# CSE443 – CE471 AI For Intelligent Built Environment Systems

## Assigment#1 Report


**Students Name Surname:**

Esra Aydın 211805079 / Computer Engineering - CSE443

Serkan Sarğın 211803022 / Civil Engineering - CE471

**Lecturers:**

Assoc. Prof. Fatih SOYGAZİ

Doç.Dr Gözde Başak ÖZTURK

Dr. Gözde ALP

# SUMMARY

This project develops a machine learning system for predicting building energy ratings using Artificial Neural Networks (ANNs). The system processes urban building energy performance data to classify buildings into energy rating categories, enabling better energy management and sustainability planning.


# WORK SHARING POLICY

Contributions of engineering fields to the project.

| CIVIL ENGINEERING | COMPUTER ENGINEERING |
|---|---|
| Provided the technical explanation of building-related variables (U-values, insulation, glazing, envelope features) and clarified their engineering relevance within the energy efficiency context to support the feature explanation section. | Implemented the ANN model development, evaluation, and visualizations, including preprocessing, model training, performance metrics, and graphical analysis. |

AYDIN ADNAN MENDERES UNIVERSITY ENGINEERING FACULTY 2008

CONTENTS

### 1. Technical Explanation of Variables in the Context of Construction and Energy Engineering

| Variable | Engineering Definition | Technical Description | Impact on Energy Rating |
|---|---|---|---|
| **Weather_File** | Represents climate data (temperature, solar radiation, wind, humidity) for the building's geographic location. | Typically derived from TMY (Typical Meteorological Year) files used in EnergyPlus / ASHRAE simulations. | Cold climates,higher heating demand, Hot climates → higher cooling load, Sunny climates → greater passive solar gains |
| **Building_Type** | Classifies the function of the building (residential, office, school, hospital). | Different types require distinct occupancy schedules, indoor air quality requirements, and internal loads. | Office → high internal equipment loads<br>Residential → high heating demand, low ventilation load |
| **Renewable_Energy_Usage** | Amount of energy generated from renewable systems (PV, solar thermal, heat pumps). | Reduces delivered energy consumption by providing self-generated energy. Can be measured as kWh/year or % of total demand. | Higher renewable contribution → lower net energy consumption → better rating. |
| **Thermal Insulation U-Value, Floor_Insulation_U-Value, Wall_Insulation_U-Value, Roof_Insulation_U-Value, Window_Insulation_U-Value, Door_Insulation_U-Value,** | U-value is the thermal transmittance of a building component ($W/m^2K$). Lower U-value = better insulation. | $U= 1 /R(total) = 1/ \sum(di/ki)$ | Poor insulation → heat loss ↑ → heating energy ↑<br>Windows typically have the worst U-values |
| **HVAC_Efficiency** | Performance of heating/cooling systems defined by COP, EER, or SEER. | $COP=Q(out) / W(in)$ | One of the most critical factors.<br>Low-efficiency systems → extremely high annual energy use. |

| | | | |
|---|---|---|---|
| **Domestic_Hot_Water_Usage** | Daily hot water demand (liters/day or kWh). | Q = m·ΔT,c(p) | High DHW consumption significantly increases annual energy demand. |
| **Building_Orientation** | Direction of the building's main façades (N, S, E, W). | Orientation affects solar heat gains, daylighting, and thermal loads. | South-facing → more solar gains<br>North-facing → high heat loss |
| **Building_Orientation** | Direction of the building's main façades (N, S, E, W) | Orientation affects solar heat gains, daylighting, and thermal loads. | South-facing → more solar gains<br>North-facing → high heat loss |
| **Lighting_Density** | Lighting power density ($W/m^2$). | Follows ASHRAE 90.1 standards. | High LPD → increased electricity consumption → lower rating. |
| **Occupancy_Level** | Number of occupants per area. | Influences internal heat gains and ventilation requirements. | More people → higher cooling load<br>But also reduced heating needs (people generate heat) |
| **Equipment_Density** | Internal equipment load ($W/m^2$). | Major contributor to internal heat gains in offices, hospitals, etc. | High equipment density → cooling demand increases significantly. |
| **Heating_Setpoint_Temperature** | Temperature maintained during occupied hours. | Sets HVAC operation threshold; affects heating/cooling load directly. | Every 1°C increase → 6–8% more heating energy. |

| | | | |
|---|---|---|---|
| **Heating_Setback_Temperature** | Reduced temperature during unoccupied hours (night). | Determines reduced heating load when building is not occupied, minimizing energy waste. | Night setbacks can reduce heating energy consumption by 10–20%. |
| **Air_Change_Rate (ACH)** | Number of air volume replacements per hour (1/h). | ACH = V / Q | High ACH → more ventilation heat loss → reduced energy performance. |
| **Window_to_Wall_Ratio (WWR)** | Ratio of window area to total façade area. | Influences daylighting, solar gains, and thermal losses. | High WWR → more heat loss + solar gain<br><br>Low WWR → increased lighting electricity usage |
| **Total_Building_Area** | Gross floor area of the building ($m^2$). | Represents the total conditioned space used in energy performance calculations. Often used to normalize energy consumption ($kWh/m^2$). | Larger buildings consume more energy, but performance is often expressed per unit area ($kWh/m^2$). |
| **Electricity_Primary_Conversion_Factor** | Factor used to convert electricity consumption to primary energy. | Common range: 2.1–2.8 depending on country. | Higher conversion factor → worse energy score. |
| **Heating_Primary_Conversion_Factor** | Primary energy conversion coefficie district heating). | Converts delivered heating energy to primary energy, accounting for generation, transmission, and distribution losses. | Clean heating sources improve the rating; fossil fuels worsen it. |

