National College of Ireland

Configuration Manual

MSc Research Project
Data Analytics

Dikshitha Arshanapalli
Student ID: 23395591

School of Computing
National College of Ireland

Supervisor: Prof. Dr Anu Sahni

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Dikshitha Arshanapalli
Student ID:	23395591
Programme:	Data Analytics **Year:** 25-2026
Module:	Research Practicum
Lecturer:	Prof. Dr Anu Sahni
Submission Due Date:	11/12/2025
Project Title:	Advancing Multi-Class Diabetes Risk Prediction Using Machine Learning, Deep Learning, and Explainable AI: A Data Analytics Approach to Imbalanced Public Health Survey Data
Word Count:	1000 **Page Count:** 10

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: Dikshitha Arshanapalli

Date: 11/12/2025

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

Dikshitha Arshanapalli
Student ID: 23395591

1 Introduction

Early identification of diabetes and prediabetes which is essential to prevent long term complications and improve population health outcomes. However, most previous studies used binary prediction and limiting the ability to detect intermediate risk states. This research develops a comprehensive multiclass diabetes prediction framework using CDC BRFSS 2015 dataset includes preprocessing, engineered glycaemic features, composite health scores and interaction-based features to enhance predictive signal. Class imbalance was addressed using SMOTE, class weighting balanced boosting. LightGBM, Tuned LightGBM, CatBoost, Deep Neural Network (DNN), 1D Convolutional Neural Network (1D-CNN), and ensemble methods (stacking and soft voting) were implemented and compared using evaluation using accuracy, macro F1, weighted F1, and per-class recall shows that baseline LightGBM delivered better overall performance successfully predicted all three classes. while tuned LightGBM improved diabetes recall and prediabetes improved from 0.00 to 2.7%. However, it remained most difficult class to detect low representation. SHAP explainability highlights the strong influence of engineered features such as HbA1c_Estimated, Glucose_Risk_Proxy, CGM_Variability_Score, BMI, and General Health. Overall, the research demonstrates feasibility of explainable multi class diabetes risk prediction and identifies key areas such as clinical data, advanced imbalance handling and transformer-based models for further future improvement.

The configuration manual outlined below ensures reproducibility, scalability and explainability of the full analytical pipeline.

2 System Configuration

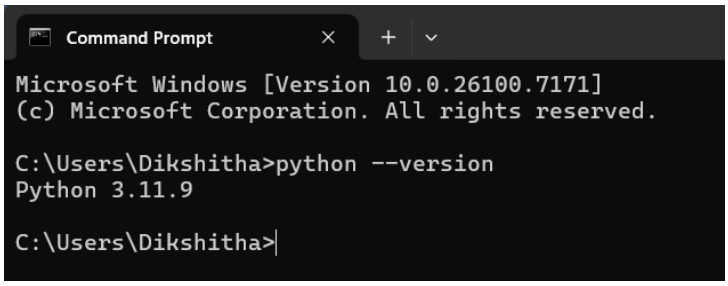
2.1 Recommended Hardware

Component	**Minimum Requirement**
Processor	Intel i5 / AMD Ryzen 5
RAM	16 GB (Recommended 32 GB for DL models)
Storage	256 GB SSD
GPU (Optional)	NVIDIA GPU with CUDA support
Architecture	64-bit

2.2 Software Requirements – Operating System

- **Operating System:** Microsoft Windows 10/11 (64-bit)
- **Development Environment:** Jupyter Notebook

- **Python Version Used:** Python 3.11.9



2.3 Required Libraries

The following Python libraries were required

Category	Libraries
Core	pandas, numpy
Visualization	matplotlib, seaborn
ML Models	SVM, LogisticRegression, XGBoost, LightGBM, CatBoost, Stacking
Deep Learning	TensorFlow, Keras
Preprocessing	StandardScaler, RobustScaler, label_binarize
Sampling	SMOTE, ADASYN, SMOTETomek
Evaluation	accuracy, precision, recall, f1, ROC-AUC, confusion_matrix
Optimization	Adam, AdamW
Utilities	warnings, gc, backend

