will automatically open a Jupyter Notebook**

1	Update PySpark driver environment variables, add these lines to your **~/.bashrc** (or **~/.zshrc**) file.
2	Restart your terminal and launch PySpark. This command should start a Jupyter Notebook in your web browser.
3	Open the Exploratory_Data_Analysis.ipynb file and run it. Please note that you should change the path of the dataset directory from HDFS to your path.

  - **Load a regular Jupyter Notebook and load PySpark using the FindSpark package**

    There is another and more generalized way to use PySpark in a Jupyter Notebook: use the FindSpark package to make a Spark Context available in your code.

1	Install FindSpark Package: Pip3 install findspark
2	Import findSpark and use findSpark.init() or findSpark.find(), I have already added this statement to my code, so there is no need to add it again. But just change the Spark HOME path
3	Open the Exploratory_Data_Analysis.ipynb file and run it. Please note that you should change the path of the dataset directory from HDFS to your path.

3-2 Preprocessing Steps

Data preprocessing is an important step to prepare high-quality data which is suitable for descriptive and predictive analyses and also increases the accuracy and efficiency of the machine learning models.

In this section, the preprocessing steps are performed in four stages: data cleaning, data integration, data resampling, and feature extraction which are described in detail in Table 2.

Note: Information about the codes belonging to this section is shown in the first row of Table 1.

Table 2: Preprocessing Steps

No.	Function names	Targets
1	Preprocessing()	Rename The Columns
2	add_new_columns()	Adding Humidity and Night Columns (Rainy=1, Dry=0 , Night=1, Day=0)
3	cal_water_quality_index()	Water Quality Estimation
4	Downsample()	Resampling The Data Based On Different Time (Per Minute, Per Hour, Per Day, …)

3-2-1   Adding External Variables

External variables play a very important role in analyzing time series data. The KU-MWQ dataset contains two main external parameters: weather conditions (Dry, Rainy) and time (Day, Night)

To investigate the effect of these parameters on the values of the sensors as well as water quality, the **add_new_columns**() function has been implemented which is responsible to add **Humidity** and **Night** columns to existing data.

Note: This dataset also contains data collected at a depth of 60 cm, which I did not review due to the lack of a pH column.

3-2-2   Water Quality Estimation

There are some statistical methods for calculating water quality based on various parameters. Inspired by these methods, in this section, I have used a very simple equation to combine Temperature, pH, and, Turbidity features to approximate water quality. Eq. 1 shows how quality parameters are calculated.

$$WQ_i = \sum_p \left(value_p{}^I / S_p \right) \qquad (1)$$

where S_p is the value of permissible standard for pth parameter and the I is the proportionality constant and taken as; **1.8615**.

Note: I don't have comprehensive domain knowledge about water. Therefore, the results of calculating quality parameters can be very noisy.

3-3   Distribution Analysis

As mentioned before, in this section, the EDA has been performed in two modes with and without time considerations which are illustrated in Tables 3 to 9.

Note: The extracted Insights from each of these analyses are reported in the last row (Findings Row) of each table.

3-3-1   Distinct Value Counts

Table 3. Distinct Value Counts

Function Name	Target
distinct_count()	Calculating Distinct Value Counts for Columns

Results
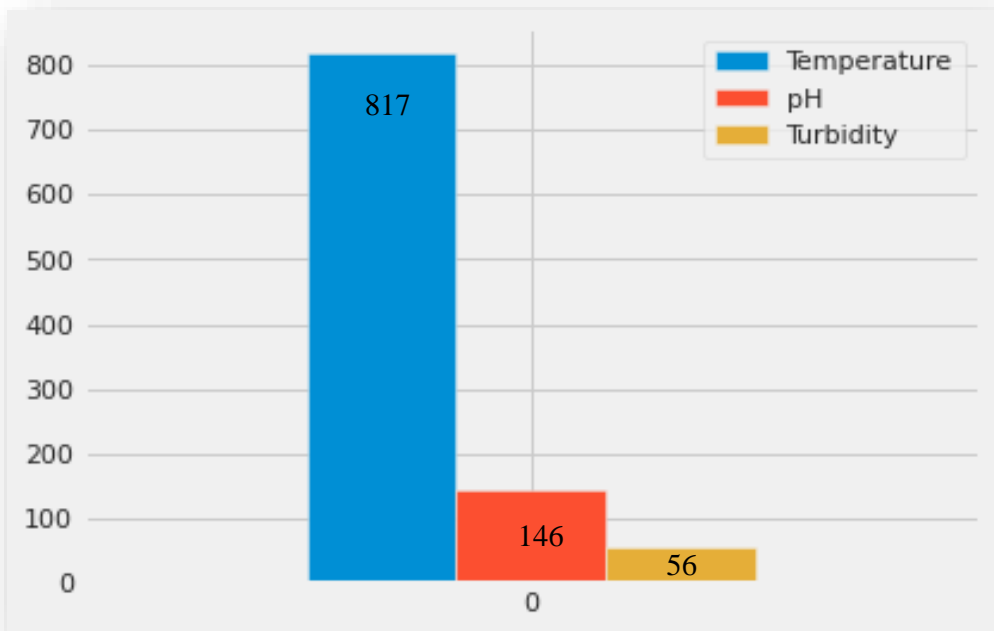
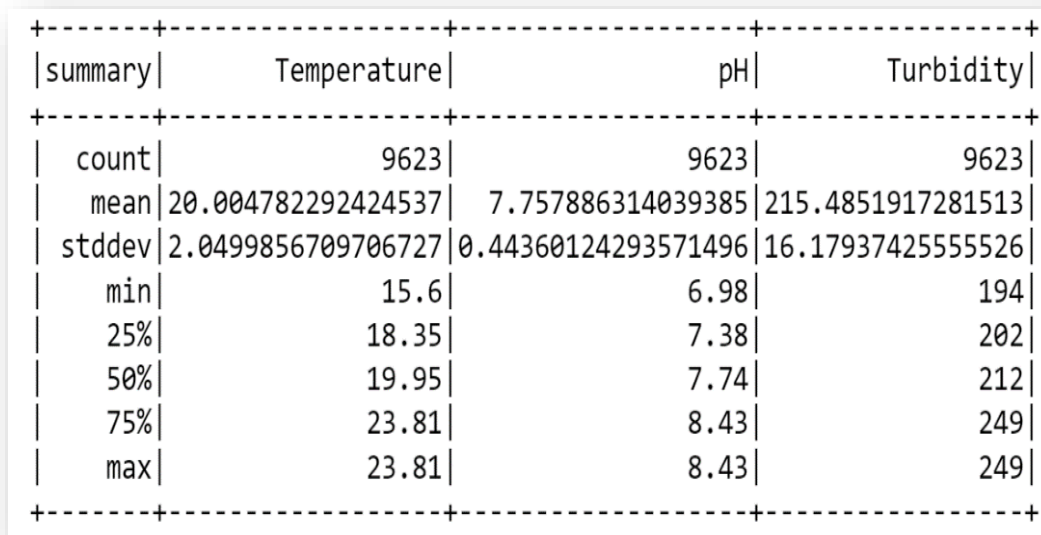**Fig. 3. Distinct Value Counts**

