National College of Ireland

Deep Learning and Machine Learning for Enhanced Anomaly Detection in EVSE systems

MSc Research Project
MSc in Cloud Computing

Shivangi Pandey
Student ID: 23188006

School of Computing
National College of Ireland

Supervisor: Jorge Mario Cortes Mendoza

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Shivangi Pandey
Student ID:	23188006
Programme:	MSc in Cloud Computing
Year:	2025
Module:	MSc Research Project
Supervisor:	Jorge Mario Cortes Mendoza
Submission Due Date:	15/09/2025
Project Title:	Deep Learning and Machine Learning for Enhanced Anomaly Detection in EVSE systems
Word Count:	1207
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:	Shivangi Pandey
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):	

Deep Learning and Machine Learning for Enhanced Anomaly Detection in EVSE systems

Shivangi Pandey
23188006

1 System Configuration, Tools and Libraries

The project was implemented in Python 3.8 using JupyterLab and Google Colab. Libraries include:

- **TensorFlow/Keras**: for building and training DNNs
- **Scikit-learn**: for ML models and evaluation metrics
- **Imbalanced-learn**: for SMOTE
- **Pandas/NumPy**: for data handling
- **Matplotlib/Seaborn**: for visualization

GridSearchCV was used for hyperparameter optimization, and EarlyStopping was applied to avoid overfitting in DNN training.

The project was implemented in Python 3.8 using Google Colab as the development environment. It utilized TensorFlow/Keras (2.18.0), Scikit-learn (1.6.1), Pandas (2.2.2), NumPy (2.0.2), Matplotlib (3.10.0), and Seaborn (0.13.2) for model development, data processing, and visualization.

In other words, The project was implemented using Python 3.8 within Google Colab, leveraging JupyterLab for experimentation. The primary libraries included TensorFlow/Keras TensorFlow (n.d.)Keras (n.d.) for building and training deep learning models, Scikit-learn (Scikit-learn, n.d.) for machine learning models and evaluation metrics, Imbalanced-learn Imbalanced-learn (n.d.)for SMOTE oversampling, Pandas and NumPy NumPy (n.d.); Pandas (n.d.) for data handling, and Matplotlib/Seaborn Matplotlib (n.d.); Seaborn (n.d.) for data visualisation. GridSearchCV Scikit-learn (n.d.) was utilised for hyperparameter tuning, and EarlyStopping callbacks (Keras, n.d.) were employed in DNN training to avoid overfitting.

Incorporated dataset for our project https://www.unb.ca/cic/datasets/evse-dataset-2024.html

2 System Setup and Implementation Steps

The implementation followed a structured pipeline aligned with the Knowledge Discovery in Databases (KDD) methodology Fayyad et al. (1996). The process began with setting up a Google Account and Google Drive, installing Google Colab, and connecting Colab to Drive for data access.

2.0.1 Step 1: Create a Google account

Visit https://accounts.google.com and sign up for a Google account.

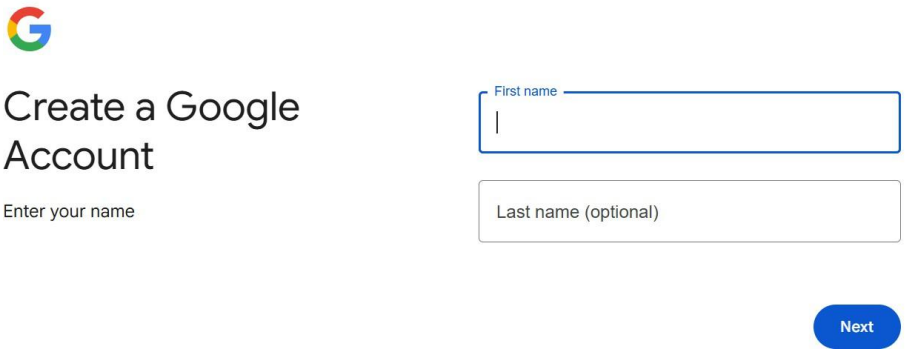

Figure 1: Google Account creation screen

2.0.2 Step 2: Open Google Drive

Go to https://drive.google.com and log in and create a new folder to organize project files.

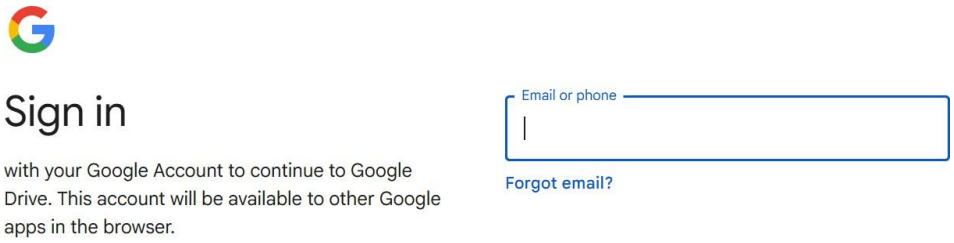

Figure 2: Google Account Sign In

2.0.3 Step 3: Install Google Collaboratory

- Open Google Drive and click New > More > Connect more apps.
- Search for **Colaboratory** in the Google Workspace Marketplace.
- Click **Install** to add it to your Drive (Colab will auto-install on first use as well).