Import Libraries & Load Dataset

[76]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
from sklearn.model_selection import train_test_split
import xgboost as xgb
import shap

from sklearn.model_selection import train_test_split, StratifiedKFold
from sklearn.preprocessing import StandardScaler, RobustScaler
from sklearn.metrics import (accuracy_score, precision_score, recall_score, f1_score, roc_auc_score,classification_report, confusion_matrix,
balanced_accuracy_score,precision_recall_fscore_support,matthews_corrcoef, cohen_kappa_score, roc_curve, precision_recall_curve, auc)
from sklearn.preprocessing import label_binarize

# Sampling
from imblearn.over_sampling import SMOTE, ADASYN
from imblearn.combine import SMOTETomek

from sklearn.svm import SVC
import lightgbm as lgb
from catboost import CatBoostClassifier
from lightgbm import LGBMClassifier
```

```python
# Deep Learning
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout
from tensorflow.keras.layers import (Dense, Dropout, BatchNormalization, Input, Concatenate,Conv1D, MaxPooling1D, GlobalAveragePooling1D, Flatten,
Reshape, Add, Activation, LayerNormalization)
from tensorflow.keras.optimizers import Adam, AdamW
from tensorflow.keras.callbacks import EarlyStopping, ReduceLROnPlateau, ModelCheckpoint
from tensorflow.keras.regularizers import l1, l2, l1_l2
from tensorflow.keras.utils import to_categorical

# Set style for better visualizations
sns.set_style('whitegrid')
plt.rcParams['figure.dpi'] = 100

# Set seeds for reproducibility
np.random.seed(42)
tf.random.set_seed(42)

from sklearn.base import BaseEstimator, ClassifierMixin
from sklearn.ensemble import StackingClassifier
from sklearn.linear_model import LogisticRegression

import gc
from keras import backend as K
```

3 Data Sources & Linking

Dataset Name: CDC Diabetes Health Indicators (Kaggle + UCI)

Source:

- Kaggle: https://www.kaggle.com/datasets/alexteboul/diabetes-health-indicators-dataset
- UCI Repository: https://archive.ics.uci.edu/dataset/891/cdc+diabetes+health+indicators

