National College of Ireland

Configuration Manual

MSc Research Project
Masters in Data analytics

Sai Kiran Goud Doddivenuka
Student ID: x23238224

School of Computing
National College of Ireland

Supervisor: Rejwanul Haque

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Sai kiran Goud Doddivenuka
Student ID:	x23238224
Programme:	Masters in Data analytics **Year:** 2025
Module:	Research Practicum (MSCDAD_A)
Lecturer:	Rejwanul Haque
Submission Due Date:	14-09-2025
Project Title:	Evaluating the Influence of Climate on Water Quality Using Machine Learning Models
Word Count:	1436 **Page Count:** 9

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: Sai Kiran Goud Doddivenuka

Date: 14-09-2025

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):	

Configuration Manual

Sai Kiran Goud Doddivenuka
Student ID: x23238224

1 Project Overview

1.1 Title: Evaluating the Influence of Climate on Water Quality Using Machine Learning Models

1.2 Objective:

Our main goal is to develop and evaluate Machine learning models that can forecast dissolved oxygen levels in the water surface by integrating both water quality parameters and climate variables. The study aims to explore how climate-related data, such as season, air temperature, and month, affect DO levels and whether these features increase prediction accuracy.

1.3 Expected Outcome:

- An integrated approach for predicting climate and water quality that shows increased model accuracy with the inclusion of climate variables.
- Identification of the most influential features affecting dual levels.
- Visual and statistical evidence of the relationship between climate factors and water quality.

2 System Requirements

2.1 Hardware Specifications:

- OS: Windows 11 Pro 64-bit
- CPU: Intel Core i7-11800H @ 2.30GHz
- RAM: 16 GB DDR4
- GPU: NVIDIA RTX 3060 (6 GB VRAM)
- Storage: 512 GB NVMe SSD

2.2 Software Specifications:

- Platform: Anaconda (Python 3.10 environment)
- IDE: Jupyter Notebook launched via Anaconda Navigator or terminal
- Version Control: Git and GitHub

2.3 Python Libraries

- Pandas==2.2.2
- numpy==1.26.4
- matplotlib==3.8.3
- seaborn==0.13.2
- scikit-learn==1.4.2
- xgboost==2.0.3
- imbalanced-learn==0.12.0
- shap==0.44.0
- tensorflow==2.15.0

2.4 Environmental Setup (Anaconda)

Create and activate a dedicated environment, then install dependencies:

```
# Create environment
conda create -n venv python=3.10 -y
conda activate venv

# Install core packages
pip install pandas==2.2.2 numpy==1.26.4 matplotlib==3.8.3
seaborn==0.13.2
pip install scikit-learn==1.4.2 xgboost==2.0.3 imbalanced-
learn==0.12.0 shap==0.44.0
pip install tensorflow==2.15.0

# Launch Jupyter
jupyter notebook
```

3 Dataset Configuration

Primary dataset file: WaterQualityData.csv (Place in the project root directory)

The dataset combines water quality parameters with climate variables for supervised learning tasks (classification and regression of Dissolved Oxygen).

3.1 Expected Columns

Column	Type	Description
Site_Id	categorical	Monitoring site identifier
Unit_Id	categorical	Instrument/Unit identifier (dropped if redundant)
Read_Date	datetime	Measurement date
Time	string	24-hour time (HH:MM)
Water Temp (C)	float	Water temperature in Celsius
Salinity (ppt)	float	Salinity in parts per thousand
pH (standard units)	float	Acidity/alkalinity
Secchi Depth (m)	float	Water clarity proxy
Water Depth (m)	float	Depth at measurement
Dissolved Oxygen (mg/L)	float	Target variable for regression; used to derive labels for classification
Air Temp-Celsius / AirTemp (C)	float	Ambient air temperature (normalized to Celsius)
Air Temp (°F)	float	Original Fahrenheit (converted to Celsius if present)
Humidity	float	Relative humidity (%) if available
Year	int	Year extracted from date
Month	int	Derived temporal feature
Season	category	Derived temporal feature (e.g., Winter/Spring/Summer/Autumn)
Hour	int	Derived from Time
DO_Class	int	Binary label: 1=Safe (≥5 mg/L), 0=Unsafe (<5 mg/L)

3.2 Loading Data

```
import pandas as pd
df = pd.read_csv('WaterQualityData.csv', encoding='utf-8')
print(df.shape)
```

```
print(df.head())
print(df.info())
```

4 Implementation Configuration

This section contains preprocessing, feature engineering, modeling and evaluation configurations used in the ipynb file.

