Configuration Manual

MSc Research Project
Data Analytics

Melin Mary Lalu
x23185104

School of Computing
National College of Ireland

Supervisor: Dr.Abid Yaqoob

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name: Melin Mary Lalu……

Student ID: X23185104……..……

Programme: Msc. Data Analytics…… **Year:** 2024-2025

Module: Msc. Research Project……..……

Lecturer: Abid Yaqoob……..……

Submission Due Date: 12-12-2024……..……

Project Title: Employing Advanced TCN Model to Predict the Success of Bank Telemarketing in Long-term Deposit Subscription

Word Count: …… **Page Count:** 5

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: Melin ……

Date: 10-12-2024 ……

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):	

Employing Advanced TCN Model to Predict the Success of Bank Telemarketing in Long-term Deposit Subscription

Configuration Manual

Melin Mary Lalu
X23185104

1. Introduction

The configuration manual clearly explains the tools and technologies used to implement this research project. It includes information about the tools, technologies, libraries, and Model predictions for the success of telemarketing.

2. Experimental Setup

Hardware Specifications:

- **Device:** Acer Aspire 7
- **Processor:** AMD Ryzen 5
- **Memory:** 8 GB RAM
- **Operating System:** Windows 11

Software Specifications

- **Python Environment:** Python 3.11
- **Editor:** Jupyter Notebook

3. Software and Libraries Used for Implementation

I. Programming Environment

- Anaconda

The Anaconda platform is the most popular way to learn and use Python for data science and machine learning

- Jupiter notebook

It is an open-source web-based environment for all programming languages.

II. Libraries Used

The following libraries were used for model training, testing, explainability, and real-time monitoring in the project:

- Data processing: pandas, NumPy
- Machine Learning Models: Scikit-learn
- Deep Learning Models: TensorFlow
- Visualization Libraries: Seaborn, matplot

4. Software Setup and Implementation Steps

Step 1: Installation of Anaconda and Jupytor Notebook

- Anaconda was downloaded and installed
- Jupytor Notebook was used for codding and data exploration

Step 2: Dataset Preparation

- Downloaded the data set from the online platform UCI machine learning repository and stored it locally.

Step 3: Importing necessary libraries

- The following Python libraries are imported to implement the model prediction for the success of bank telemarketing.

```
import pandas as pd
import numpy as np
from sklearn.impute import SimpleImputer
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.metrics import accuracy_score, f1_score, confusion_matrix, roc_auc_score
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv1D, Dense, Flatten
from tensorflow.keras.callbacks import EarlyStopping
from sklearn.model_selection import RandomizedSearchCV
from tensorflow.keras.layers import BatchNormalization
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.models import load_model
import pickle
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC
from xgboost import XGBClassifier
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from sklearn.metrics import accuracy_score, f1_score, roc_auc_score, confusion_matrix
```

Fig 1. Importing Libraries

Step 4: Loading and Preprocessing the Data

- The dataset Bank_Telemarketing_data.csv is loaded using the pandas library in Python.
- The Preprocessing steps are Handling missing values, Encoding the categorical values and Normalize the numeric features.

```
# Load Data
data = pd.read_csv('Bank_Telemarketing_data.csv')
```

Fig 2. Load Data

```
# Replace 'unknown' with NaN and handle missing values
data.replace('unknown', np.nan, inplace=True)
data.dropna(inplace=True)

# Encode categorical variables
label_encoder = LabelEncoder()
for col in ['job', 'marital', 'education', 'default', 'housing', 'loan', 'contact', 'month', 'poutcome']:
    data[col] = label_encoder.fit_transform(data[col])

# Convert target variable 'y' to binary
data['y'] = data['y'].apply(lambda x: 1 if x == 'yes' else 0)
```

Fig 3. Data Preprocessing

Step 5: Splitting and scaling the data

- The dataset was split into training and testing datasets using an 80-20 split.
- Features were scaled using StandardScaler.

```
# Splitting data into features and target
X = data.drop('y', axis=1)
y = data['y']

# Split 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)

# Standardize the numerical columns
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

Fig 4. Splitting and Scaling

Step 6: Model Training

- The TCN (Temporal Convolutional Network) Model is trained using the bank telemarketing dataset.