Findings

√ **The Temperature parameter has higher variability than other parameters.**

3-3-2   Statistical Summary for Each Column

Table 4. Statistical Summary

Function Name	Target
summary()	Computing specified statistics for columns

Results	


```
+-------+------------------+---------------------+------------------+
|summary|       Temperature|                   pH|         Turbidity|
+-------+------------------+---------------------+------------------+
|  count|              9623|                 9623|              9623|
|   mean|20.004782292424537|    7.757886314039385|215.4851917281513|
| stddev|2.0499856709706727|0.44360124293571496|16.17937425555526|
|    min|              15.6|                 6.98|              194|
|    25%|             18.35|                 7.38|              202|
|    50%|             19.95|                 7.74|              212|
|    75%|             23.81|                 8.43|              249|
|    max|             23.81|                 8.43|              249|
+-------+------------------+---------------------+------------------+
```

Fig. 4. Statistical Summary

Findings

√ **The Turbidity parameter has the highest standard division, so needs to be normalized.**

3-3-3   Feature Correlation Matrix

Table 5. Correlation Matrix

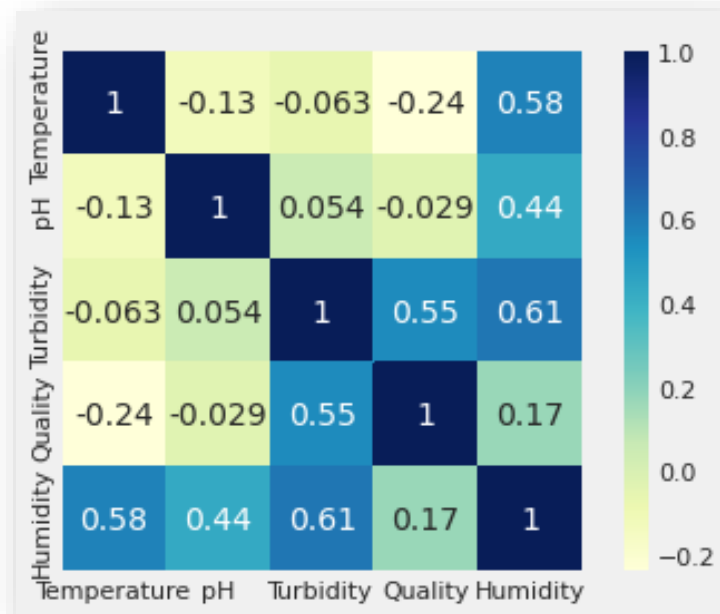| Function Name	Target
cal_corr ()	Calculating correlation matrix for parameters

Results

Fig. 5. Correlation Matrix

Findings

√**There is no hard positive correlation between Temperature, pH, and Turbidity. Hence we cannot drop any of them or use them to impute missing values.**

√**There is a negative correlation between the pH, and Temperature (-0.13). Therefore, increasing the Temperature is expected to reduce pH and vice versa.**

√**There is a significant correlation between Humidity and Turbidity (0.61). Therefore, Turbidity is expected to be higher on rainy days than on dry days.**