4.1 Preprocessing

- Missing Values: Median for numerical columns, mode fill for categorical columns.
- Temperature Normalization: Combine temperature columns into a single 'AirTemp_C' and convert Fahrenheit to Celsius.
- Label Encoding: LabelEncoder to convert categorical variables
- Class Imbalance: SMOTE is utilized in situations involving both binary and multi-class categorization.
- train/validation/test split: Scikit-learn train_test_split; typical test_size=0.2 and random_state=42.

```
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.model_selection import train_test_split
from imblearn.over_sampling import SMOTE

# Label encoding example
le = LabelEncoder()
if 'Season' in df.columns:
    df['Season_enc'] = le.fit_transform(df['Season'])

# Split
features = [
          'WaterTemp_C',    'AirTemp_C',    'Salinity',    'pH',
'SecchiDepth', 'WaterDepth',
           'Month',     'Site_Id_encoded',     'Season_Winter',
'Season_Spring', 'Season_Summer', 'Season_Fall',
     'Temp_Diff', 'TempSalinityIndex', 'WaterStratification',
'DO_lag1', 'AirTemp_roll3'
]

# Clean model dataset
df_model = df.dropna(subset=features + ['DO', 'DO_Class'])

X = df_model[features]
y_reg = df_model['DO']
y_class = df_model['DO_Class']

# Train/test split
X_train, X_test, y_train_reg, y_test_reg = train_test_split(X,
y_reg, test_size=0.2, random_state=42)
```

```
X_train_c, X_test_c, y_train_c, y_test_c = train_test_split(X,
y_class, test_size=0.2, random_state=42)


# SMOTE
sm = SMOTE(random_state=42)
X_res, y_res = sm.fit_resample(X_train_scaled, y_train)
```

4.2 Feature Engineering

From 'Read_date' and 'Time' columns – Month, Hour, season features are derived.

New Interaction features such as TempSalinityIndex (e.g., Water Temp × Salinity normalized), Temp_Diff (AirTemp_C − Water Temp) are derived.

```
# Temporal features
df['Read_Date'] = pd.to_datetime(df['Read_Date'])
df['Month'] = df['Read_Date'].dt.month
df['Hour']    =    pd.to_datetime(df['Time'],    format='%H:%M',
errors='coerce').dt.hour

def month_to_season(m):
    return ('Winter','Spring','Summer','Autumn')[(m%12)//3]
df['Season'] = df['Month'].apply(month_to_season)


#feature engineering
df['Temp_Diff'] = df['AirTemp_C'] - df['WaterTemp_C']
df['TempSalinityIndex'] = df['WaterTemp_C'] * df['Salinity']
df['WaterStratification']       =       df['WaterDepth']       /
df['SecchiDepth']
df['DO_lag1'] = df.groupby('Site_Id')['DO'].shift(1)
df['AirTemp_roll3']                                                =
df.groupby('Site_Id')['AirTemp_C'].rolling(3).mean().reset_ind
ex(level=0, drop=True)
```

4.3 Modeling Configuration

The following models are implemented in the notebook: Random Forest Classifier, XGBoost Classifier, Logistic Regression, MLP,(Keras/Tensorflow).

```
# Regression model (For regression)
from sklearn.ensemble import RandomForestRegressor
rf = RandomForestRegressor()
rf.fit(X_train, y_train_reg)

# Random Forest (Classification)
```

```
from sklearn.ensemble import RandomForestClassifier
RandomForest= RandomForestClassifier()
RandomForest.fit(X_res, y_res)

# XGBoost
from xgboost import XGBClassifier
XGBoost=XGBClassifier(use_label_encoder=False,eval_metric='log
loss')
XGBoost.fit(X_res, y_res)

# Logistic Regression
from sklearn.linear_model import LogisticRegression
LogisticRegression      =      LogisticRegression(max_iter=1000,
solver='liblinear')
LogisticRegression.fit(X_res, y_res)


#Gradient Boosting
from sklearn.ensemble import GradientBoostingClassifier
GradientBoosting=GradientBoostingClassifier(n_estimators=300,
learning_rate=0.05,          max_depth=6,          subsample=0.8,
random_state=42)

GradientBoosting.fit(X_res,y_res)



# MLP (Keras)
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout
model = Sequential([
        Dense(64,                                activation='relu',
input_shape=(X_resampled.shape[1],)),
    Dropout(0.3),
    Dense(32, activation='relu'),
    Dropout(0.2),
    Dense(3, activation='softmax')  ])

# Compile the model
model.compile(optimizer='adam',
loss='categorical_crossentropy', metrics=['accuracy'])

# Train the model
history = model.fit(X_resampled, y_resampled_cat, epochs=50,
batch_size=32, validation_split=0.2, verbose=1)
```

4.4 Execution Workflow (Jupyter Notebook)

1. Open Anaconda Prompt → activate the environment → run `jupyter notebook`.
2. Open `waterquality1.ipynb`. Execute cells in sequence: imports → data load → preprocessing → EDA → modeling → evaluation.
3. To re-run a specific model only, re-execute the relevant preprocessing and model cells.
4. All random seeds set to 42 to improve reproducibility.