```
# Compiling model
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

# Training the model
early_stopping = EarlyStopping(monitor='val_loss', patience=3)
history = model.fit(X_train_tcn, y_train, epochs=10, validation_data=(X_test_tcn, y_test), callbacks=[early_stopping])
```

Fig 5 . Train the TCN model

Step 7: Model Evaluation

- Evaluate the predicted model by the metrics like Accuracy, F1 Score, AUC-ROC value

```
# Evaluating the model
y_pred = (model.predict(X_test_tcn) > 0.5).astype("int32")
print("Accuracy:", accuracy_score(y_test, y_pred))
print("F1 Score:", f1_score(y_test, y_pred))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("AUC-ROC Score:", roc_auc_score(y_test, y_pred))
```

Fig 6. Evaluate the Model

Step 8: Model Comparison

- Model Comparison is made with the ML and DL models like Decision Tree, ANN, KNN, SVM, and Naïve Bays.
- Compared with the matrics of Accuracy, F1 score, and AUC-ROC score.
- Visualize the values by Bar plot for more understanding.

```
def evaluate_model(name, y_true, y_pred):
    accuracy = accuracy_score(y_true, y_pred)
    f1 = f1_score(y_true, y_pred)
    auc = roc_auc_score(y_true, y_pred)
    print(f"{name} - Accuracy: {accuracy:.2f}, F1-Score: {f1:.2f}, AUC-ROC: {auc:.2f}")
    return accuracy, f1, auc

results = {}
results['ANN'] = evaluate_model("ANN", y_test, ann_pred)
results['TCN'] = evaluate_model("TCN", y_test, tcn_pred)
results['Decision Tree'] = evaluate_model("Decision Tree", y_test, dt_pred)
results['KNN'] = evaluate_model("KNN", y_test, knn_pred)
results['Naive Bayes'] = evaluate_model("Naive Bayes", y_test, nb_pred)
results['SVM'] = evaluate_model("SVM", y_test, svm_pred)
```

Fig 7. Evaluate the compared models

```
# Extract metrics for visualization
models = list(results.keys())
accuracies = [res[0] for res in results.values()]
f1_scores = [res[1] for res in results.values()]
auc_roc_scores = [res[2] for res in results.values()]

#Accuracy
plt.figure(figsize=(10, 6))
plt.bar(models, accuracies, color='blue', alpha=0.7)
plt.title('Model Comparison - Accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Models')
plt.ylim(0, 1)
plt.xticks(rotation=45)
plt.show()
```

Fig 8. Visualize the Accuracy of Models

Conclusion

The configuration manual presents a framework protocol for coordinating, implementing, and overseeing the project related to the MSc Research effort. It describes the structural format required when documenting configurations, stressing those submission protocols that should always be followed when preparing a paper; such issues as referencing and plagiarism. It includes areas for project-specific information, to make it optimized for various kinds of research fields. The separated portions of the manual also help in such optimization, as it provides an opportunity to change, edit, or extend required aspects of the Manual, for example, installation rules and steps, requirements to the environment, deployment procedures, etc., to fit requirements of particular research. Thus, by paying much attention to the documented structure, the manual guarantees the realization of the research outcome in the future.

Despite the high level of pre-structuring that has been accomplished, the use of the presented template requires accurate implementation in the context of the particular project. For future upgrade options, tips for specific configuration arrangements, more directions related to the problem-solving process, or standardized checklists for maintaining quality might be added. In conclusion, this configuration manual helps make it predictable and attractive in this research project.

References

Anaconda, 2024. *Download Anaconda*. Available at: https://www.anaconda.com/download [Accessed 5 Nov. 2024].