3-3-4   Investigating the Effects of Humidity on the Parameters
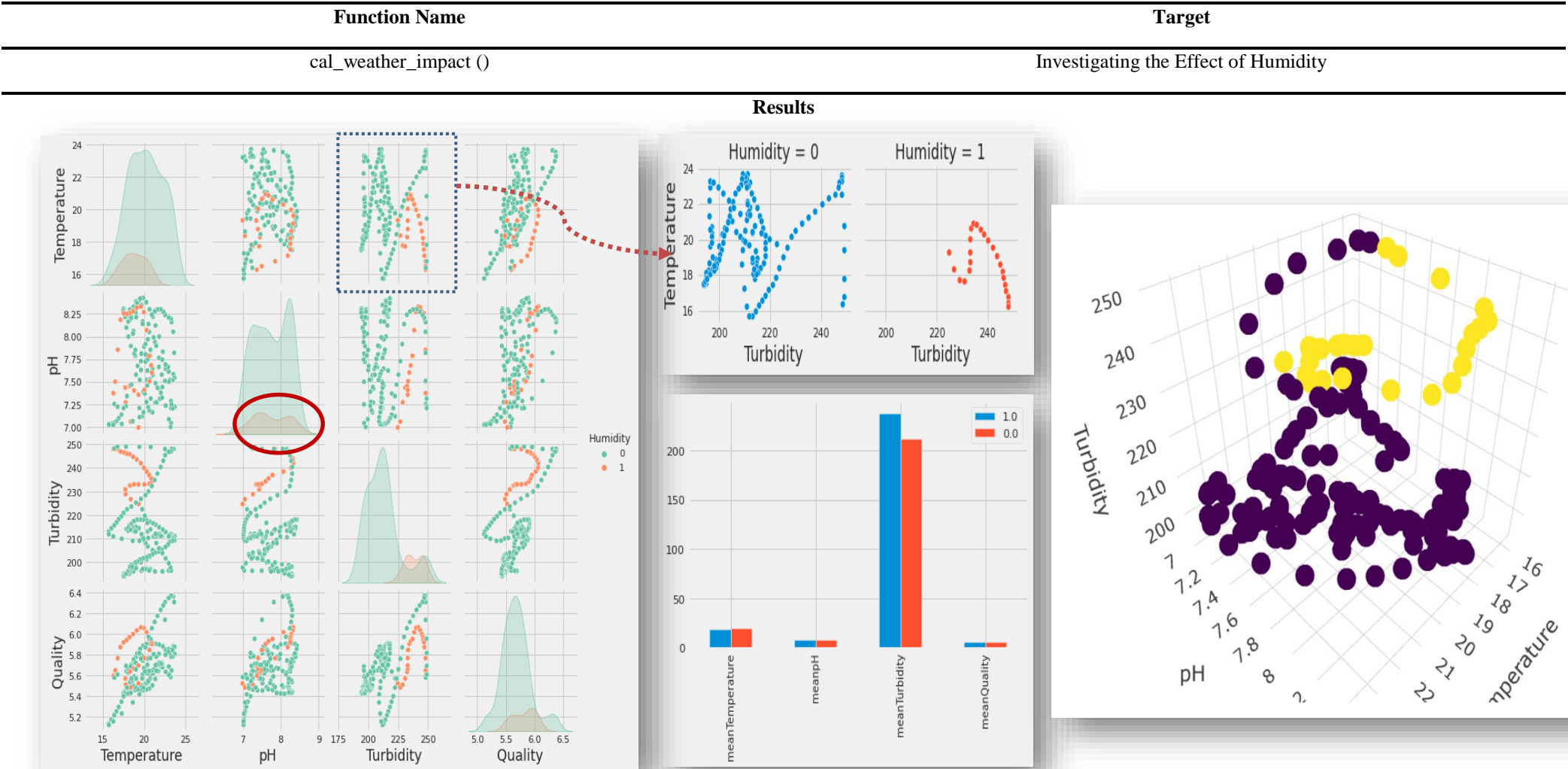
Table6. Humidity Effects

Function Name	Target
cal_weather_impact ()	Investigating the Effect of Humidity

Results

Fig.6. Humidity Effects

Findings

√ **As expected these figures confirm the direct effect of Humidity on water Turbidity. It can be concluded that on rainy days the Turbidity of water increases and as a result, its quality decreases. Turbidity range: (225,250)**

√ **It is also observed that the Temperature has decreased on rainy days. Temperature range: (16,21)**

√**The strange and remarkable thing that can be seen in these figures is the lack of effect of Humidity on the pH which should be investigated at time series plots. In other words, the pH does not increase with decreasing temperature. It can be said that an anomaly has occurred that should be extracted using anomaly detection algorithms.**