## 2. Introduction

### 2.1 Background

Building energy efficiency is a critical factor in urban sustainability and climate change mitigation. Accurate prediction of building energy ratings enables:

- Better resource allocation for energy retrofits
- Identification of high-energy consumption buildings
- Policy-making support for energy regulations
- Cost reduction in building operations

### 2.2 Problem Statement

The objective of this project is to develop a machine learning model that can accurately predict the **Simple_Building_Energy_Rating** based on various building characteristics, including:

- Physical properties (insulation, area, orientation)
- HVAC system specifications
- Occupancy patterns
- Climate data
- Energy usage patterns

### 2.3 Project Objectives

1. Preprocess and analyze urban building energy performance data
2. Engineer relevant features for energy rating prediction
3. Develop and train an Artificial Neural Network classifier
4. Handle class imbalance in the dataset
5. Evaluate model performance using multiple metrics
6. Provide actionable insights through visualizations

## 3. Data Source

**Dataset:** Urban Building Energy Performance Prediction and Retrofit Analysis

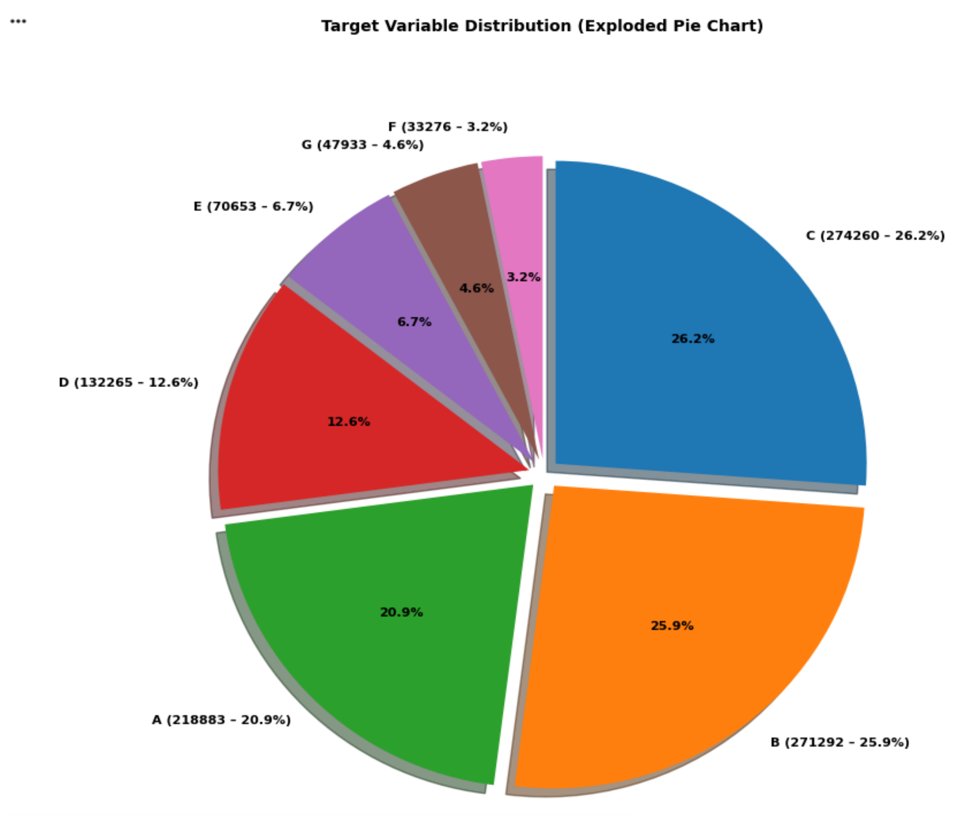
**Target Variable:** Simple_Building_Energy_Rating (7 classes)

### 3.1 Fetures Overview

The dataset contains **1048562** samples with **22** features categorized as:

