# Evaluation of Various Algorithms' Performance in Supervised Binary Classification for Occupant Detection Using a Dataset from a Residential Building

Andersen, Kamilla Heimar; Schaffer, Markus; Johra, Hicham; Marszal-Pomianowska, Anna; O'Brien, William; Heiselberg, Per Kvols

DEPARTMENT OF THE BUILT ENVIRONMENT
AALBORG UNIVERSITY

# Evaluation of Various Algorithms' Performance in Supervised Binary Classification for Occupant Detection Using a Dataset from a Residential Building

Kamilla Heimar Andersen
Markus Schaffer
Hicham Johra
Anna Marszal-Pomianowska
William O'Brien
Per Kvols Heiselberg

ISSN 1901-726X
DCE Technical Report No. 319

Aalborg University
Department of the Built Environment
Division of Sustainability, Energy & Indoor Environment

**Technical Report No. 319**

# Evaluation of Various Algorithms' Performance in Supervised Binary Classification for Occupant Detection Using a Dataset from a Residential Building

by

Kamilla Heimar Andersen
Markus Schaffer
Hicham Johra
Anna Marszal-Pomianowska
William O'Brien
Per Kvols Heiselberg

September 2023

# Scientific Publications at the Department of the Built Environment

***Technical Reports*** are published for timely dissemination of research results and scientific work carried out at the Department of the Built Environment (DCE) at Aalborg University. This medium allows publication of more detailed explanations and results than typically allowed in scientific journals.

***Technical Memoranda*** are produced to enable the preliminary dissemination of scientific work by the personnel of the DCE where such release is deemed to be appropriate. Documents of this kind may be incomplete or temporary versions of papers—or part of continuing work. This should be kept in mind when references are given to publications of this kind.

***Contract Reports*** are produced to report scientific work carried out under contract. Publications of this kind contain confidential matter and are reserved for the sponsors and the DCE. Therefore, Contract Reports are generally not available for public circulation.

***Lecture Notes*** contain material produced by the lecturers at the DCE for educational purposes. This may be scientific notes, lecture books, example problems or manuals for laboratory work, or computer programs developed at the DCE.

***Theses*** are monograms or collections of papers published to report the scientific work carried out at the DCE to obtain a degree as either PhD or Doctor of Technology. The thesis is publicly available after the defence of the degree. Since 2015, Aalborg University Press has published all Ph.D. dissertations in faculty series under the respective faculty. The AAU Ph.D.-portal will host the E-books, where you also find references to all PhDs dissertations published from Aalborg University.

***Latest News*** is published to enable rapid communication of information about scientific work carried out at the DCE. This includes the status of research projects, developments in the laboratories, information about collaborative work and recent research results.

Published 2022 by
Aalborg University
Department of the Built Environment
Thomas Manns Vej 23
DK-9220 Aalborg E, Denmark

Printed in Aalborg at Aalborg University

ISSN 1901-726X
DCE Technical Report No. 319

# Contents

# Introduction

This technical report describes the evaluation process of various machine learning algorithms' performance used for supervised binary classification for occupant detection, using a dataset from a residential building in the North of Denmark. It supports the publication of *Development and Application of an XGBoost-Based Occupant Detection Model for Residential Buildings Using Supervised Learning* (sent to review in Building & Environment September 2023) [1].

# The dataset used to test the algorithms'

The dataset used in this study is Dataset 1 from the following repository [2], and is further described in the open-access technical report: [3]

The parameters used for the occupant detection models are indoor $CO_2$ concentration, indoor air temperature, indoor relative humidity, room type, and hour and day of the week. The models are further described in the sections below.

# Code for running the various algorithms

For the full code and documentation for both modeling approaches, see the following GitHub repository [2].

## Selection of algorithms for binary classification

Machine learning can be explored by several approaches depending on the type of problem to be solved, see Figure 1. This specific problem is delimited to a room-based occupant presence or absence (interpreted as 0 or 1 with interpreted ground truth) binary supervised classification.

