National College of Ireland

Research Configuration Manual

MSc Research Project
Data Analytics

Prajwal Suhas Pusadkar
Student ID: 23304367

School of Computing
National College of Ireland

Supervisor: Prof. Hicham Rifai

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Prajwal Suhas Pusadkar
Student ID:	23304367
Programme:	Data Analytics
Year:	2024
Module:	MSc Research Project
Supervisor:	Mr. Hicham Rifai
Submission Due Date:	15/09/2025
Project Title:	Research Configuration Manual
Word Count:	705
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:	Prajwal Suhas Pusadkar
Date:	15th September 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):	

Research Configuration Manual

Prajwal Suhas Pusadkar
23304367

1 Introduction

This configuration manual provides step-by-step guidance to set up, run, and maintain the Fair and Interpretable Credit Risk Modelling with Multi-Agent Architecture and SHAP system. The system uses four agents — Risk Scoring, Credit Limit Recommendation, Loan Decisioning, and Explainability — to replicate real-world credit approval workflows. It integrates XGBoost for prediction, SHAP for model explainability, Fairlearn for fairness auditing, and Streamlit for an interactive dashboard.

Following this manual will enable users to configure the environment, prepare the Taiwan Credit Card Default dataset, train and deploy the model, perform fairness audits, and run real-time or batch loan evaluations with full transparency and reproducibility.

2 System Configuration

2.1 Hardware Configuration

Below figure depicts the hardware configuration used in my research.

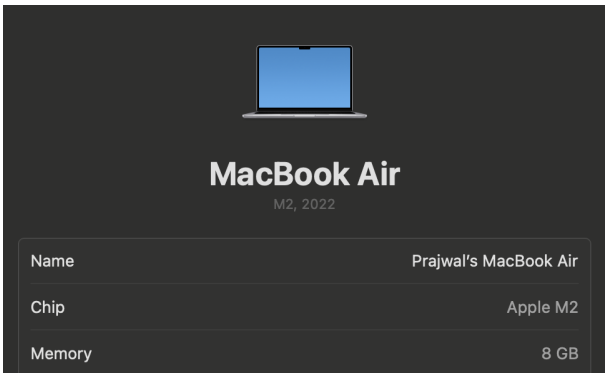


Figure 1: Hardware Configuration

Recommended for Optimal Performance

- CPU: 6+ core Intel i7 / AMD Ryzen
- RAM: 16 GB or higher
- GPU: NVIDIA RTX series (e.g., RTX 3060 or better, CUDA-enabled)
- Storage: SSD with at least 20 GB free space

2.2 Software Configuration

- Operating System - macOS Monterey Sequoia 15.5
- Programming Language - Python 3.11
- Development Tool - Visual Studio Code
- Core Libraries and Dependencies:
    1. pandas – Data manipulation
    2. numpy – Numerical computing
    3. scikit-learn – Preprocessing and ML utilities
    4. xgboost – Core predictive model
    5. shap – Explainability framework
    6. fairlearn – Fairness auditing
    7. streamlit – Dashboard interface
    8. matplotlib, seaborn – Visualizations

2.3 Runtime Environment Setup

Install Dependencies:

- pip install -r requirements.txt

Verify Installation:

- python –version
- pip list

2.4 Overleaf

This research report was created using Overleaf, a cloud document editing tool. Overleaf uses the LaTeX language for document formatting which is very common for document creation because of its ease of use as well as ease of viewing the document. Provided below is a screenshot showing Overleaf's UI.

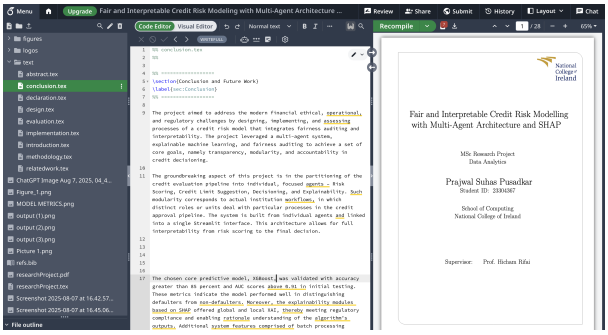


Figure 2: Overleaf

3 Data Preparation

3.1 Dataset Acquisition

Dataset: Taiwan Credit Card Default dataset

Source: Publicly available financial dataset containing 30,000 client records with demographic and financial attributes.