```python
[2]: # Load dataset
df = pd.read_csv("diabetes_health_indicators.csv")
df.head()
```

[2]:

	Diabetes_binary	HighBP	HighChol	CholCheck	BMI	Smoker	Stroke	HeartDiseaseorAttack	PhysActivity	Fruits	...	AnyHealthcare	NoDocbcCost	GenHlth	MentH
0	0	1	1	1	40	1	0	0	0	0	...	1	0	5	
1	0	0	0	0	25	1	0	0	1	0	...	0	1	3	
2	0	1	1	1	28	0	0	0	0	1	...	1	1	5	
3	0	1	0	1	27	0	0	0	1	1	...	1	0	2	
4	0	1	1	1	24	0	0	0	1	1	...	1	0	2	

5 rows × 22 columns

3.1 Data Cleaning

- Duplicate records removed
- Outlier treatment using percentile capping
- Handling class imbalance using SMOTE

3.2 Feature Engineering

Interaction features: Age_BMI_Interaction

Polynomial features: BMI_squared, Age_squared

Risk proxies: Glucose_Risk_Proxy, HbA1c_Estimated

Composite scores: LifestyleScore, HealthRiskScore, CGM_Variability_Score

Feature Engineering

[11]:
```
# Glucose Risk Proxy based on BMI, BP, Cholesterol, Age
df["Glucose_Risk_Proxy"] = (
    0.4 * (df["BMI"] / df["BMI"].max()) +