3-3-5   Parameters Distribution Analysis

Table 7. Distribution plots

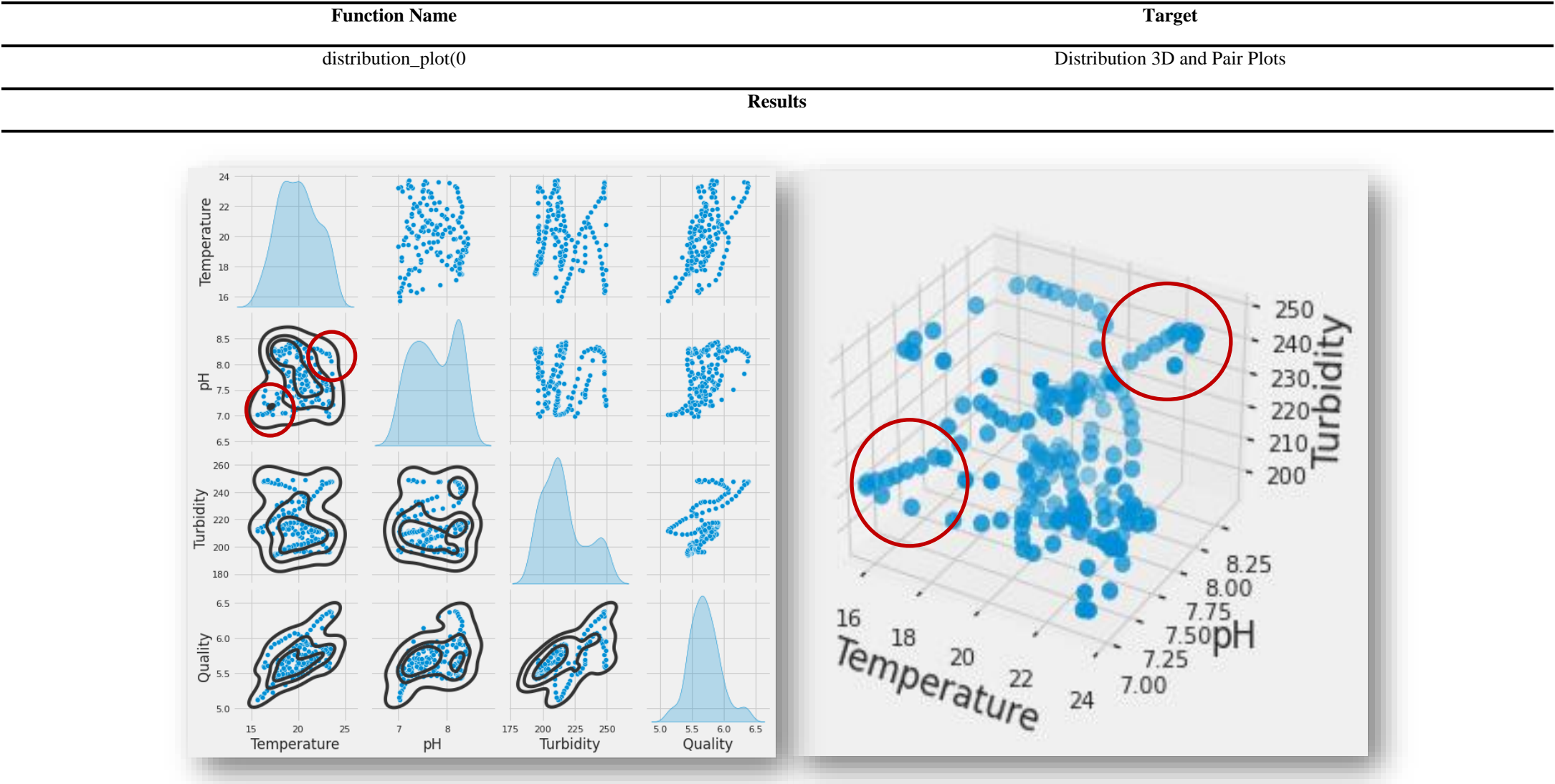| Function Name	Target
distribution_plot(0	Distribution 3D and Pair Plots

Results

Fig.7. Distribution plots

Finding

√ The Temperature has a bell-shaped curve, so has a normal distribution. The pH and Turbidity have a right and left-skewed respectively, so need to be normalized.

√ The pair plot shows that there is an appropriate relationship between (Temperature, pH) and (Temperature, Turbidity) which can be used for water quality clustering.

√ Some abnormal points are seen in the plot between pH and Temperature, which are indicated by red circles. It seems that a negative correlation between these two parameters is not maintained at these points. In other words, increasing the Temperature in these areas has not reduced the pH of the water and vice versa.

