Room Occupancy Detection

Javad Mokhtari Koushyar, Bao Hoang Duc Vo

2024

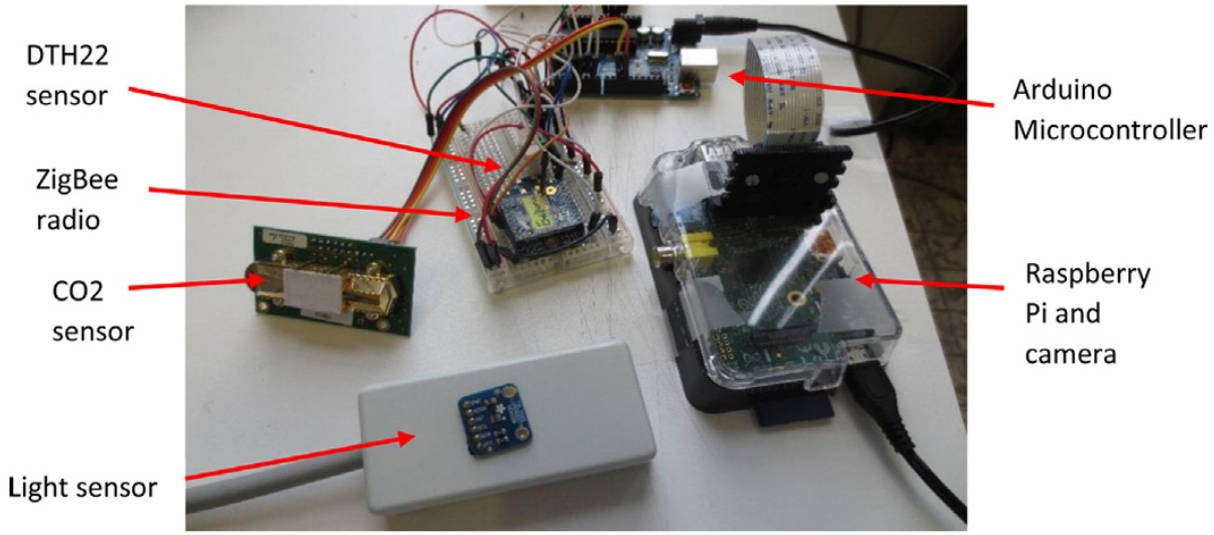
Outline

- Introduction
- Exploratory Data Analysis (EDA)
- Data Preparation
- Methodology
- Summary
- References

Introduction

- Motivation: Studies Show Occupancy Detection Can Cut Energy Use by 30-42% [1]
- Task: Detect occupancy of an office room accurately
- Applications:
  - Providing Energy Efficiency via Occupancy Detection
  - Security and Occupant Behavior Analysis
  - Privacy-Friendly Occupant Detection System
- Dataset: Three data sets were used in this work, one for training, and two for testing the models considering the **office door opened and closed** during occupancy
- Our Solution: Provided 5 different solutions using different AI algorithms

Exploratory Data Analysis (EDA)



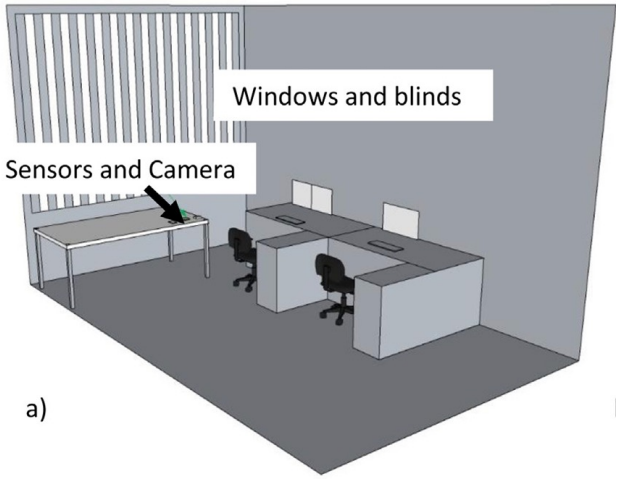
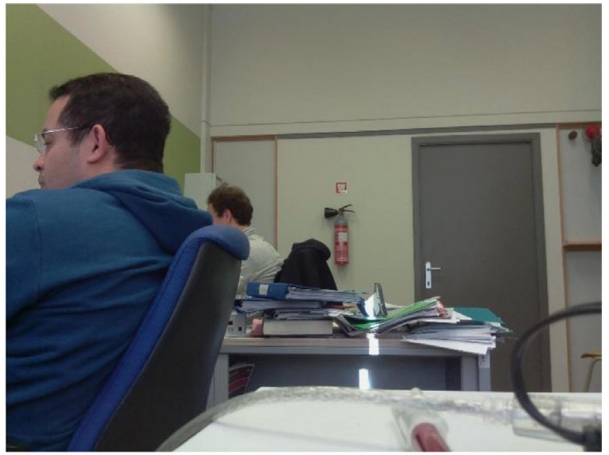


