National College of Ireland

Configuration Manual

MSc Research Project
MSc in Fintech

Huy Hoang Nguyen
Student ID: 23429461

School of Computing
National College of Ireland

Supervisor: Faithful Onwuegbuche

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Huy Hoang Nguyen		
Student ID:	23429461		
Programme:	MSc in Fintech	**Year:**	2024-2025
Module:	MSc (Research) Practicum/Internship Part 2		
Lecturer:	Faithful Onwuegbuche		
Submission Due Date:	11/08/2025		
Project Title:	AI in Loan prediction		
Word Count:	910	**Page Count:**	6

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: Huy Hoang Nguyen

Date: 11/08/2025

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

Huy Hoang Nguyen
Student ID: 23429461

1 System Requirements

1.1 Hardware

- **Processor:** Any modern CPU (Intel i5/i7 or AMD equivalent).
- **GPU:** NVIDIA T4 GPU or equivalent (required for efficient training in Google Colab).
- **RAM:** Minimum 8 GB (16 GB recommended).
- **Storage:** Minimum 20 GB free space (for dataset and model artifacts).

1.2 Software

- **Operating System:** Compatible with Google Colab (Linux-based backend).
- **Python Version:** 3.10
- **Execution Environment:** Google Colab Notebook.

1.3 Dataset Requirements

- Must contain **numerical and categorical features** relevant to loan records.
- Must include:
  - **Continuous target variable** (loan amount).
  - **Financial attributes** (income, credit score).
  - **Demographic attributes** (age, gender, location).
  - **Loan-related attributes** (loan term, interest rate).
- Dataset should be representative of real-world loan applications and collected under **stable economic conditions**

1.4 Python Libraries

Install required libraries in the first Colab cell:

```
!pip install category_encoders xgboost kagglehub
```

Imported libraries used in the notebook:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

```
import seaborn as sns
from IPython.display import display

from sklearn.utils import resample
import category_encoders as ce
from sklearn.preprocessing import RobustScaler, PolynomialFeatures
from sklearn.model_selection import train_test_split

from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score, median_absolute_error
from sklearn.model_selection import cross_val_score
from sklearn.neighbors import KNeighborsRegressor
from sklearn.tree import DecisionTreeRegressor
from sklearn.svm import SVR
from xgboost import XGBRegressor

import time
import joblib
import warnings
warnings.filterwarnings("ignore")
```

2 Configuration & Execution Steps

Your second section. Change the header and label to something appropriate.

2.1 Set Runtime Environment

1. Open the Google Colab notebook.
2. Navigate to **Runtime → Change runtime type**.
3. Set **Hardware accelerator** to **GPU**.
4. Select **GPU type: T4**.
5. Save and restart runtime if prompted.

2.2 Load Dataset

The dataset is downloaded and loaded using the kagglehub package:

```
import kagglehub
import pandas as pd

# Download latest version of the dataset
path = kagglehub.dataset_download("taweilo/loan-approval-classification-data")
print("Path to dataset files:", path)

# Load dataset from cache
data = pd.read_csv('/root/.cache/kagglehub/datasets/taweilo/loan-approval-classification-data/versions/1/loan_data.csv')
```

```
display(data.head())
```

2.3 Exploratory Data Analysis (EDA)

Run the EDA section of the notebook to:

- Inspect data dimensions and feature types.
- Identify missing or inconsistent data.
- Visualize feature distributions and relationships using plots (histograms, correlation heatmaps).

2.4 Data Preprocessing

Before training, the dataset undergoes these preprocessing steps:

- **Outlier Handling:** Detect and treat outliers in numerical features to reduce skewness.
- **Rejected Loan Processing:** Process records of rejected loans by filtering or imputing missing data to maintain dataset consistency.
- **Feature Scaling:** Apply RobustScaler to normalize numerical features and improve model performance.

These steps prepare the data for effective model training.

2.5 Model Training and Evaluation

The system trains six regression models:

- Linear Regression
- Random Forest Regressor
- Decision Tree Regressor
- K-Nearest Neighbors Regressor
- Support Vector Regressor
- XGBoost Regressor

2.6 Training Process:

- Models are trained sequentially within a loop for consistent processing.
- Polynomial feature transformation is applied only for Linear Regression to model non-linearities.
- Performance metrics and training times are recorded for comparison.

Representative training loop:

```
trained_models = {}
predictions = {}
results = []

for name, model in models.items():
    if name == "Linear Regression":
        poly = PolynomialFeatures(degree=2, include_bias=False)
        X_train_mod = poly.fit_transform(X_train_scaled)
        X_val_mod = poly.transform(X_val_scaled)
    else:
        X_train_mod = X_train_scaled
        X_val_mod = X_val_scaled

    start_time = time.time()
```

```
    model.fit(X_train_mod, y_train)
    training_time = time.time() - start_time

    trained_models[name] = (model, X_train_mod, X_val_mod)
    y_val_pred = model.predict(X_val_mod)
    predictions[name] = y_val_pred

    train_score = model.score(X_train_mod, y_train)
    test_score = model.score(X_val_mod, y_val)
    mse = mean_squared_error(y_val, y_val_pred)
    rmse = np.sqrt(mse)
    mae = mean_absolute_error(y_val, y_val_pred)
    medae = median_absolute_error(y_val, y_val_pred)
    r2 = r2_score(y_val, y_val_pred)
    cv_r2 = cross_val_score(model, X_train_mod, y_train, cv=5, scoring='r2').mean()

    results.append({
        'Model': name,
        'Train R² Score': train_score,
        'Test R² Score': test_score,
        'Mean Squared Error': mse,
        'Root Mean Squared Error': rmse,
        'Mean Absolute Error': mae,
        'Median Absolute Error': medae,
        'R² Score': r2,
        'Cross-Validation R²': cv_r2,
        'Training Time (s)': training_time
    })
```

3 Source Code & Deployment Access

The full implementation, including dataset download, preprocessing, EDA, model training, and evaluation, is available in the Google Colab notebook:

Colab Notebook:
https://colab.research.google.com/drive/1ZIK5y_MKIF58XSaL1EmzbzmBWLLBf4ez?usp=sharing