3-3-6   Temperature and pH Time Series Analysis

Table 8. Temperature and pH Time Series plots

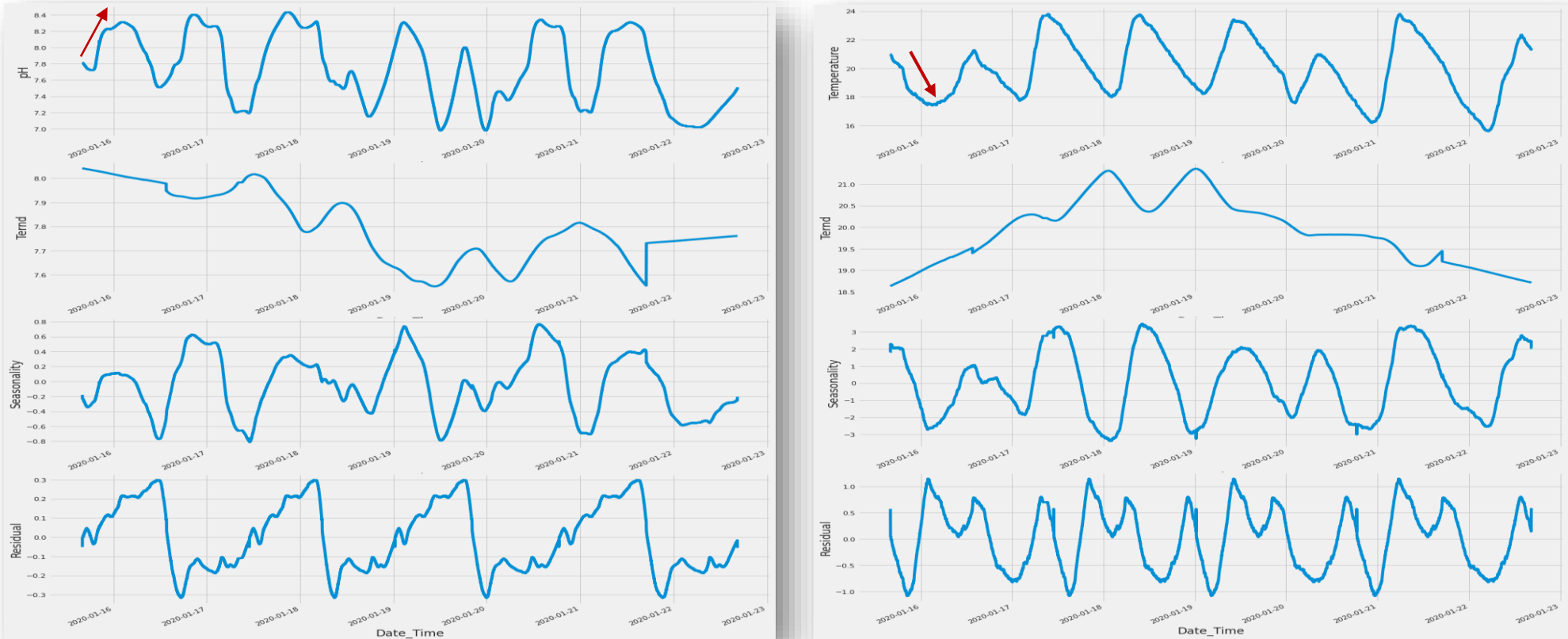| Function Name	Target
timeseries_plot() & Checking_trend_seasonality()	Decomposing the Temperature and pH Time Series

Results

Fig.8. Temperature and pH Time Series plots

Finding

√ As shown by the red arrows, these figures also confirm the negative correlation between Temperature and pH.

√ An obvious daily seasonality is seen in both time series.

√ The Temperature figure shows an up-trend until date (2020-01-19), and then a down-trend has happened from that date.

√ The pH figure shows an overall up-trend.

√ The residual figures show that the Temperature and pH parameters are predictable.

3-3-7   Turbidity and Quality Time Series Analysis
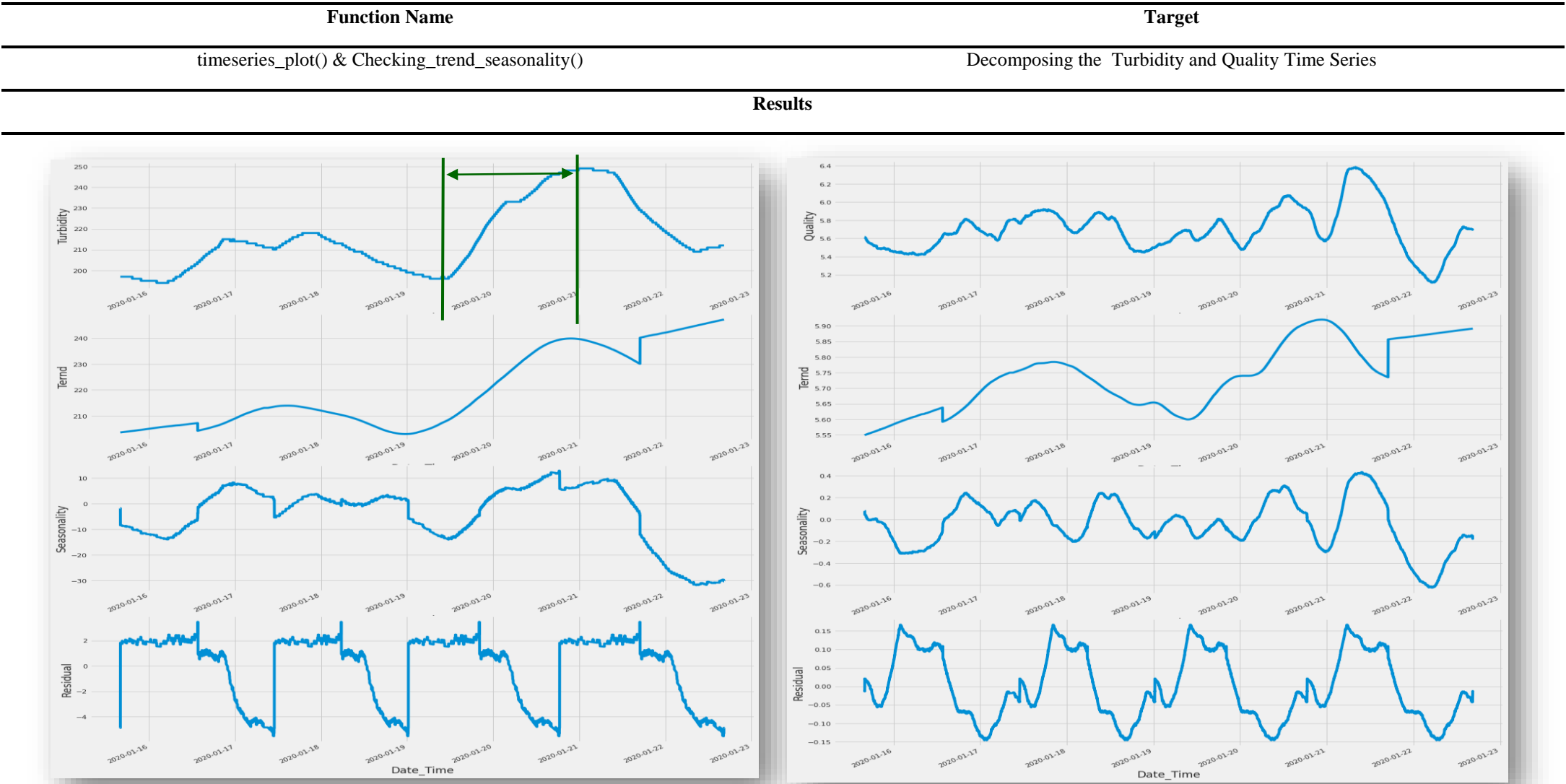
Table 9. Turbidity and Quality plots