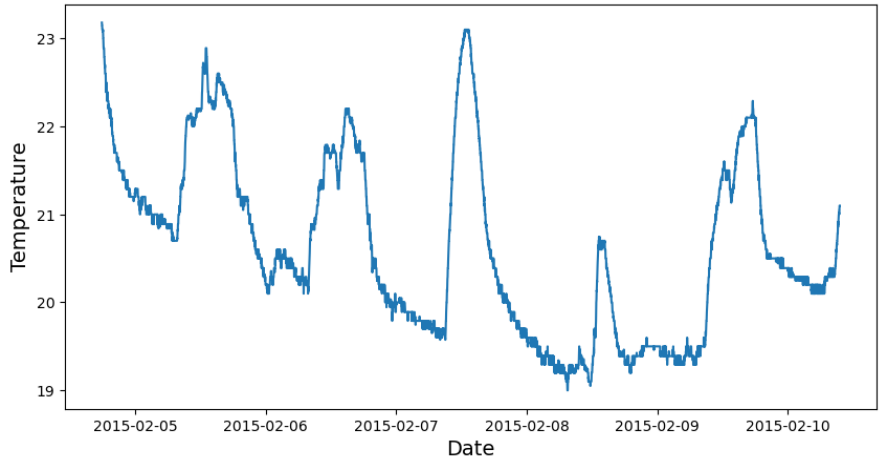
Table 1: Description of dataset attributes

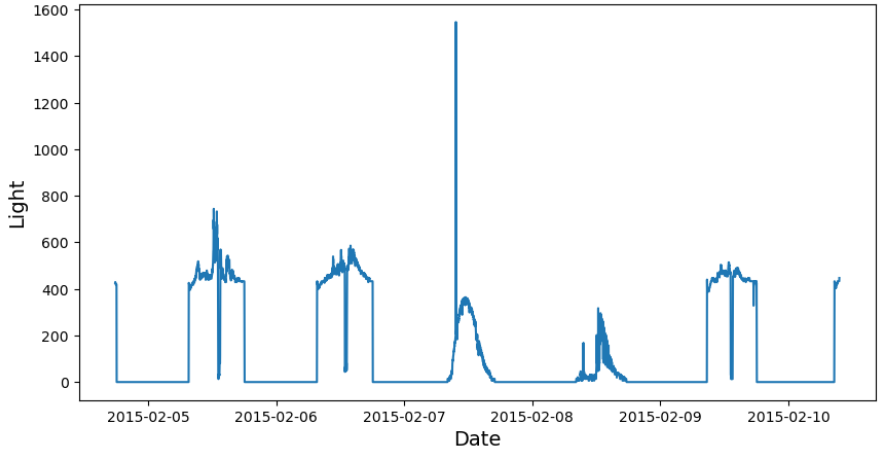
Attribute	Description
Date	Records the exact date and time of the data entry.
Temperature	Indicates the ambient temperature of the environment.
Relative Humidity	Shows the amount of moisture in the air.
Light	Represents the light level of the environment.
CO2	Can indicate human occupancy and affect the perceived air quality.
Humidity Ratio	This metric is useful for understanding the air's capacity to hold moisture.
Occupancy	This is the target variable for prediction.