Project Directory Structure

project-root/
├─ waterquality1.ipynb
├─ WaterQualityData.csv
├─ models/
│ ├─ rf_model
│ ├─ Et_model
│ ├─ gb_model
│ ├─ MLP model
├─ outputs/
│ ├─ confusion_matrices/
│ ├─ shap_plots/
│ └─ feature_importance

5. Evaluation & results Configuration

Classification metrics used: Accuracy, precision, recall, F1 score, and ROC-AUC. For multi-class labels, macro/micro-averaged, F1 score and confusion matrix are produced. Regression RMSE, MAE, and R^2 to evaluate continuous DO prediction.

```
from sklearn.metrics import classification_report, confusion_matrix, roc_auc_score
y_pred_rf = rf.predict(X_test_scaled)
print(classification_report(y_test, y_pred_rf))
print(confusion_matrix(y_test, y_pred_rf))
```

5.1 Model explainability

SHAP is used to explain model predictions at the global and local levels, offering insight into which features contribute most to the DO risk classification.

```
import shap
explainer = shap.TreeExplainer(xgb)
shap_values = explainer.shap_values(X_test_scaled)
shap.summary_plot(shap_values, X_test_scaled)
```

6. Troubleshooting

You might run into certain problems when setting up and running the system. Make sure the notebook is running in a Jupyter environment that allows interactive visualizations and try updating SHAP to the most recent version if SHAP plots are not displaying. To fix TensorFlow installation problems (such as incompatibilities), make sure your Python version corresponds to the TensorFlow release that is supported, then use pip install tensorflow== 2.15.0 to reinstall. Reduce batch sizes, use torch.huda.empty_cache() to remove the cache, or shutdown unused notebooks if you encounter GPU memory errors. Before starting Jupyter notebook, make sure the appropriate Anaconda environment is activated in order to fix package import issues.

References

[1] Göz, F., Gürbüz, E. and Özdemir, S., 2019. *Machine learning application of dissolved oxygen prediction in river water quality*. Turkish Journal of Electrical Engineering & Computer Sciences, 27(3), pp.1893-1904.

[2] Hu, Y., Yu, H. and Liu, F., 2024. *Deep learning-based prediction of dissolved oxygen using meteorological and hydrological data*. Environmental Modelling & Software, 170, p.105624.

[3] Qin, Y., 2019. *Comparison of deep learning approaches for dissolved oxygen forecasting in Yangtze River*. Journal of Hydrology, 578, p.124015.

[4] Zhao, W. and Chen, X., 2025. *A hybrid machine learning model integrating KPCA and XGBoost for river water quality prediction*. Journal of Environmental Informatics, 39(2), pp.201-214.

[5] Liu, J., Zhou, T. and Wang, Y., 2025. *A transformer-based framework for spatio- temporal forecasting of water quality under climate change*. Advances in Water Resources, 178, p.104146.

[6] Shen, C., Jager, H.I. and Gunda, T., 2021. *Continental-scale prediction of dissolved oxygen using deep learning*. Environmental Research Letters, 16(10), p.104002.

[7] Pal, M., Deswal, S. and Singh, R., 2017. *Modelling and analysis of water quality parameters using optimized ELM techniques*. Water Science and Technology, 75(5), pp.1054–1064.

[8] Li, X., Xu, H., Zhang, Y. and Wang, X., 2020. Dissolved oxygen prediction in aquaculture based on LSTM neural network. *Aquacultural Engineering*, 91, p.102116.

[9] Kisi, O. and Parmar, K.S., 2016. Application of least square support vector machine and multivariate adaptive regression spline models in long-term prediction of river water quality. *Journal of Hydrology*, 534, pp.104–112.

[10] Lee, S. and Kim, J., 2022. Incorporating climate variability into water quality modelling: An ensemble machine learning approach. *Science of the Total Environment*, 811, p.152284.

[11] Choi, Y., Park, M.J. and Kim, D., 2023. Hybrid deep learning architecture for predicting temporal variation of dissolved oxygen in rivers. *Environmental Science and Pollution Research*, 30(4), pp.4120–4134.

[12] Wang, Y., Gao, J. and Zhang, Q., 2021. Multi-scale convolutional neural network for dissolved oxygen prediction with climate data integration. *Ecological Informatics*, 64, p.101376.
[13] Li, C., Zhang, J., Peng, Y. and Zhao, Z., 2019. Water quality prediction using ensemble learning approaches with climate and hydrological features. *Water Research*, 165, p.114969.