Function Name	Target
timeseries_plot() & Checking_trend_seasonality()	Decomposing the Turbidity and Quality Time Series

Results

Fig.9. Turbidity and Quality plots

Finding

√ **There is no obvious daily seasonality in both time series. So they have to be stationary to predict**

√ **Both figures show up-trend.**

 √ **As can be seen in the figure on the left, the amount of Turbidity in the 2020-01-19 22:30 and 2020-01-20 23:00 time periods have increased with a steep slope, which is indicated by green vertical lines. This sudden change has occurred due to rainfall during these periods. In other words, Turbidity usually increases sharply during heavy rainfall when particles from the soil surface are washed into the pond and the pond bed sediment is re-suspended**

√ **The image on the right also shows that the water quality was almost constant (between 5.4-5.8) before it started to rain, but has decreased significantly since then (between 6-6.4).**

4   Machine Learning-Based Analysis (Diagnostic&Predictive)

Diagnostic and predictive analytics takes descriptive results a step further and provides deeper analysis to extract main patterns from data. Hence, in this section, three unsupervised algorithms are used to detect anomalies, cluster data based on water quality and, multivariate forecast which is described in the following sections, respectively.

4-1   Code Running Instructions

For running the codes of this section which contains .py files, you can simply follow these instructions:

1 Cd Water_Quality/codes/

2 spark-submit --py-files Preprocessing.py --class --deploy-mode cluster
org.apache.spark.examples.SparkPi --master yarn Kmeans_Clustering.py

3   Note: **Please note that you should change the path of the dataset directory from HDFS to your path.**

4-2   Multivariate Anomaly Detection Using Isolation Forest Algorithm

Isolation Forest is a fast tree-based algorithm for anomaly and outlier detection. And Since there are no predefined labels, it is an unsupervised model. This algorithm takes advantage of the two characteristics of anomalies, that they are few and different. Furthermore, it is built on the premise that anomalous points are easier to isolate than regular points through random partitioning of data. In an Isolation Forest, randomly sub-sampled data is processed in a tree structure based on randomly selected features. The samples that travel deeper into the tree are less likely to be anomalies as they required more cuts to isolate them. Similarly, the samples which end up in shorter branches indicate anomalies as it was easier for the tree to separate them from other observations.