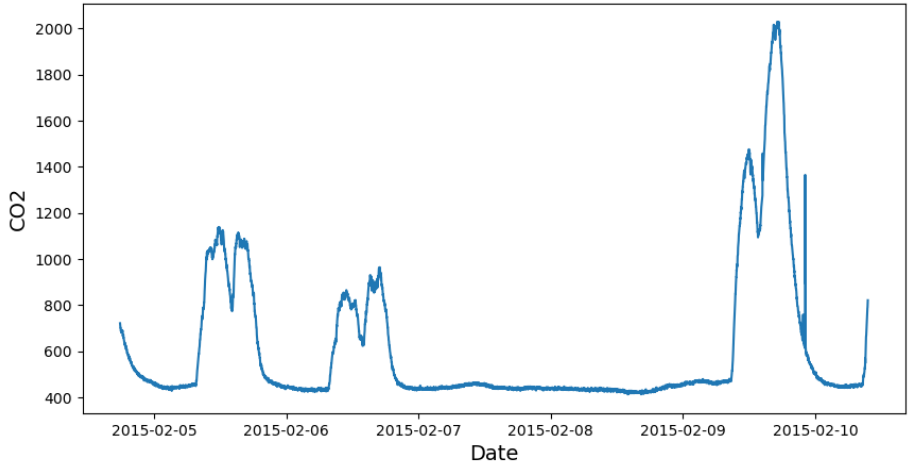


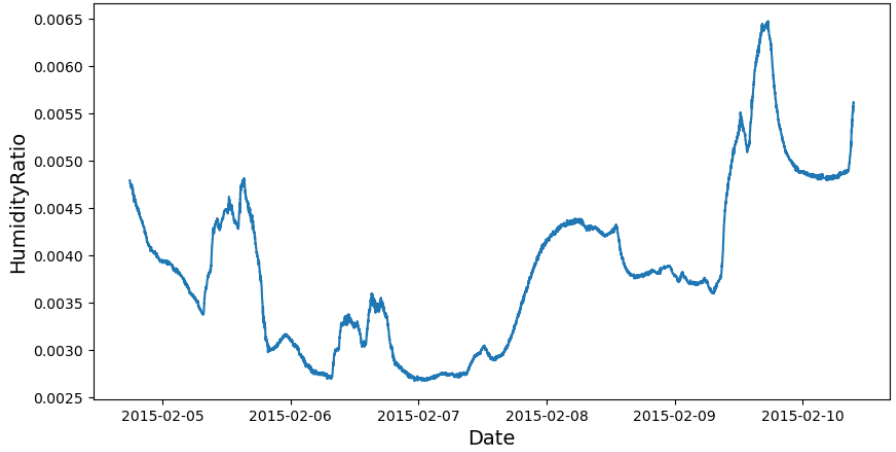
Images by Luis M. Candanedo et al. [1]

Exploratory Data Analysis (EDA)