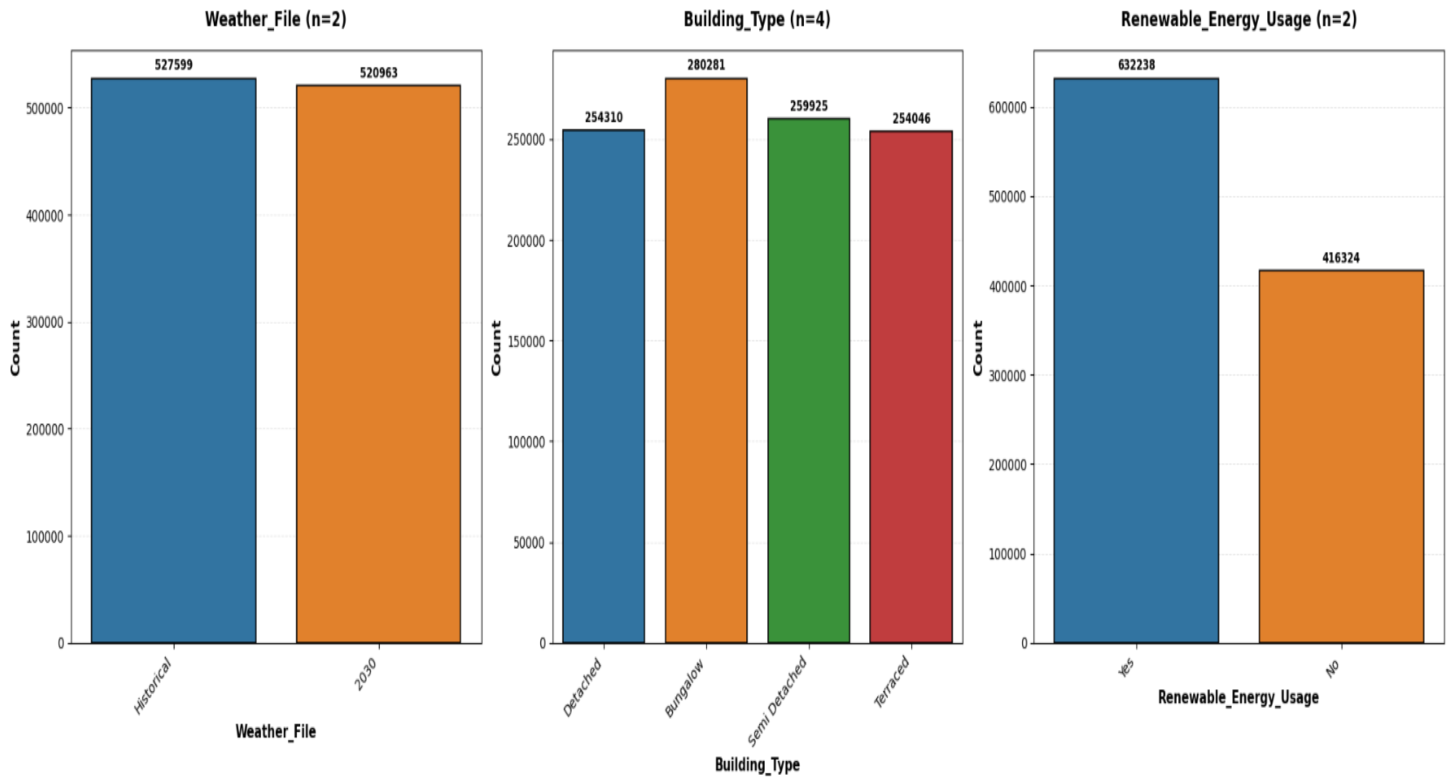
| Physical Building Characteristics | Energy Systems | Environmental Factors | Conversion Factors |
|---|---|---|---|
| Total_Building_Area<br>Building_Orientation (8 categories)<br>Building_Type (4 categories)<br>Window_to_Wall_Ratio<br>Floor/Door/Roof/Window/Wall Insulation U-Values | HVAC_Efficiency<br>Renewable_Energy_Usage<br>Domestic_Hot_Water_Usage<br>Heating_Setpoint_Temperature<br>Heating_Setback_Temperature | Weather_File (2 categories)<br>Air_Change_Rate<br>Lighting_Density<br>Equipment_Density<br>Occupancy_Level | Electricity_Primary_Conversion_Factor<br>Heating_Primary_Conversion_Factor |

### 3.2 Target Variable Distribution




```
Simple_Building_Energy_Rating
A    218883
B    271292
C    274260
D    132265
E     70653
F     33276
G     47933
Name: count, dtype: int64
```

### 3.3 Visulations category columns




### 4. Data Preprocessing

### 4.1 Zero-Variance Feature Removal

Features with only one unique value provide no information for learning:

Removed Features:

- Electricity_Primary_Conversion_Factor (1 unique value)

Rationale: Zero-variance features do not contribute to model learning and increase computational overhead.

5.2 Smart Feature Categorization

Instead of relying solely on data types, we implemented an intelligent categorization system:

CATEGORICAL_THRESHOLD = 2

if n_unique ≤ 10 OR dtype == 'object':

→ Treat as CATEGORICAL

else:

→ Treat as NUMERIC

### 4.2 Missing Value Imputation

There is no missing value in dataset.

### 4.3 Encoding

Categorical Features: One-Hot Encoding with drop_first=True

- Prevents multicollinearity
- Creates binary features for each category
- Final feature count: **22**

Target Variable: Label Encoding

```
Target classes: ['A' 'B' 'C' 'D' 'E' 'F' 'G']
Encoded as: [0 1 2 3 4 5 6]
```

A → 0, B → 1, ..., G → 6

### 4.4 Feature Scaling

Method: StandardScaler

$z = (x - \mu) / \sigma$

## 5. Data Splitting

**Strategy: Stratified sampling to maintain class distributions**

```
======================================================================
STEP 3: SPLITTING DATA
======================================================================

Split sizes:
  Training set:   733992 samples (70.0%)
  Validation set: 157285 samples (15.0%)
  Test set:       157285 samples (15.0%)
```