In this analysis, the Isolation Forest algorithm has been used to detect and highlight anomalies and its results are described in Table 10.
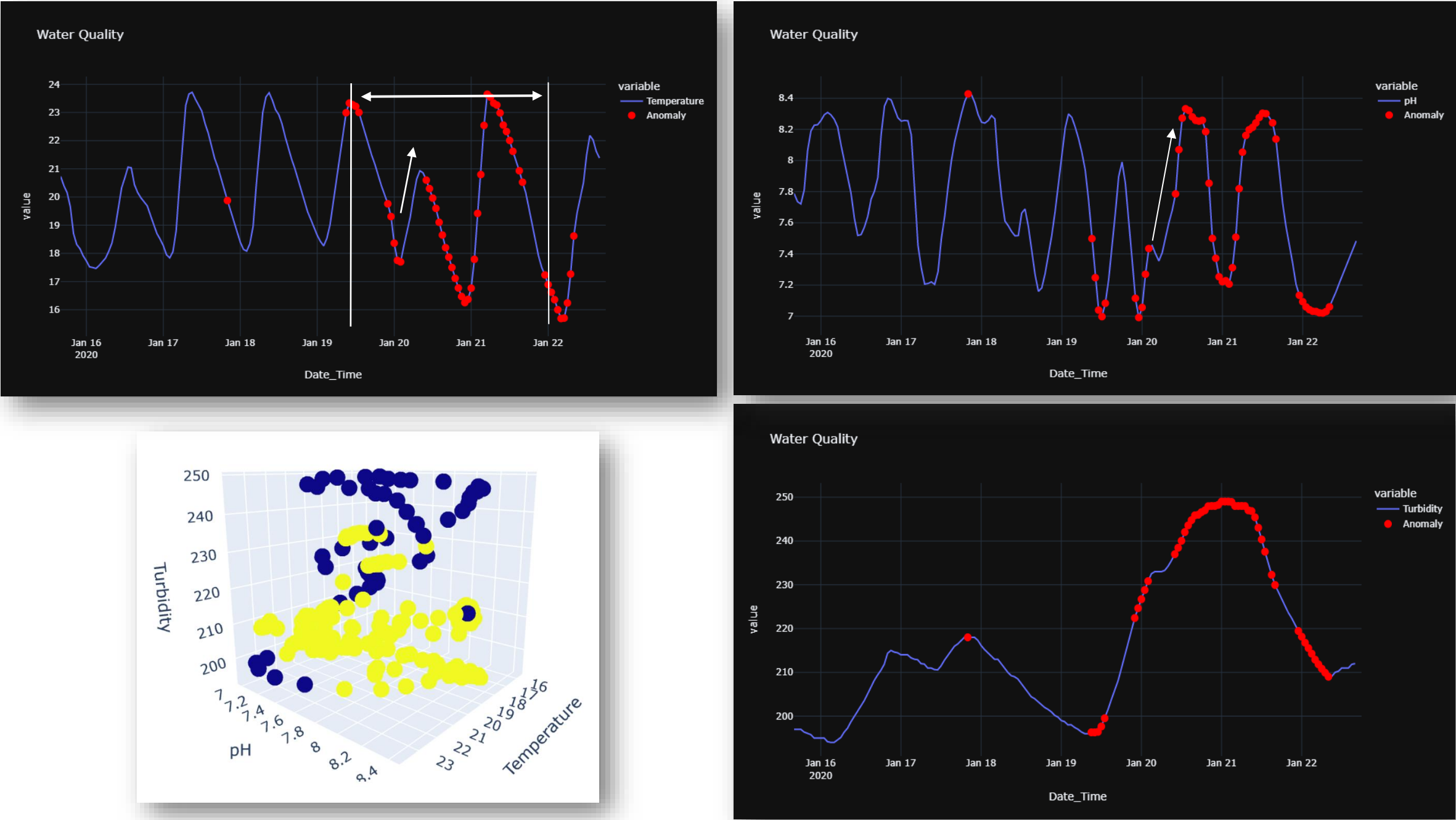
Note: Information about the codes belonging to this section is shown in the third row of Table1.

Table 10. Anomaly Detection

Parameters : n_estimators=100, max_samples='auto', contamination=0.3, random_state=200, n_jobs=-1

Results

Fig. 10. Anomaly Detection

Findings

√ As expected you can also see in these figures that the anomalous points have been detected between the **2020-01-19 22:30** and **2020-01-20 23:00** times. These ranges are marked with white vertical lines.

√ In these points, it is obvious that the Temperature and pH have a positive correlation, unlike the previous days. In other words, the pH does not increase with decreasing Temperature. These sections are indicated by white arrows.

√Turbidity usually increases sharply during heavy rainfall when particles from the soil surface are washed into the pond and the pond bed sediment is re-suspended.

√ In general, it can be concluded that rainfall changes the relationship between these two parameters and also leads to a sharp increase in water turbidity.

4-3   Water Quality Clustering Using MLlib KMeans Algorithms

k-means is one of the most commonly used clustering algorithms that cluster the data points into a predefined number of clusters. The MLlib implementation includes a parallelized variant of the k-means++ method called KMeans. In this section, KMeans has been used for water quality clustering, and its results are described in Table 11.

Note: Information about the codes belonging to this section is shown in the seventh row of Table1.

4-4   Multivariate Time Series Prediction Using Vanilla Transformer

Transformers are a state-of-the-art solution to Natural Language Processing (NLP) tasks. They have based on the Multihead-Self-Attention (MSA) mechanism, in which each token along the input sequence is compared to every other token to gather information and learn dynamic contextual information. The Transformer learns an information-passing graph between its inputs. Because they do not analyze their input sequentially, Transformers largely solve the vanishing gradient problem that hinders Recurrent Neural Networks (RNNs) in long-term prediction. For this reason, Transformers have been applied to datasets with long historical information, including time series forecasting.

In the final part, I tried to use transformers to create a multivariate time series prediction. Of course, it should be noted that due to the time constraints, this analysis is just a proof of concept and most likely not particularly efficient. But the results on this dataset are promising and have led me to study the use of transformers in prediction applications.

And the final point is that, for implementing transformer layers, I have used a basic code of this URL "https://github.com/nklingen/Transformer-Time-Series-Forecasting" and changed it for this task.

The results of this model are illustrated in Table 12.

Note: Information about the codes belonging to this section is shown in the fourth row of Table1.