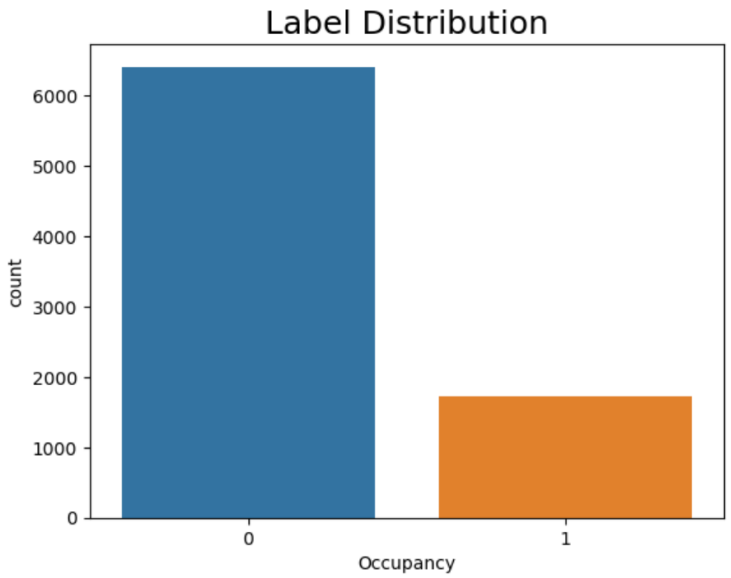


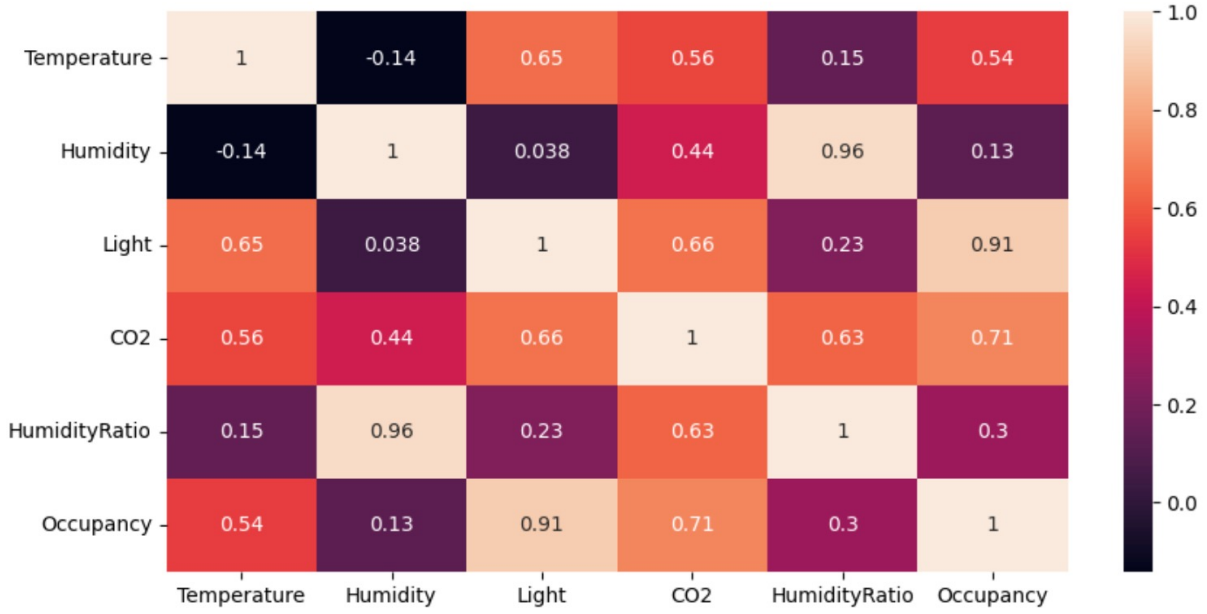






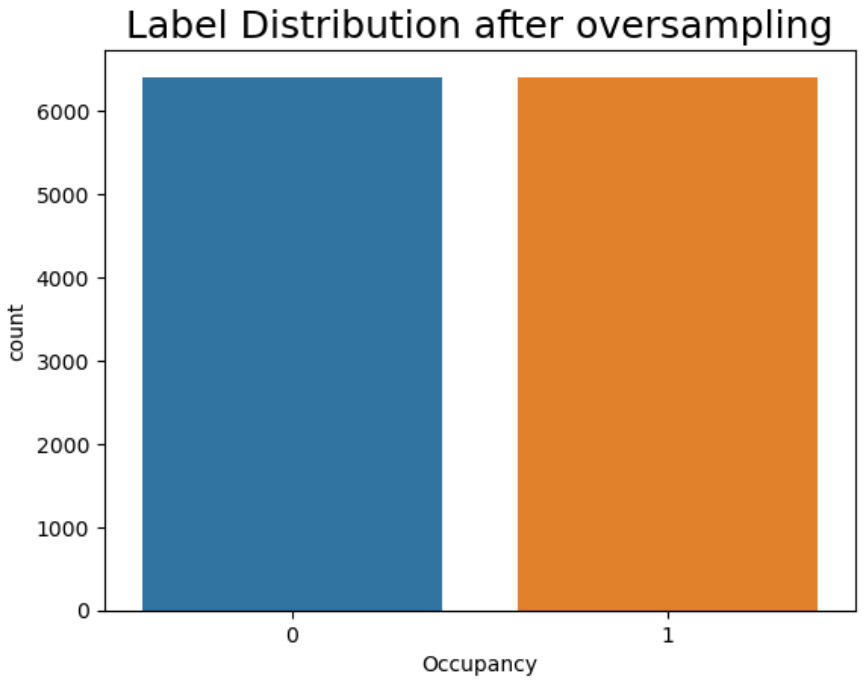
Exploratory Data Analysis (EDA)