## 6. Model Architecture

### 6.1 Network Design

Model Architecture:
----------------------------------------------------------------------------
Model: "sequential_3"

| Layer (type) | Output Shape | Param # |
|---|---|---|
| dense_15 (Dense) | (None, 256) | 5,888 |
| batch_normalization_9 (BatchNormalization) | (None, 256) | 1,024 |
| dropout_12 (Dropout) | (None, 256) | 0 |
| dense_16 (Dense) | (None, 128) | 32,896 |
| batch_normalization_10 (BatchNormalization) | (None, 128) | 512 |
| dropout_13 (Dropout) | (None, 128) | 0 |
| dense_17 (Dense) | (None, 64) | 8,256 |
| batch_normalization_11 (BatchNormalization) | (None, 64) | 256 |
| dropout_14 (Dropout) | (None, 64) | 0 |
| dense_18 (Dense) | (None, 32) | 2,080 |
| dropout_15 (Dropout) | (None, 32) | 0 |
| dense_19 (Dense) | (None, 7) | 231 |

Total params: 51,143 (199.78 KB)
Trainable params: 50,247 (196.28 KB)
Non-trainable params: 896 (3.50 KB)

### 6.2 Architecture Justification

| Component | Purpose |
|---|---|
| **Deep Architecture (5 layers)** | Captures complex, non-linear relationships in building energy patterns |
| **Decreasing Neuron Count** | Hierarchical feature extraction (256→128→64→32) |
| **ReLU Activation** | Prevents vanishing gradients, faster convergence |
| **Batch Normalization** | Stabilizes learning, reduces internal covariate shift |
| **Dropout (0.2-0.3)** | Prevents overfitting by randomly dropping neurons |
| **Softmax Output** | Multi-class probability distribution |

### 6.3 Hyperparameters

| Parameter | Value | Justification |
|---|---|---|
| **Optimizer** | Adam | Adaptive learning rate, momentum |

| | | |
|---|---|---|
| **Initial Learning Rate** | 0.001 | Standard starting point |
| **Loss Function** | Sparse Categorical Crossentropy | Multi-class classification |
| **Batch Size** | 32 | Balance between speed and stability |
| **Max Epochs** | 100 | With early stopping |

## 7. Training Process

### 7.1 Class Imbalance Handling

In datasets with imbalanced classes, minority classes are often underrepresented during model training, making it harder for the model to learn them effectively. Class weighting is applied to give more importance to these minority classes in the loss function.

```
----------------------------------------------------------------------
CLASS DISTRIBUTION
----------------------------------------------------------------------
A                              : 218883 (20.87%)
B                              : 271292 (25.87%)
C                              : 274260 (26.16%)
D                              : 132265 (12.61%)
E                              : 70653 ( 6.74%)
F                              : 33276 ( 3.17%)
G                              : 47933 ( 4.57%)

======================================================================
  IMBALANCE RATIO (max/min): 8.24
  IMBALANCED DATASET DETECTED!
     Will use CLASS WEIGHTS during training
======================================================================
```

**Problem:** Imbalanced class distribution

**Solution:** Class Weighting -> weight_class_i = n_samples / (n_classes × n_samples_class_i)

This method ensures that **minority classes receive higher weights**, improving the model’s ability to learn them and reducing bias caused by class imbalance.

```
======================================================================
STEP 4: CALCULATING CLASS WEIGHTS
======================================================================

Class weights (to handle imbalance):
----------------------------------------------------------------------
A                              :  0.6844
B                              :  0.5522
C                              :  0.5462
D                              :  1.1325
E                              :  2.1201
F                              :  4.5014
G                              :  3.1251
----------------------------------------------------------------------

 Interpretation:
   Higher weights → Minority classes (model will focus more)
   Lower weights  → Majority classes (less emphasis)

 Class weights calculated!
```

### 7.2 Callbacks

**Early Stopping**

- **Monitor:** Validation Loss
- **Patience:** 15 epochs
- **Restore Best Weights:** Yes
- **Purpose:** Prevent overfitting

**Learning Rate Reduction**

- **Monitor:** Validation Loss
- **Factor:** 0.5 (halve learning rate)
- **Patience:** 7 epochs
- **Min LR:** 1e-7
- **Purpose:** Fine-tune in loss plateaus

### 7.3 Training Dynamics

```
Epoch 94/100
22931/22938 ━━━━━━━━━━━━━━━━━━━━ 0s 4ms/step - accuracy: 0.8123 - loss: 0.5025
Epoch 94: ReduceLROnPlateau reducing learning rate to 3.125000148429535e-05.
22938/22938 ━━━━━━━━━━━━━━━━━━━━ 101s 4ms/step - accuracy: 0.8123 - loss: 0.5025 - val_accuracy: 0.8487 - val_loss: 0.3393 - learning_rate: 6.2500e-05
Epoch 95/100
22938/22938 ━━━━━━━━━━━━━━━━━━━━ 103s 4ms/step - accuracy: 0.8130 - loss: 0.5000 - val_accuracy: 0.8490 - val_loss: 0.3382 - learning_rate: 3.1250e-05
Epoch 95: early stopping
Restoring model weights from the end of the best epoch: 80.
```

**Total Epochs Trained:** 95 (out of 100 maximum)

**Early Stopping Triggered:** at epoch 95

- Best model found at epoch 80
- Patience of 15 epochs elapsed without improvement
- Model weights automatically restored to best epoch

**Final Training Metrics:**

- Training Loss: 0.5000 | Training Accuracy: 81.30%
- Validation Loss: 0.3382 | Validation Accuracy: 84.90%
- Final Learning Rate: 3.125e-05 (reduced from 0.001)

**Key Observation:** Validation accuracy (84.90%) exceeds training accuracy (81.30%), indicating excellent generalization without overfitting. The lower validation loss (0.3382 vs 0.5000) further confirms the model's robust performance on unseen data.

