Configuration Manual

MSc Research Project
Data Analytics

Pooja Sree Maniga
Student ID: 23243236

School of Computing
National College of Ireland

Supervisor: Dr. Anu Sahni

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Pooja Sree Maniga …… …………………………………………………………
Student ID:	…23243236…………………………………………………………………………………
Programme:	……Data Analytics…………………………………… **Year:** ……2024……………………..
Module:	…Msc in Research Project…………………………………………………………………
Lecturer:	………Dr. Anu Sahni……………………………………………………………………
Submission Due Date:	…………12/12/2024…………………………………………………………………………
Project Title:	…Comparative Analysis of ML and DL Models to Predict water quality……
Word Count:	………689……………………… **Page Count:** ………………4…………………………………

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: ……Poojasree Maniga…………………………………………………

Date: ……12/12/2024……………………………………………………………………

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

Pooja Sree Maniga
23243236

1 Section 1

The provided notebooks implement machine learning (ML) and deep learning (DL) pipelines for water potability prediction using structured datasets. The ML notebook focuses on models like Random Forest, XGBoost, and Decision Tree, with extensive hyperparameter tuning and evaluation metrics like accuracy and confusion matrices. The DL notebook, on the other hand, leverages deep learning architectures, likely incorporating TensorFlow or PyTorch, and includes provisions for GPU utilization through CUDA settings. Both notebooks use common preprocessing steps such as label encoding and dataset splitting.

2 Section 2

The following Python packages are required to run the notebooks

- **General Libraries**: pandas, numpy, matplotlib, seaborn

  pip install pandas numpy matplotlib seaborn

- **Machine Learning Frameworks**:
  - scikit-learn: For preprocessing, training, and evaluation of ML models.
  - xgboost: For XGBoost-specific implementations.

    pip install scikit-learn xgboost

- **Deep Learning Frameworks**:

  pip install tensorflow torch

3 System Requirements

To run these notebooks efficiently, ensure the system meets the following specifications:

- **Processor**: Minimum quad-core CPU.
- **RAM**: At least 8 GB, 16 GB recommended.

- **GPU**: CUDA-enabled GPU for deep learning tasks.
- **Storage**: At least 1 GB of free disk space for datasets and intermediate files.

4 Hyperparameters

Machine Learning models:

- **Grid Search Parameters**:
  - **Random Forest**: n_estimators, max_depth, entropy
  - **XGBoost**: n_estimators
  - **Decision Tree**: max_depth
  - **Logistic Regression**: Regularization parameter C

Deep Learning models:

- **Hyperparameters**:
  - Batch size
  - Learning rate
  - Number of epochs

5 Execution

1) Load dataset

```
df_sample = pd.read_csv('sampled_dataset.csv', header=None)
```

2) Perform necessary preprocessing steps such as Label encoding for categorical data

```
label_encoder = LabelEncoder()

for column in X.columns:
    X[column] = label_encoder.fit_transform(X[column].astype(str))

print(X.head())
```

3) Split data into training and test sets

```
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
#train and validation split on validation data
X_train, X_val, y_train, y_val = train_test_split(X_train, y_train, test_size=0.2, random_state=42)
```

4) **Select and Train Models**

   - **For example, training random forest model**

```
# Define the parameter grid for n_estimators
param_grid = {
    'n_estimators': [5, 10, 50, 100, 300],
    'max_depth': [ 5, 10, 20,30,50,100, None],
    'criterion': ['entropy', 'gini']
}

# Initialize the Random Forest classifier
rf_model = RandomForestClassifier(random_state=42)

# Perform GridSearchCV
grid_search = GridSearchCV(estimator=rf_model, param_grid=param_grid, cv=5, scoring='accuracy', n_jobs=-1, verbose=1)
grid_search.fit(X_train, y_train)
```

- For Logistic regression model inialising and training,

```
from sklearn.linear_model import LogisticRegression

# Define the parameter grid for C (regularization strength)
param_grid_lr = {'C': [0.01, 0.1, 1, 10, 100]}

# Initialize the Logistic Regression classifier
lr_model = LogisticRegression(random_state=42, max_iter=1000)

# Perform GridSearchCV
grid_search_lr = GridSearchCV(estimator=lr_model, param_grid=param_grid_lr, cv=5, scoring='accuracy', n_jobs=-1, verbose=1)
grid_search_lr.fit(X_train, y_train)
```

- Initalising and training deep learning models

```
# Define the LSTM model
class BinaryClassificationLSTM(nn.Module):
    def __init__(self, input_size, hidden_size, num_layers):
        super(BinaryClassificationLSTM, self).__init__()
        self.lstm = nn.LSTM(input_size, hidden_size, num_layers, batch_first=True)
        self.fc = nn.Linear(hidden_size, 1)  # Fully connected layer for output
        self.sigmoid = nn.Sigmoid()          # Sigmoid activation for binary classification

    def forward(self, x):
        # LSTM outputs
        _, (hidden, _) = self.lstm(x)  # Get the last hidden state
        out = hidden[-1]               # Take the hidden state of the last LSTM layer
        out = self.fc(out)             # Fully connected layer
        out = self.sigmoid(out)        # Apply sigmoid activation
        return out
```

5) For deep learning and machine learning models all the plots and confusion matrices including hyper parameter tuning plots are saved to folder.

6) Results can be seen in uploaded notebooks.

References

Ghosh, H., Tusher, M.A., Rahat, I.S., Khasim, S. and Mohanty, S.N., 2023, February. Water quality assessment through predictive machine learning. In *International Conference on Intelligent Computing and Networking* (pp. 77-88). Singapore: Springer Nature Singapore.

Krushna, Bode Vamsi, and D. Sasikala. "Comparative Analysis of Machine Learning Models for Water Quality Prediction." In *2024 Fourth International Conference on Advances in Electrical, Computing, Communication and Sustainable Technologies (ICAECT)*, pp. 1-6. IEEE, 2024.

Lawal, Z.K., Aldrees, A., Yassin, H., Dan'azumi, S., Naganna, S.R., Abba, S.I. and Sammen, S.S., 2024. Optimized Ensemble Methods for Classifying Imbalanced Water Quality Index Data. *IEEE Access*.