Data Preparation

- Balancing the dataset using SMOTE technique
  - Synthetic Minority Over-sampling Technique as presented in [2]
- Normalizing the values



Method #1: RandomForest Classifier

- A RandomForest Classifier with 100 trees

Table 3: Validation scores for RandomForest Model

Metric	Value
Accuracy	0.99
Percision	0.99
Recall	0.99
F1 Score	0.99

Table 4: Test1 scores for RandomForest Model

Metric	Value
Accuracy	0.96
Percision	0.94
Recall	0.95
F1 Score	0.95

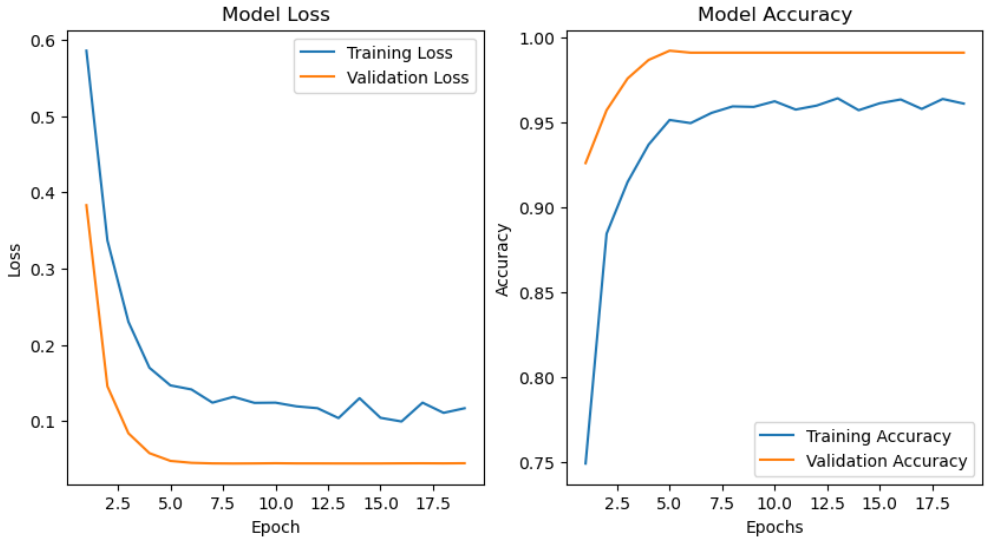
Table 5: Test2 scores for RandomForest Model

Metric	Value
Accuracy	0.96
Percision	0.87
Recall	0.98
F1 Score	0.92

Method #2: Multi-layer Perceptron

- The MLP model comprises three linear layers with ReLU for two first and sigmoid for the last layer

```
==========================================================================================
Layer (type:depth-idx)                   Output Shape              Param #
==========================================================================================
MLP                                      [1]                       --
├─Linear: 1-1                            [10]                      60
├─ReLU: 1-2                              [10]                      --
├─Dropout: 1-3                           [10]                      --
├─Linear: 1-4                            [10]                      110
├─ReLU: 1-5                              [10]                      --
├─Dropout: 1-6                           [10]                      --
├─Linear: 1-7                            [1]                       11
├─Sigmoid: 1-8                           [1]                       --
==========================================================================================
Total params: 181
Trainable params: 181
Non-trainable params: 0
Total mult-adds (M): 0.00
==========================================================================================
Input size (MB): 0.00
Forward/backward pass size (MB): 0.00
Params size (MB): 0.00
Estimated Total Size (MB): 0.00
==========================================================================================
```



Method #2: Multi-layer Perceptron

- The MLP model comprises three linear layers with ReLU for two first and sigmoid for the last layer