Download Link: UCI Repository – Taiwan Credit Card Default Dataset

File format: CSV

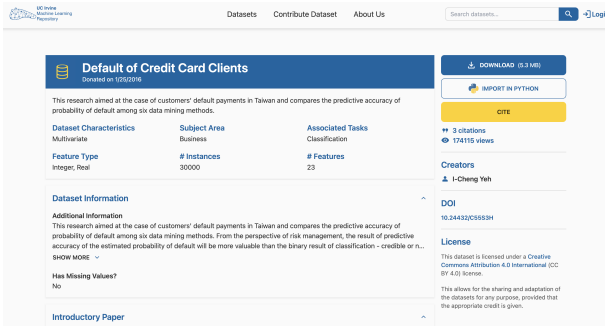


Figure 3: Dataset Page

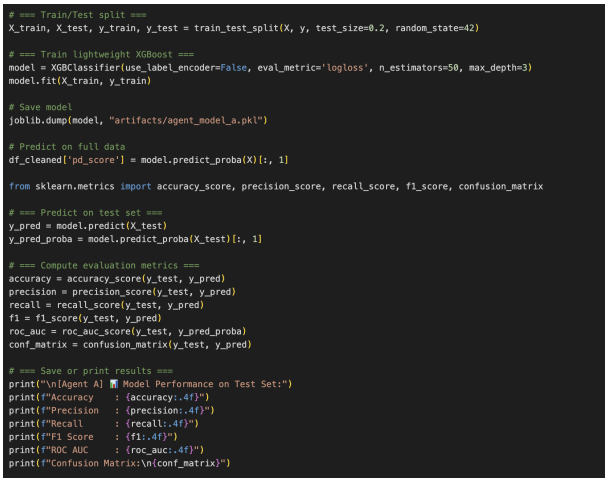
4 Model Development

4.1 Agent A – Risk Scoring (XGBoost Classifier)

Purpose - Agent A predicts the Probability of Default (PD) for each applicant using an XGBoost classifier trained on the processed Taiwan Credit Card Default dataset.

Model Training Steps:

1. Load preprocessed dataset from data/processed/credit_processed.csv
2. Split into train/test sets (80/20) with stratified sampling
3. Apply class weighting to address imbalance
4. Tune hyperparameters via grid search
5. Train final model and serialize to models/xgb_risk_model.pkl



```
# === Train/Test split ===
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# === Train lightweight XGBoost ===
model = XGBClassifier(use_label_encoder=False, eval_metric='logloss', n_estimators=50, max_depth=3)
model.fit(X_train, y_train)

# Save model
joblib.dump(model, "artifacts/agent_model_a.pkl")

# Predict on full data
df_cleaned['pd_score'] = model.predict_proba(X)[:, 1]

from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, confusion_matrix

# === Predict on test set ===
y_pred = model.predict(X_test)
y_pred_proba = model.predict_proba(X_test)[:, 1]

# === Compute evaluation metrics ===
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred)
recall = recall_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred)
roc_auc = roc_auc_score(y_test, y_pred_proba)
conf_matrix = confusion_matrix(y_test, y_pred)

# === Save or print results ===
print("\n[Agent A] 📊 Model Performance on Test Set:")
print(f"Accuracy    : {accuracy:.4f}")
print(f"Precision   : {precision:.4f}")
print(f"Recall      : {recall:.4f}")
print(f"F1 Score    : {f1:.4f}")
print(f"ROC AUC     : {roc_auc:.4f}")
print(f"Confusion Matrix:\n{conf_matrix}")
```

Figure 4: Model Building

4.2 Agents B, C, and D Summary

The system's decision-making process after risk scoring is handled by three interconnected agents:

- **Agent B – Credit Limit Recommendation:** Uses PD scores from Agent A, repayment history, and financial heuristics to recommend an optimal credit limit. The logic combines configurable business rules with statistical insights, ensuring limits are competitive yet risk-aware.

- **Agent C – Loan Decisioning:** Approves or rejects loan applications based on PD scores, repayment delay history, and preset policy thresholds. The decision logic is stored in a configuration file, enabling threshold adjustments without retraining the model.

- **Agent D – Explainability and Fairness:** Generates global (dataset-wide) and local (applicant-specific) explanations using SHAP values. These visualizations identify the most influential features for each decision. Agent D also integrates with Fairlearn to conduct fairness audits across demographic groups (e.g., gender, education, marital status), producing structured reports for compliance and bias monitoring.

Together, these agents form the decision layer of the multi-agent system, translating raw PD scores into actionable, explainable, and auditable credit decisions.

4.3 Fairness Auditing (Integrated with Agent D)

The system incorporates Fairlearn to evaluate demographic fairness in loan approvals. Fairness auditing is triggered automatically after batch predictions, enabling compliance teams to detect and quantify disparities.

Key Features:

Monitored Attributes: Gender (SEX), Education Level (EDUCATION), Marital Status (MARRIAGE).