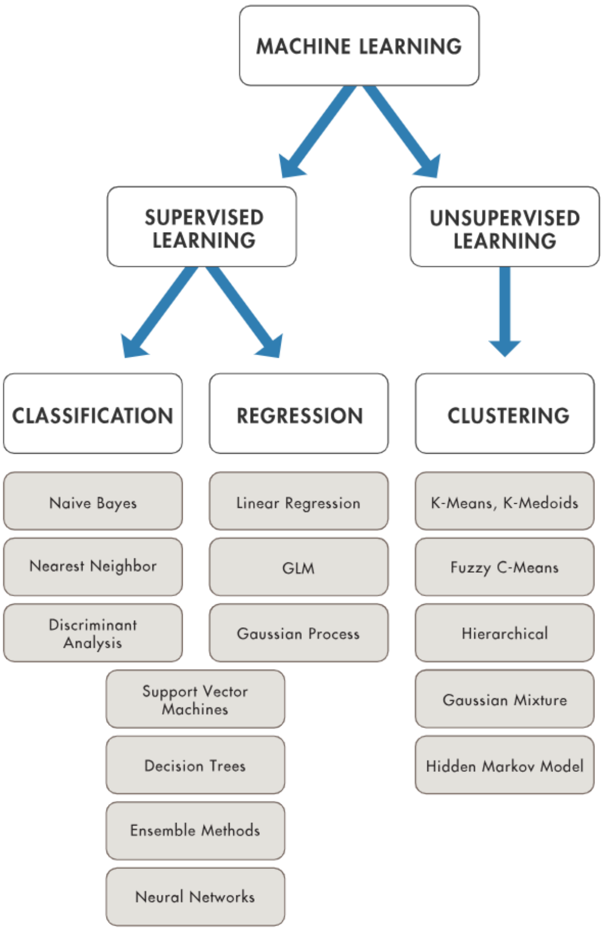


**Figure 1: General overview of supervised- and unsupervised learning based on [4] [5].**

Many unsupervised and supervised learning models have widely studied occupant detection problems in both residential- and non-residential buildings [6-10], [11, 12], [13]. The algorithms selected for this analysis were chosen based on 1) is tailored for supervised binary classification tasks, 2) what has existing literature explored earlier, 3) offer a diverse range of adjustable parameters and hyperparameters compatible with our dataset and 4) backed by comprehensive documentation and proven validation. Furthermore, it was important to consider the unique attributes of the dataset, the computational constraints, and the balance between model explainability and efficacy.

The choice of modeling approach often depends on the input data type (smart meter data, indoor environmental sensors, motion sensors, or others) and access to ground truth data. Supervised learning approaches use known inputs and output data. Here, classification or regression models are typically used. Existing algorithms in the literature are support vector machine (SVM), Naïve Bayes

(NB), Decision Trees (DTs), and Nearest Neighbour (NN). For supervised learning, labeled data (output) is necessary for testing the performance of the models. Traditional unsupervised learning approaches aim to find patterns, correlations, or structures in the input data. Hidden Markov Models, K-Means, and Hierarchical or physic-based algorithms are also used for occupant detection modeling. However, ground truth is always used to validate the developed method for this problem. [5] [14].

In the existing peer-reviewed (Scopus-indexed) literature with a focus on algorithms used for occupant detection in residential buildings, the following key algorithms were found (See Appendix) out of 19 articles:

- Support Vector variations or combinations were found in 9 articles
- Random Forest variations or combinations were found in 7 articles
- Variations of Neural Networks were used 6 times
- Gradient Boosting variations were used 5 times
- k-Nearest Neighbour was used 5 times
- Decision Tree variations were used 4 times
- Logistic Regression was used 2 times
- Naïve Bayes was used 2 times
- Other variations of algorithms found were Principal Component Analysis, expert systems (thresholds), and other unsupervised algorithms (Hidden Markov Models).

Supervised learning is predominantly used, as it was found in 12 out of 19 articles.

Therefore, the following algorithms were selected to be further tested with the selected dataset:

- Support Vector Machine
- Logistic Regression
- Random Forest
- k-Nearest Neighbor
- Naive Bayes
- XGBoost

The Decision Tree algorithm and Neural Network are neglected as it does not meet the criteria above.

A short outline of each algorithm is presented below [15, 16].