Table 6: Test1 scores for MLP

Metric	Value
Accuracy	0.97
Percision	0.94
Recall	0.99
F1 Score	0.97

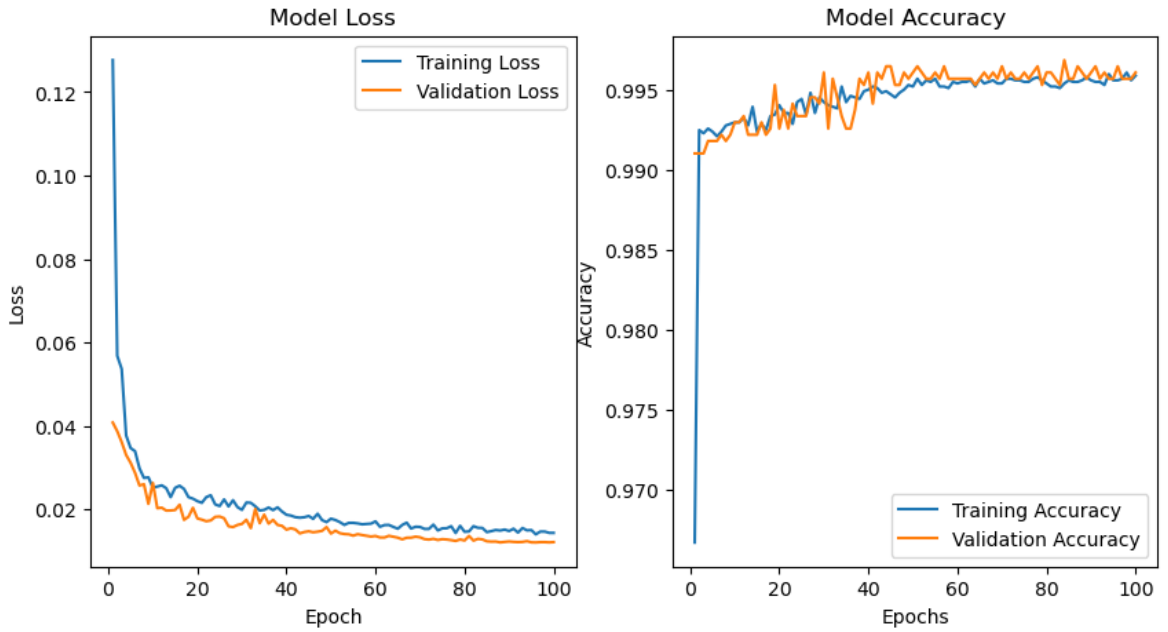
Table 7: Test2 scores for MLP

Metric	Value
Accuracy	0.99
Percision	0.96
Recall	0.99
F1 Score	0.98

Method #3: Convolutional Neural Network

- The model is defined with various layers (i.e., Conv1d, MaxPool1d, Dropout and Linear)

```
==========================================================================================
Layer (type:depth-idx)                   Output Shape              Param #
==========================================================================================
CNN                                      [1, 1]                    --
├─Conv1d: 1-1                            [1, 32, 5]                128
├─Conv1d: 1-2                            [1, 64, 5]                6,208
├─MaxPool1d: 1-3                         [1, 64, 2]                --
├─Dropout: 1-4                           [1, 128]                  --
├─Linear: 1-5                            [1, 64]                   8,256
├─Dropout: 1-6                           [1, 64]                   --
├─Linear: 1-7                            [1, 1]                    65
==========================================================================================
Total params: 14,657
Trainable params: 14,657
Non-trainable params: 0
Total mult-adds (M): 0.04
==========================================================================================
Input size (MB): 0.00
Forward/backward pass size (MB): 0.00
Params size (MB): 0.06
Estimated Total Size (MB): 0.06
==========================================================================================
```



Method #3: Convolutional Neural Network

- The model is defined with various layers (i.e., Conv1d, MaxPool1d, Dropout and Linear)