Note: To run the code please:

1-   **Cd to /codes/transformer_based_prediction /**
2-   **Execute python3 main.py.**

Table 11. Water Quality Clustering

Parameter: Seed = 5 , k = 3

Evaluation Metrics: Silhouette with squared Euclidean Distance = 0.79
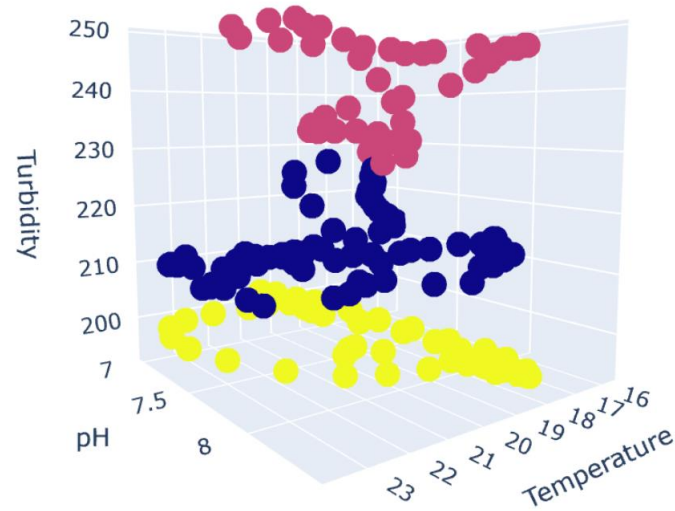
Results

Fig. 11. Water Quality Clustering

Findings

√**This figure shows that the algorithm has clustered the data in three categories, which can be interp reted as clusters containing high, medium, and low water quality, respectively. The center of the clus ters is listed below.**

Cluster Centers:
Cluster 1(High Quality):   [19.98756454, 7.76960577, 198.66727167]
Cluster 2(Medium Quality):   [20.02542647, 7.67621439, 213.98159473]
Cluster 3 (Low Quality):   [19.86443129, 7.84666842, 241.25012599]

Parameter: Epoch = 1000, Training Length=24, Forecasting Window=12, Learning Rate=0.001

Dropout = 1, Optimizer: Adam , Batch Size=1, Frequency=100
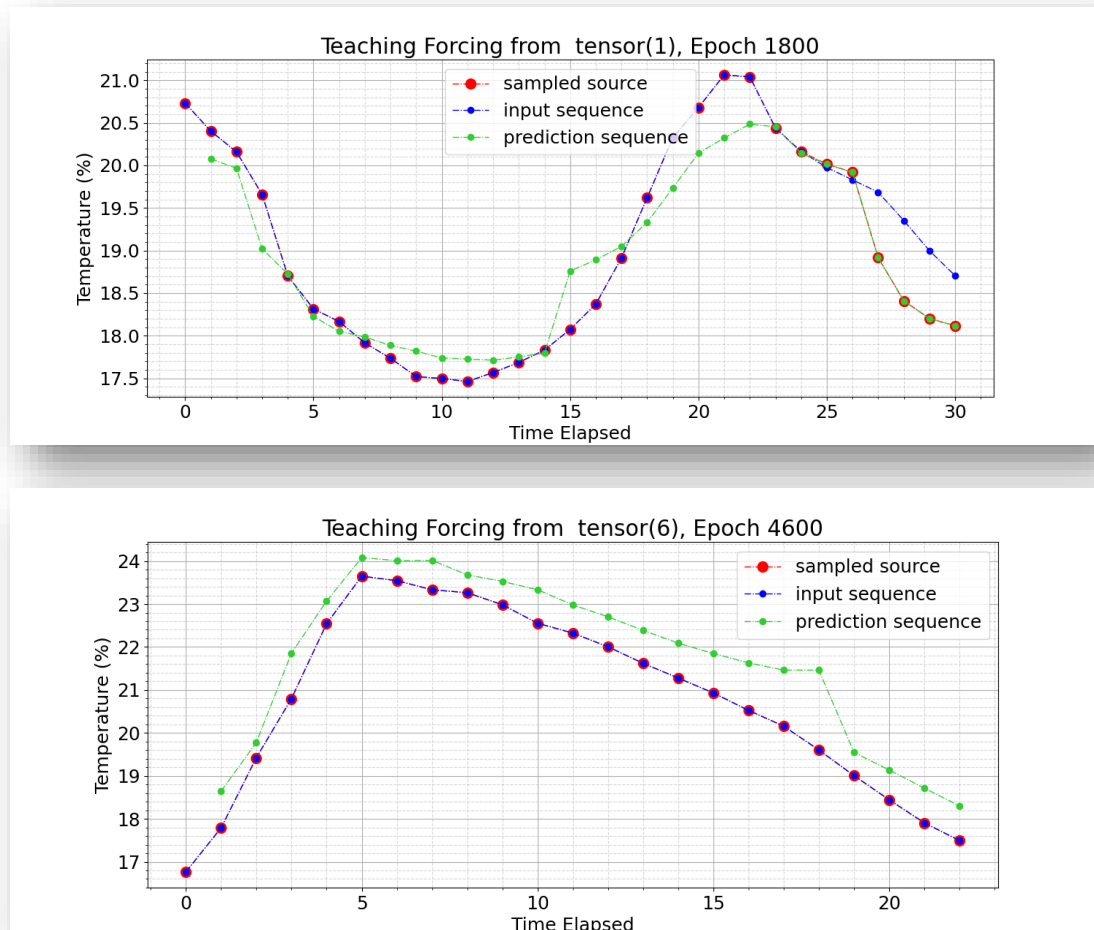
Evaluation Metrics: RMSE= 0.03

Results

Fig. 12. Multivariate Time Series Forecasting

Findings

√ **Based on the above examples, this model almost successfully predicts Temperature parameter in different time steps. In conclusion, the Transformer architecture, which has traditionally been used for NLP and computer vision problems, has a significant potential in time series forecasting.**