    0.3 * df["HighBP"] +
    0.2 * df["HighChol"] +
    0.1 * (df["Age"] / df["Age"].max())
)
```

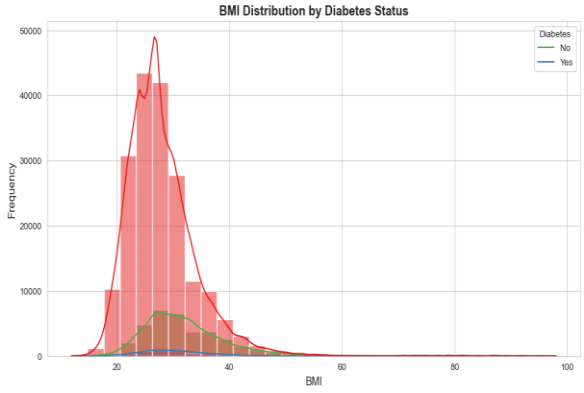
[12]:
```
# Synthetic HbA1c Estimate based on risk factors
df["HbA1c_Estimated"] = (
    5 +                                  # baseline healthy value
    (df["BMI"] / 50) * 1.2 +             # effect of obesity
    df["HighBP"] * 0.5 +                 # effect of hypertension
    df["HighChol"] * 0.4                 # effect of cholesterol
)
```

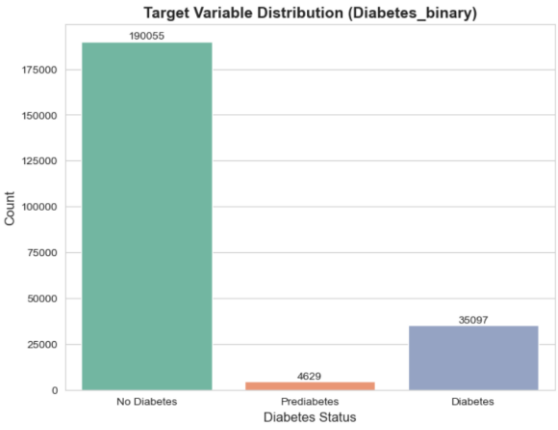
[13]:
```
# CGM Variability Score (simulated based on risk + comorbidities)
df["CGM_Variability_Score"] = (
    df["Glucose_Risk_Proxy"] *
    (1 + df["HeartDiseaseorAttack"] * 0.2 + df["Stroke"] * 0.3)
)
```

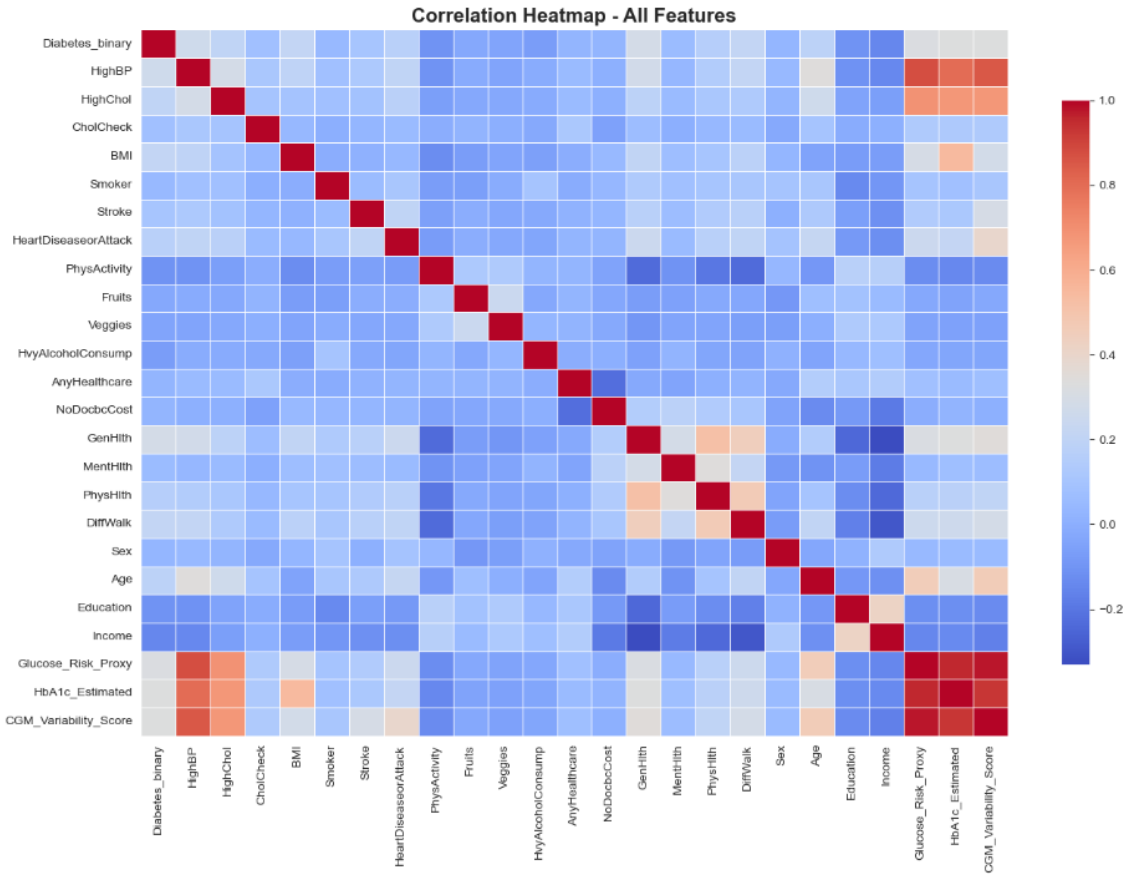
```
Creating advanced features...
Feature engineering complete!
Original features: 25
Total features after engineering: 40
New features created: 15