Table 8: Test1 scores for CNN

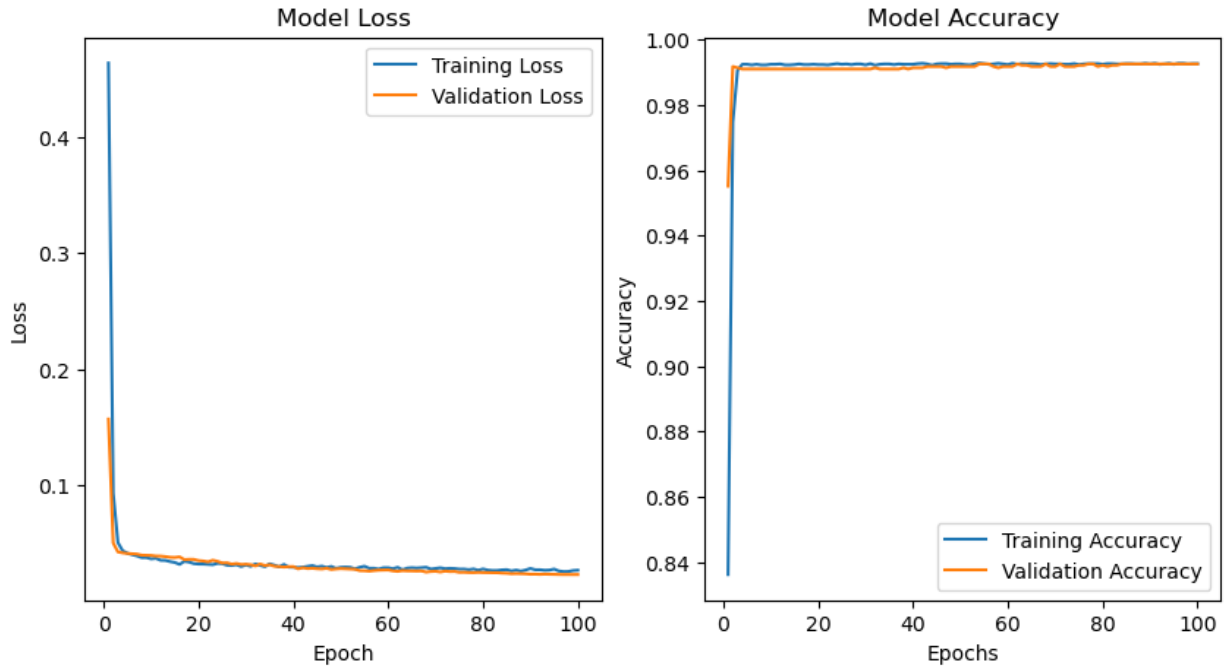
Metric	Value
Accuracy	0.93
Percision	0.92
Recall	0.87
F1 Score	0.89

Table 9: Test2 scores for CNN

Metric	Value
Accuracy	0.94
Percision	0.86
Recall	0.84
F1 Score	0.85

Method #4: Long Short-Term Memory

- A Long Short-Term Memory Network (LSTM), which is particularly suited for time-series prediction due to its ability to remember long-term dependencies.



Method #4: Long Short-Term Memory

- A Long Short-Term Memory Network (LSTM), which is particularly suited for time-series prediction due to its ability to remember long-term dependencies.