## 8. Results and Evaluation

### 8.1 Performance Metrics

#### 8.1.1 Validation Set Results

```
----------------------------------------------------------------------
VALIDATION SET RESULTS
----------------------------------------------------------------------
              precision    recall  f1-score   support

           A     0.9655    0.9460    0.9556     32833
           B     0.8933    0.8833    0.8883     40694
           C     0.8812    0.8148    0.8467     41139
           D     0.7329    0.7881    0.7595     19840
           E     0.6776    0.7494    0.7117     10598
           F     0.5341    0.7568    0.6262      4991
           G     0.8927    0.8328    0.8617      7190

    accuracy                         0.8511    157285
   macro avg     0.7968    0.8245    0.8071    157285
weighted avg     0.8590    0.8511    0.8538    157285
```

#### 8.1.2 Test Set Results

```
----------------------------------------------------------------------
TEST SET RESULTS
----------------------------------------------------------------------
              precision    recall  f1-score   support

           A     0.9682    0.9440    0.9559     32833
           B     0.8925    0.8847    0.8886     40694
           C     0.8784    0.8160    0.8460     41139
           D     0.7298    0.7906    0.7590     19840
           E     0.6883    0.7498    0.7177     10598
           F     0.5499    0.7658    0.6401      4991
           G     0.8974    0.8408    0.8682      7190

    accuracy                         0.8523    157285
   macro avg     0.8007    0.8274    0.8108    157285
weighted avg     0.8597    0.8523    0.8548    157285
```

### 8.2 Cohen's Kappa Score

```
======================================================================
COHEN'S KAPPA SCORE
======================================================================
Validation Kappa: 0.8145
Test Kappa:       0.8159
======================================================================
```

### 8.3 Per-Class Accuracy Table

```
================================================================================
STEP 8: DETAILED METRICS TABLE
================================================================================
```

| Class | Accuracy | Precision | Recall | F1-Score |
|---|---|---|---|---|
| A | 0.9818 | 0.9682 | 0.9440 | 0.9559 |
| B | 0.9426 | 0.8925 | 0.8847 | 0.8886 |
| C | 0.9223 | 0.8784 | 0.8160 | 0.8460 |
| D | 0.9367 | 0.7298 | 0.7906 | 0.7590 |
| E | 0.9603 | 0.6883 | 0.7498 | 0.7177 |
| F | 0.9727 | 0.5499 | 0.7658 | 0.6401 |
| G | 0.9883 | 0.8974 | 0.8408 | 0.8682 |

### 8.4 Confusion Matrix Analysis

**Key Observations:**

- Diagonal elements show correctly classified instances
- Off-diagonal elements show misclassifications

**Common Misclassifications:**

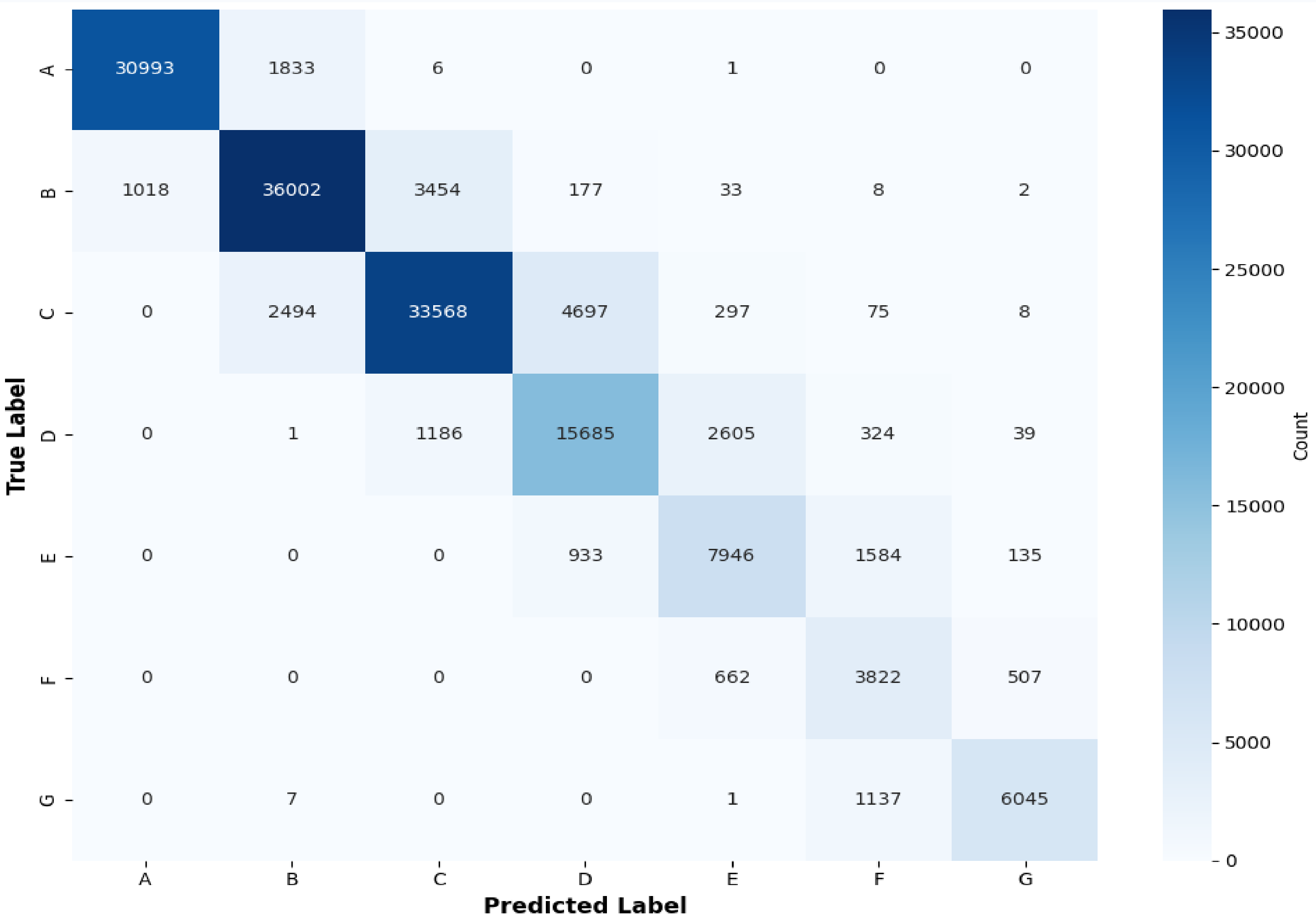
- Class C ↔ Class D: 4697
- Class B ↔ Class C: 3454