New features created:
 1. HealthRiskScore
 2. LifestyleScore
 3. BMI_Category
 4. MentalPhysicalHealth
 5. TotalHealthDays
 6. HealthDaysRatio
 7. Age_BMI_Interaction
 8. HealthcareAccess
 9. CardiovascularRisk
10. BehavioralRisk
11. SocioeconomicStatus
12. BMI_squared
13. Age_squared
14. DiffWalk_Age
15. GenHlth_Inverse
```

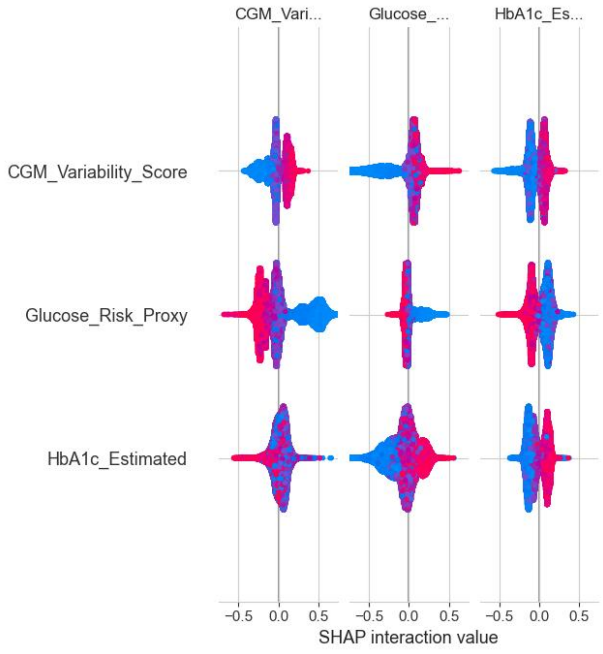
3.3 Data Visualization

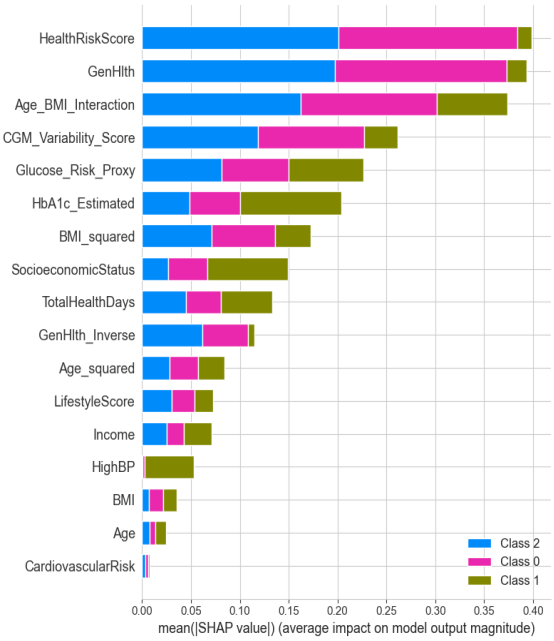
The following visualizations were implemented such as Class distribution plots, Correlation heatmaps Feature distributions, SHAP summary plots, SHAP interaction plots, Confusion matrices, Line graphs for Accuracy Macro F1 Weighted F1 Per-class recall trend graphs These visuals support model interpretability and evaluation transparency.






Correlation Heatmap - All Features
Diabetes_binary
HighBP
HighChol
CholCheck
BMI
Smoker
Stroke
HeartDiseaseorAttack
PhysActivity
Fruits
Veggies
HvyAlcoholConsump
AnyHealthcare
NoDocbcCost
GenHlth
MentHlth
PhysHlth
DiffWalk
Sex
Age
Education
Income
Glucose_Risk_Proxy
HbA1c_Estimated
CGM_Variability_Score
1.0
0.8
0.6
0.4
0.2
0.0
−0.2


CGM_Vari...
Glucose_...
HbA1c_Es...
CGM_Variability_Score
Glucose_Risk_Proxy
HbA1c_Estimated
−0.5
0.0
0.5
SHAP interaction value


HealthRiskScore
GenHlth
Age_BMI_Interaction
CGM_Variability_Score
Glucose_Risk_Proxy
HbA1c_Estimated
BMI_squared
SocioeconomicStatus
TotalHealthDays
GenHlth_Inverse
Age_squared
LifestyleScore
Income
HighBP
BMI
Age
CardiovascularRisk
Class 2
Class 0
Class 1
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0.35
0.40
mean(|SHAP value|) (average impact on model output magnitude)

3.4 Data Preprocessing

3.4.1 Standard Scaling