- Demographic Parity Difference (DPD) – Measures the difference in approval rates between groups.
- Equalized Odds Difference (EOD) – Measures the difference in error rates between groups.
- Output Reports: CSV and PDF summaries with per-group approval rates, recall, and precision metrics.

```python
# Helper function
def compute_fairness_metrics(df, group_col):
    metric_frame = MetricFrame(
        metrics={
            'Accuracy': accuracy_score,
            'Recall': recall_score,
            'Precision': precision_score,
            'Selection Rate': selection_rate
        },
        y_true=df['approved'],
        y_pred=df['approved'],
        sensitive_features=df[group_col]
    )

    dp_diff = demographic_parity_difference(y_true, y_pred, sensitive_features=df[group_col])
    eo_diff = equalized_odds_difference(y_true, y_pred, sensitive_features=df[group_col])

    return metric_frame.by_group, dp_diff, eo_diff

# 📊 Evaluate across Gender
gender_metrics, dp_sex, eo_sex = compute_fairness_metrics(df, 'SEX')
print("\n📊 Fairness by GENDER (SEX):")
print(gender_metrics)
print(f"→ Demographic Parity Difference: {dp_sex:.3f}")
print(f"→ Equalized Odds Difference: {eo_sex:.3f}")

# 📊 Evaluate across Marital Status
marital_metrics, dp_marriage, eo_marriage = compute_fairness_metrics(df, 'MARRIAGE')
print("\n📊 Fairness by MARITAL STATUS:")
print(marital_metrics)
print(f"→ Demographic Parity Difference: {dp_marriage:.3f}")
print(f"→ Equalized Odds Difference: {eo_marriage:.3f}")

# 📊 Evaluate across Education
education_metrics, dp_edu, eo_edu = compute_fairness_metrics(df, 'EDUCATION')
print("\n📊 Fairness by EDUCATION:")
print(education_metrics)
print(f"→ Demographic Parity Difference: {dp_edu:.3f}")
print(f"→ Equalized Odds Difference: {eo_edu:.3f}")
```

Figure 5: Fairness Analysis Code Snippet

5 Deployment and Execution

5.1 Streamlit Dashboard Implementation

The system is deployed via a Streamlit web application, providing an interactive interface for real-time and batch loan application evaluations.

Core Features:

- CSV Upload: Users can upload applicant datasets for processing.
- Real-Time Scoring: Uploaded data is passed sequentially through Agents A–D for risk scoring, credit limit recommendation, decisioning, and explanation.
- SHAP Visualization: Displays both global (summary plot) and local (force plot) explanations directly in the dashboard.
- Fairness Auditing: Runs demographic fairness checks automatically after predictions, displaying key metrics and allowing report downloads.
- Batch Results Export: Users can download CSV reports containing PD scores, approval decisions, recommended limits, SHAP summaries, and fairness metrics.

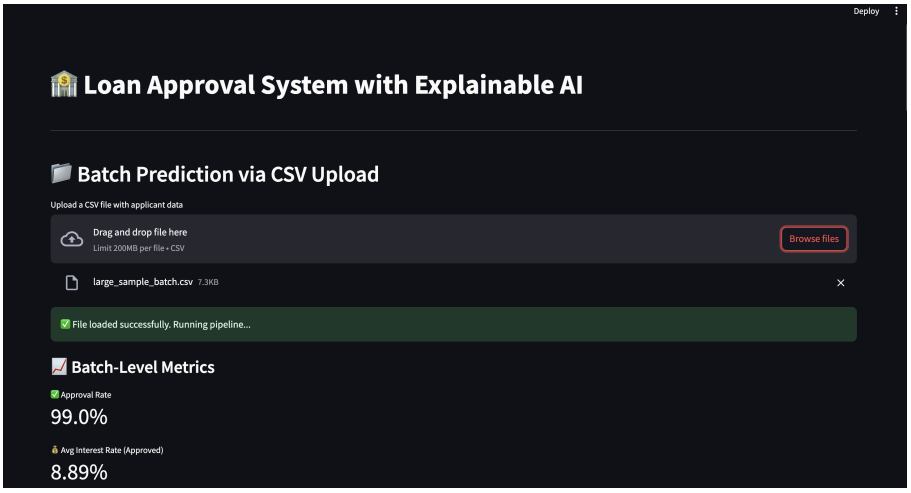


Figure 6: Frontend

Usage Workflow:

1. Start Streamlit with the launch command.
2. Upload the applicant dataset in CSV format.
3. View the PD scores, loan decisions, and recommended credit limits.
4. Review SHAP explanations and fairness audit results.
5. Download the final report for record-keeping.