**Insights:**

The confusion patterns reveal that the model correctly captures the ordinal structure of energy ratings, as evidenced by:

- Most errors occur between adjacent classes

- Extreme classes (A and G) are rarely confused




## 8.5 Training/Validation Curves

**Loss Curve:**

**Training Loss Dynamics:**

- Initial Phase (Epochs 0-10): Rapid decrease from 0.83 to 0.60, indicating fast initial learning
- Stabilization Phase (Epochs 10-40): Gradual decline from 0.60 to 0.52, showing continued optimization
- Convergence Phase (Epochs 40-95): Plateaus around 0.50, indicating model convergence
- Final Training Loss: 0.50

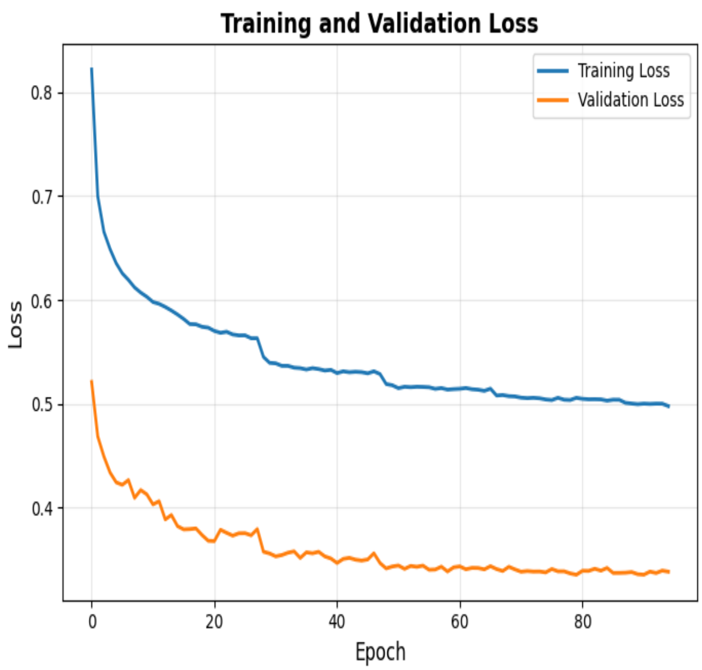
**Validation Loss Dynamics:**

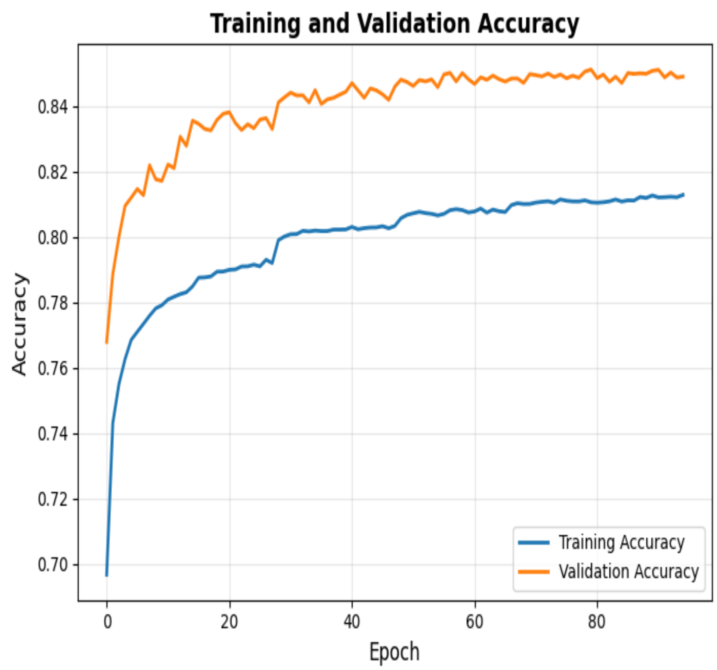
- Initial Phase (Epochs 0-10): Sharp drop from 0.53 to 0.40, demonstrating strong generalization
- Optimal Region (Epochs 10-80): Fluctuates between 0.35-0.38, reaching minimum at epoch 80