```
[39]: scaler = StandardScaler()
      X_train = scaler.fit_transform(X_train)
      X_test = scaler.transform(X_test)
      X_train = X_train[:30000]
      y_train = y_train[:30000]
```

3.4.2 Data Partitioning

```
[36]: # Split data with stratification
      X_train, X_test, y_train, y_test = train_test_split(
          X, y,
          test_size=0.2,
          random_state=42,
          stratify=y
      )

      print("Data Split:")
      print(f"Training set: {X_train.shape}")
      print(f"Test set: {X_test.shape}")
      print(f"\nTraining set class distribution:")
      print(y_train.value_counts())

      Data Split:
      Training set: (183824, 39)
      Test set: (45957, 39)

      Training set class distribution:
      Diabetes_binary
      0     152043
      2      28078
      1       3703
      Name: count, dtype: int64
```

4 Model Implementation Overview

Models Implemented

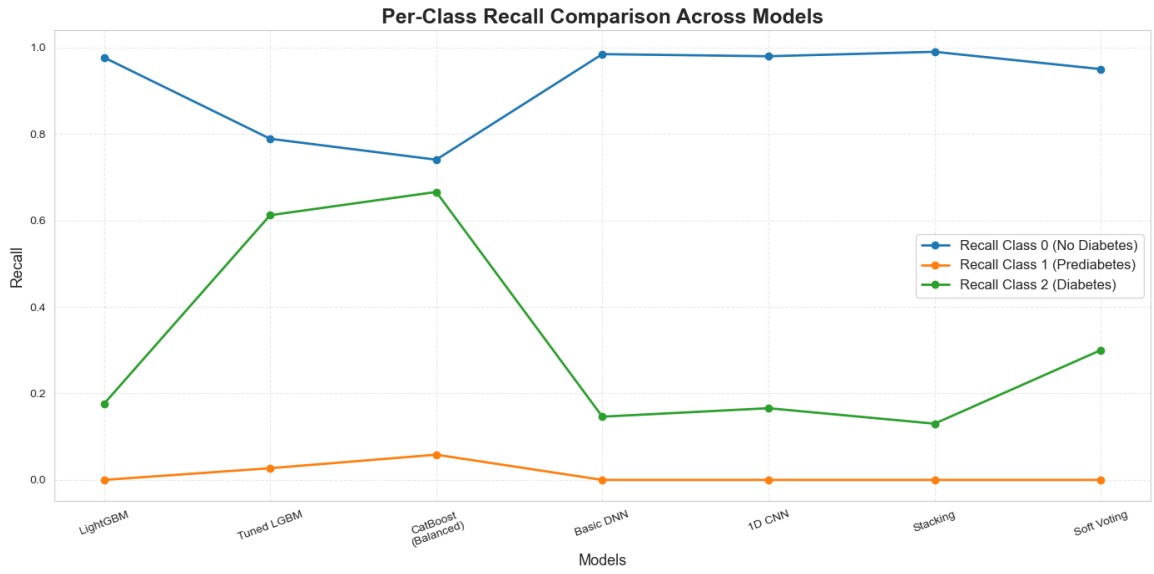
This study implements multi-model machine learning and deep learning framework for multi-class diabetes prediction (No Diabetes, Prediabetes, Diabetes). Our system Integrates Boosting models and neural networks to enable performance benchmarking, comparative evaluation, and ensemble-based validation. All these models were trained on the CDC Diabetes Health Indicators dataset using a stratified train–test split.

Category	Model Name	Purpose in System	Key Characteristics
Machine Learning	LightGBM	Primary baseline ML classifier	Fast gradient boosting, tree-based learning
Machine Learning	Tuned LightGBM	Optimized performance for imbalance	Regularized, class-weighted boosting
Machine Learning	CatBoost (Balanced)	Robust imbalance handling	Automatic class weight adjustment
Deep Learning	Basic DNN	Non-linear feature learning	Fully connected neural network
Deep Learning	1D CNN	Local pattern extraction	Convolutional feature learning
Ensemble (Experimental)	Stacking Ensemble	Combined ML prediction	LightGBM + CatBoost + Logistic Regression

Category	Model Name	Purpose in System	Key Characteristics
Ensemble (Experimental)	Soft Voting Ensemble	Combined ML + DL prediction	LightGBM + CatBoost + DNN + CNN

Table 1: Models Used in the System

Evaluation Metrics



References

Microsoft Corporation. (2024). *LightGBM documentation*. https://lightgbm.readthedocs.io/

Dorogush, A. V., Ershov, V., & Gulin, A. (2018). *CatBoost: Gradient boosting with categorical features support*.
https://catboost.ai/

Lundberg, S. M., & Lee, S.-I. (2017). *A unified approach to interpreting model predictions*. Advances in Neural Information Processing Systems (NeurIPS). https://shap.readthedocs.io/

Pedregosa, F., Varoquaux, G., Gramfort, A., et al. (2011). *Scikit-learn: Machine learning in Python*. Journal of Machine Learning Research, 12, 2825–2830. https://scikit-learn.org/stable/

TensorFlow Developers. (2024). *TensorFlow: An end-to-end open-source machine learning platform*.
https://www.tensorflow.org/

Base Paper Reference

Arman, B., Jazayeri, K., Çelebi, E., Alpan, K. and Dimililer, K. (2025) 'Ensemble learning for diabetes classification using voting classifier on CDC health indicators dataset', *Cyprus Journal of Medical Sciences*, 10(Suppl 1), pp. 111–115. doi: 10.4274/cjms.2025.2024-126.