2.0.4 Step: 4 Start new google colab notebook file

In Google Drive, click New > More > Colaboratory. This opens a new .ipynb (Colab Notebook).

2.0.5 Step: 5 Write code for implementation

Inside the Colab notebook, start entering Python code. Use + Code to add code cells and + Text for notes or headings.

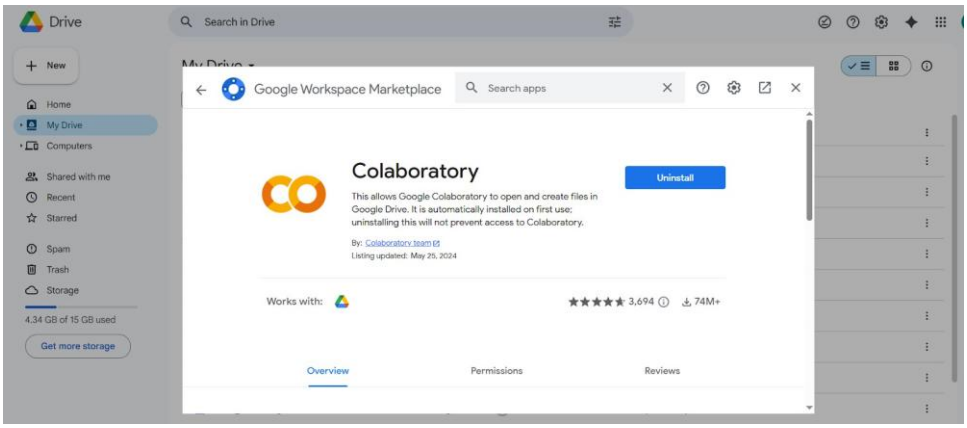

Figure 3: Installation of Colab

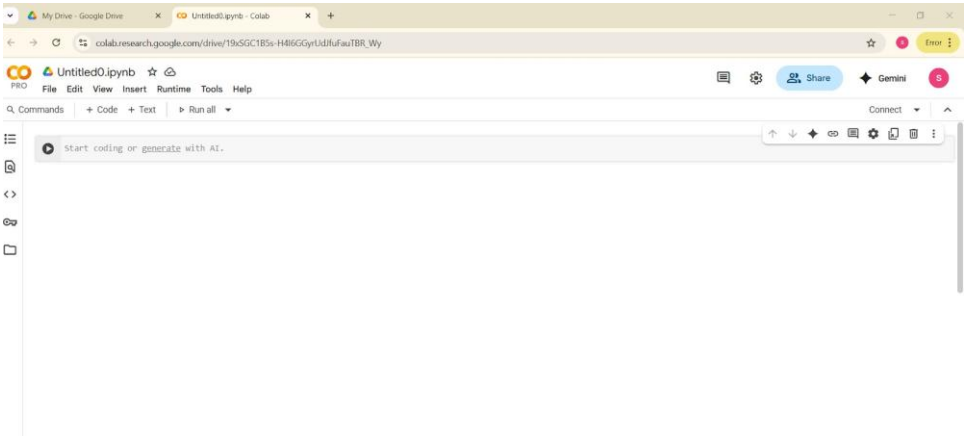

Figure 4: Notebook of Colab

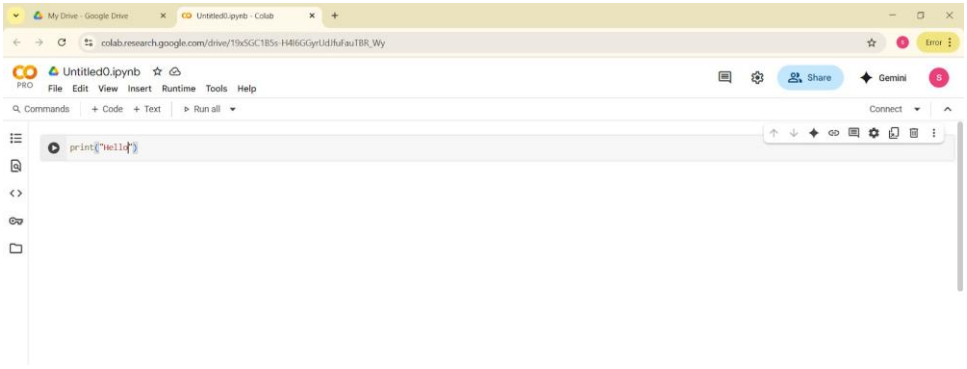

Figure 5: Initial Setup of Colab

2.0.6 Step: 6 Mount a Drive

This step allows to access files stored in my Google Drive:

- After executing the above cell, authorize access.
- Drive will be mounted at /content/drive.