Table 10: Test1 scores for LSTM

Metric	Value
Accuracy	0.97
Percision	0.94
Recall	0.99
F1 Score	0.96

Table 11: Test2 scores for LSTM

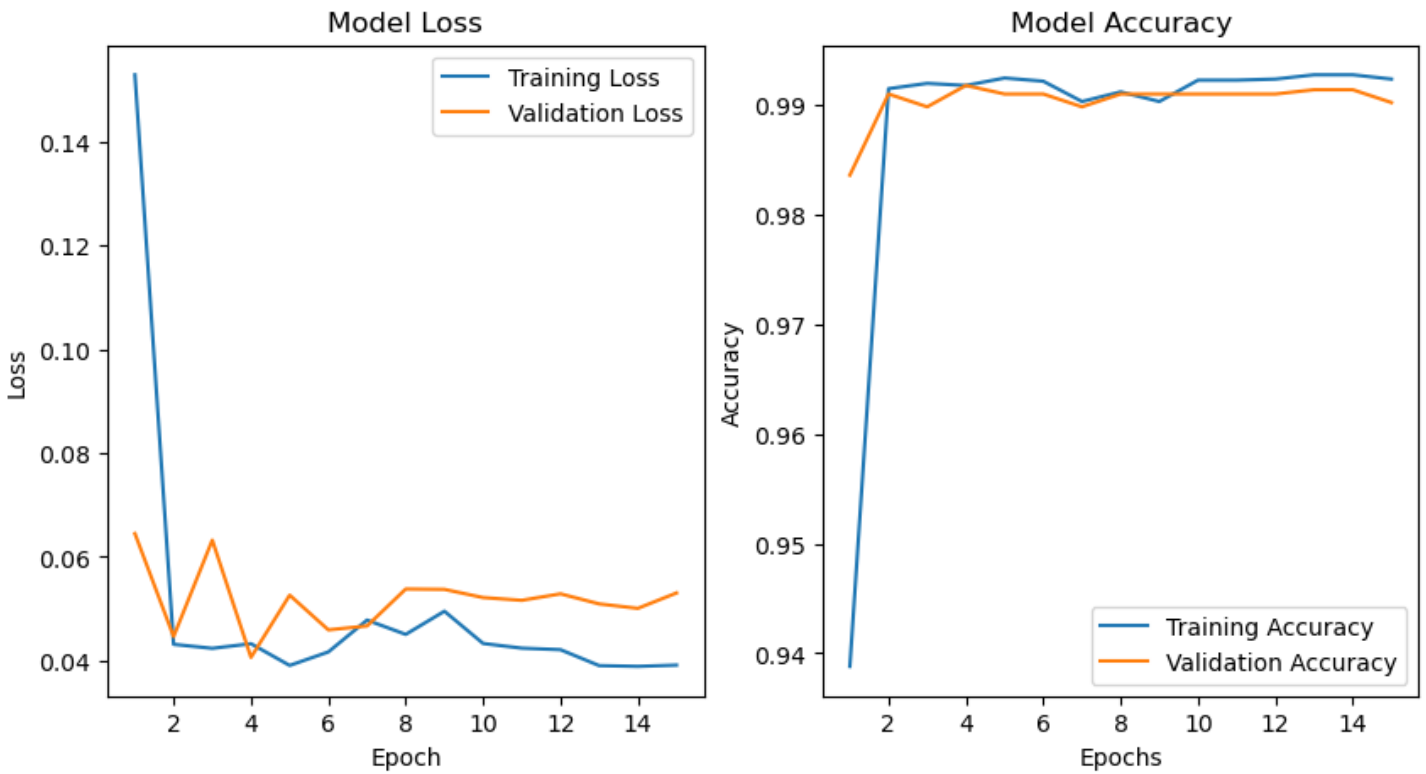
Metric	Value
Accuracy	0.95
Percision	0.82
Recall	0.99
F1 Score	0.90

Method #5: Transformer

- Contains a decoder and an Encoder
- Encoder: Three TransformerEncoderLayers
- Decoder: Six TransformerDecoderLayers
- Both encoder and decoder utilize MultiheadAttention to capture dependencies
- About 34 Million parameters

Method #5: Transformer

- Contains a decoder and an Encoder



Method #5: Transformer

- Contains a decoder and an Encoder

Table 12: Test1 scores for Transformer model

Metric	Value
Accuracy	0.97
Percision	0.94
Recall	0.98
F1 Score	0.96

Table 13: Test2 scores for Transformer model

Metric	Value
Accuracy	0.98
Percision	0.93
Recall	0.98
F1 Score	0.95

Summary

- This project utilizes a dataset to predict room occupancy
- Database classes were imbalanced, so we used SMOTE to turn it into a balanced dataset
- The dataset uses environmental observations such as temperature, humidity, and CO2 levels for this task
- This system can accurately detect the presence of occupants without using a camera (due to privacy concerns)
- We have developed and tested five different models for this problem which all of them look promising

References

1. Candanedo, Luis M., and Véronique Feldheim. "Accurate occupancy detection of an office room from light, temperature, humidity and CO2 measurements using statistical learning models." *Energy and buildings* 112 (2016): 28-39.
2. N. V. Chawla, K. W. Bowyer, L. O.Hall, W. P. Kegelmeyer, “SMOTE: synthetic minority over-sampling technique,” Journal of artificial intelligence research, 321-357, 2002.

Thank You!