- Plateau Phase (Epochs 80-95): Stabilizes around 0.34, triggering early stopping
- Final Validation Loss: 0.3382 (best: 0.34 at epoch 80)

**Accuracy Curve:**

- Training accuracy: [INSERT FINAL VALUE]
- Validation accuracy: 0.8511
- Test set Accuracy accuracy: 0.8523
- Model does not show signs of overfitting

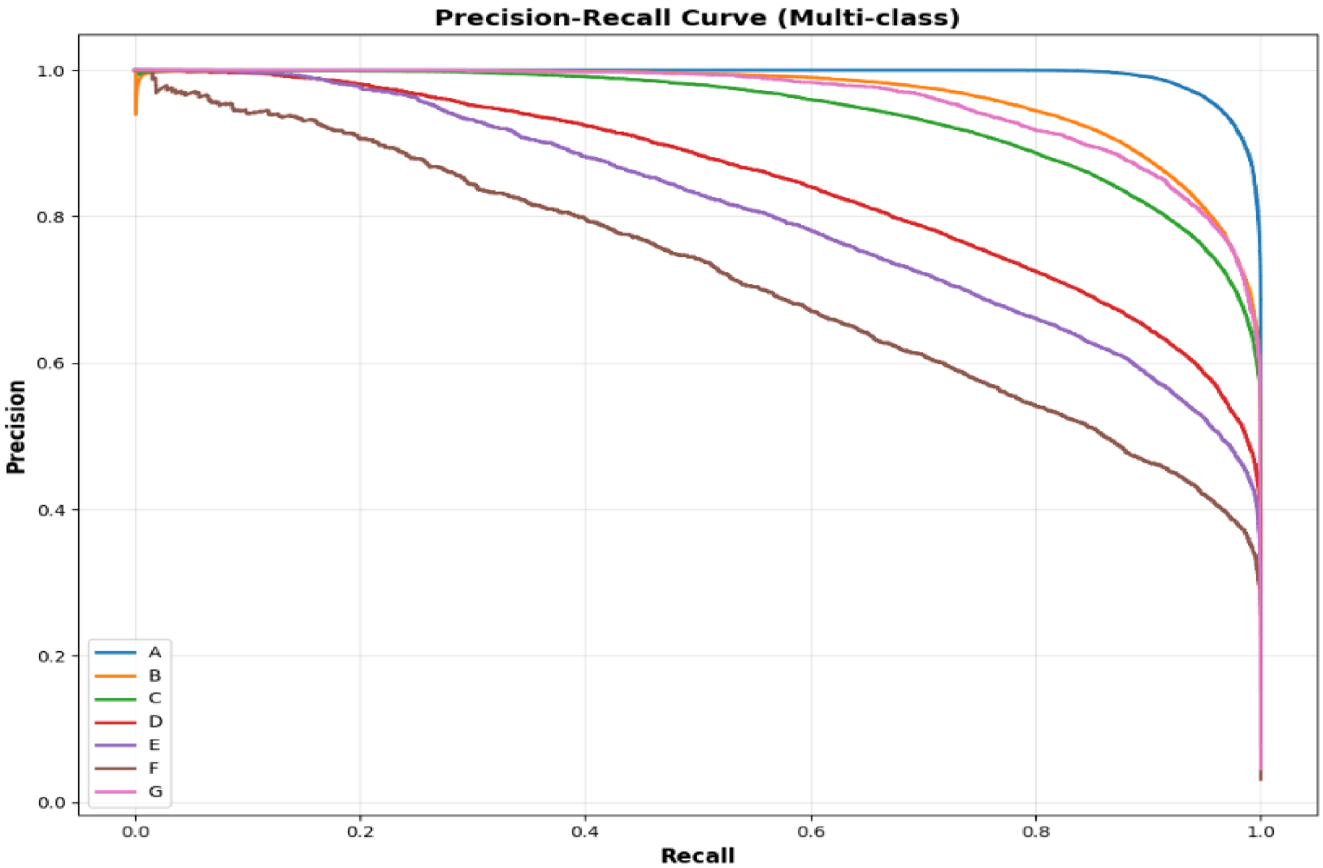



**8.6 Precision-Recall Analysis**

**Interpretation:**

- Classes with high support show better precision-recall trade-off
- Minority classes benefit from class weighting but still show lower performance