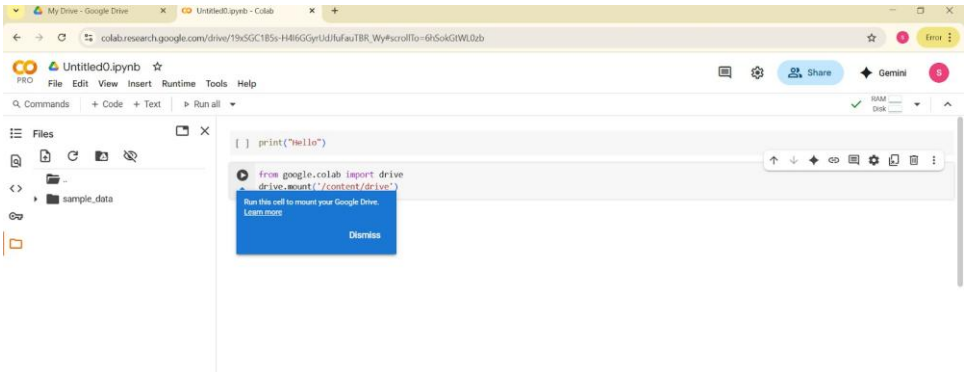

Figure 6: Mounting a drive

2.0.7 Step: 7 Set the path and run the cell to get output

Set the file path to dataset (CSV or other format) stored in Google Drive. Use libraries like Pandas, NumPy, Sklearn, Imbalanced-learn, and TensorFlow/Keras.

Figure 7: Initial Stage of Implementation

The CICEVSE2024 dataset was selected due to its realistic, labelled EVSE network traffic data. Data preprocessing involved cleaning missing values, removing duplicates, encoding categorical features, and normalising numerical attributes. Severe class imbalance (100,935 attack samples vs. 14,363 benign samples) was addressed using the Synthetic Minority Oversampling Technique (SMOTE), resulting in a balanced dataset of 128,482 samples. Exploratory Data Analysis (EDA) included Kernel Density Estimation (KDE) plots to visualise feature distributions. Following preprocessing, the data was split into training (80%) and testing (20%) sets. Four models were trained and tuned individually: Random Forest, Support Vector Machine, Gradient Boosting, and a Deep Neural Network. Hyperparameter tuning was conducted using GridSearchCV with 5-fold cross-validation.

3 Code Execution Workflow

The code execution process in Google Colab is designed to seamlessly integrate local code development with cloud-based resources and Google Drive storage. The workflow begins by mounting Google Drive within the Colab environment to access datasets and store outputs persistently. This is achieved using the ‘google.colab.drive‘ module. Once mounted, the dataset path is set by specifying the file location within the Drive directory structure. Data is then loaded into memory using libraries such as Pandas. Code cells are executed sequentially, allowing the integration of preprocessing, model training, and

evaluation in distinct, manageable blocks. Markdown and text cells are used to document each step, ensuring clarity and reproducibility. During execution, outputs such as evaluation metrics, confusion matrices, and plots are generated inline within the Colab notebook. Large model artifacts, trained weights, and result CSV files are saved back to Google Drive for future use or sharing. This approach ensures that both intermediate results and final outputs are securely stored, avoids loss of progress due to Colab session resets, and supports collaborative work by enabling shared Drive access.

References

Fayyad, U., Piatetsky-Shapiro, G. and Smyth, P. (1996). From data mining to knowledge discovery in databases, *AI Magazine* **17**(3): 37–37.

Imbalanced-learn (n.d.). Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning, https://imbalanced-learn.org/stable/. Version 0.12.3, Accessed 10 Aug. 2025.

Keras (n.d.). Keras api reference, https://keras.io/. Version 3.3.3, Accessed 10 Aug. 2025.

Matplotlib (n.d.). Matplotlib: Visualization with python, https://matplotlib.org/stable/. Version 3.9.2, Accessed 10 Aug. 2025.

NumPy (n.d.). Numpy, https://numpy.org/. Version 2.3.2, Accessed 10 Aug. 2025.

Pandas (n.d.). Pandas: Python data analysis library, https://pandas.pydata.org/. Version 2.2.2, Accessed 10 Aug. 2025.

Scikit-learn (n.d.). Machine learning in python, https://scikit-learn.org/stable/. Version 1.7.1, Accessed 10 Aug. 2025.

Seaborn (n.d.). Seaborn: Statistical data visualization, https://seaborn.pydata.org/. Version 0.13.2, Accessed 10 Aug. 2025.

TensorFlow (n.d.). Tensorflow, https://www.tensorflow.org/. Version 2.17.0, Accessed 10 Aug. 2025.