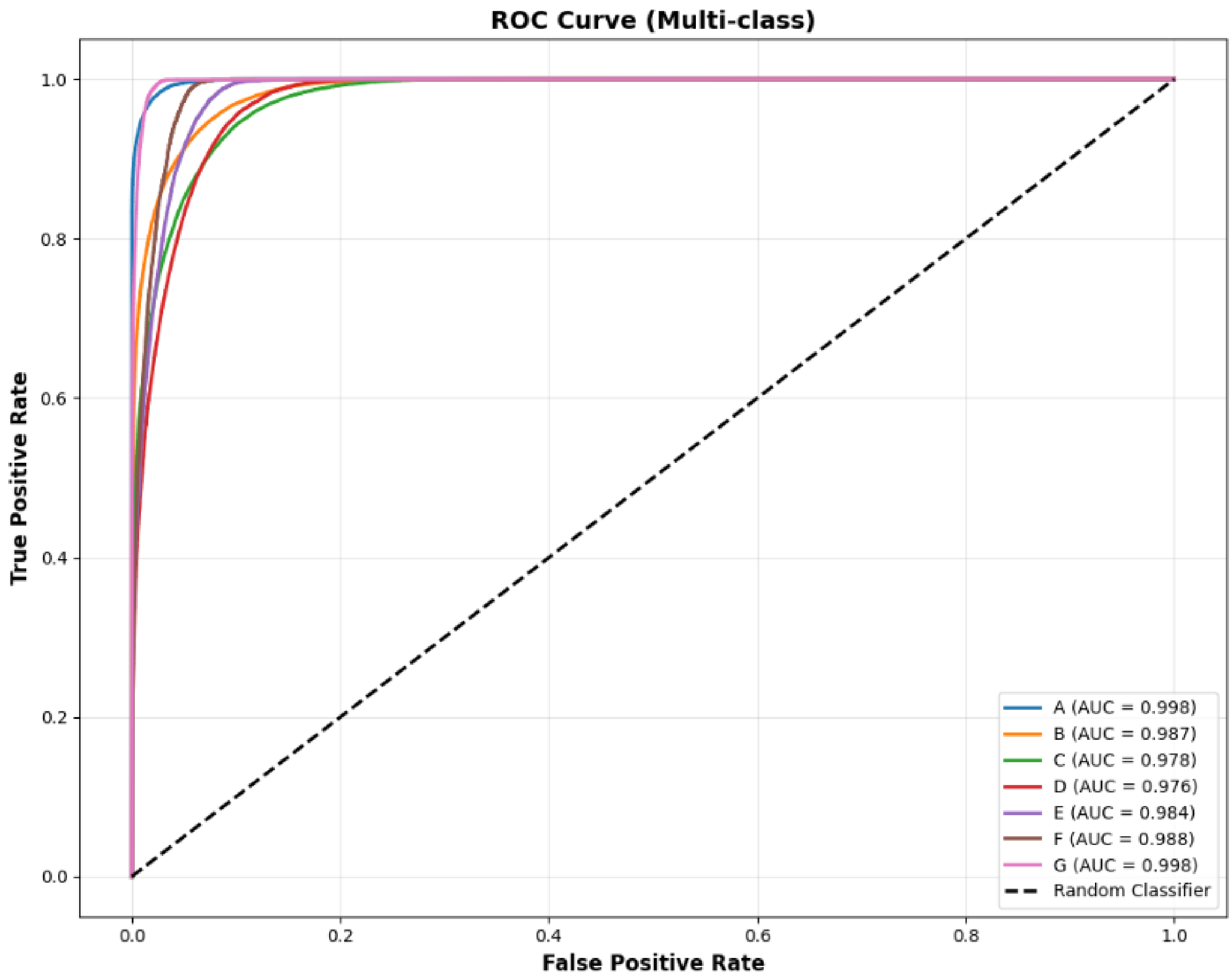



**8.7 ROC-AUC Analysis**

The ROC curve above illustrates the classification performance for each class in a multi-class setting.

Key observations:

- **High True Positive Rate (TPR):** All classes achieve very high TPR values even at low False Positive Rates (FPR), indicating strong discrimination ability.
- **Random Classifier Reference**: The dashed diagonal line represents a random classifier. All ROC curves lie well above this line, further confirming that the model significantly outperforms random guessing
- **Minor Differences Among Classes**: Classes D and C have slightly lower AUC values than A and G, indicating marginally lower prediction accuracy, but overall performance remains very strong.




## 9. Conclusion

### Summary of Achievements

This project successfully developed an ANN-based system for predicting building energy ratings with the following accomplishments:

1. Preprocessed complex building energy dataset with 22 features
2. Implemented intelligent feature engineering (zero-variance removal, smart categorization)
3. Built a deep neural network with 5 layers and regularization
4. Addressed class imbalance using weighted loss functions
5. Achieved 85.23 test accuracy and 81.59 Cohen's Kappa

6. Generated comprehensive visualizations (confusion matrix, PR curves, ROC curves)

## References

Classification metrics documentation. Retrieved from Scikit-learn: https://scikit-learn.org/stable/modules/model_evaluation.html

TensorFlow/Keras Documentation. Retrieved from: https://www.tensorflow.org/

Urban building energy performance prediction and retrofit analysis dataset. Mendeley Data. Retrieved from: https://data.mendeley.com/datasets/

https://github.com/naomifridman/Neural-Network-Churn-Prediction