- **Logistic Regression (LR):** It models the relationship between the input features and the probability of the target class using a logistic function.
- **Support Vector Machine (SVM):** It finds an optimal hyperplane that separates the two classes by maximizing the margin between them.
- **Random Forest (RF):** Random Forest is an ensemble learning method that combines multiple decision trees. Each tree is trained on a random subset of features, and the final prediction is based on the majority vote of the trees.
- **k-Nearest Neighbor (kNN):** kNN is a type of instance-based learning method. The algorithm operates on the premise that similar instances will likely have similar outcomes. It finds the 'k' most similar instances in the training set for a given input and predicts based on their outcomes.
- **Naive Bayes (NB):** Naive Bayes is a probabilistic classifier based on Bayes' theorem. It assumes independence between features, given the class label.
- **Gradient Boosting Models (GBM):** Gradient boosting models, such as XGBoost, are powerful ensemble methods that combine multiple weak learners (decision trees) to make accurate predictions.

Table 1 describes a general overview of the selected algorithms for binary classification categorized by pros and cons [17, 18].

**Table 1: General overview of the selected algorithms for binary classification categorized by pros and cons.**

| Algorithm | Pros | Cons | References |
|---|---|---|---|
| Logistic Regression (LR) | • Interpretable results (f.ex. odd ratios)<br>• In general, fast training and prediction<br>• Typically works well with small to medium-sized datasets<br>• Less prone to overfitting with regularization with smaller number of features | • It may not capture complex relationships in the data<br>• Can be sensitive to irrelevant or highly correlated features<br>• Assumes a linear relationship between features and the target, which may not always be true<br>• May underperform with non-linear data | [19, 20] [5] |
| Support Vector Machine (SVM) | • Effective in high-dimensional spaces<br>• Works well with a clear margin of separation | • Computational complexity scales with the number of samples<br>• Requires appropriate kernel selection | [17, 19, 20] |

| | | | |
|---|---|---|---|
| | • Can be robust to overfitting with regularization<br>• Effective with small to medium-sized datasets<br>• Can handle non-linear relationships with kernel tricks | • Training and tuning time can be high for large datasets<br>• Generally memory-intensive for large datasets<br>• It can be difficult to interpret the resulting models (non-linear kernels)<br>• Can be sensitive to noise | |
| Random Forest (RF) | • Can handle high-dimensional data<br>• Can capture complex feature interactions<br>• Robust to outliers (due to averaging mechanism)<br>• Can handle imbalanced datasets (f.ex. bootstrapping)<br>• It might not require extensive feature engineering | • May overfit with noisy data<br>• Longer training time compared to some algorithms (due to the necessity to train multiple trees)<br>• Can lack interpretability compared to linear models due to the ensemble method of trees<br>• It may require tuning for optimal performance | [21, 22] |
| k-Nearest Neighbor (kNN) | • Simple and easy to understand<br>• There are no assumptions about the data distribution, making it useful for non-linear data<br>• Can perform well with a sufficient number of data points<br>• Model training is generally faster as it simply stores instances of the training data | • It can be computationally expensive and slow during the prediction phase, especially with large datasets<br>• Sensitive to the scale of the data and irrelevant features<br>• It needs appropriate choice of 'k' and the distance metric<br>• Performance can degrade with high-dimension data (curse of dimensionality)<br>• There is no model interpretability, as there is no explicit learning phase | [23] |
| Naïve Bayes (NB) | • Can have a fast training and prediction phase<br>• Can perform well with high-dimensional data<br>• Handles irrelevant features well (due to the probabilistic nature)<br>• Can work well with small to medium-sized datasets<br>• Low memory footprint | • Assumes independence between features<br>• It may not capture complex relationships (due to the naïve assumption)<br>• It relies on the strong independence assumption<br>• May underperform when independence assumption is violated | [24] |

| | | | |
|---|---|---|---|
| Gradient Boosting Models (GBM) | • Known for high performance and accuracy (if applied correctly)<br>• Can handle imbalanced datasets (functions and hyperparameters)<br>• It has a built-in feature importance analysis<br>• Has parallel processing for scalability<br>• It has a wide range of hyperparameters for the customization of model | • It can require more hyperparameter tuning compared to some other models<br>• It can be computationally expensive with large datasets<br>• Can require more data for training compared to simpler models<br>• Less interpretable compared to simpler models (due to the ensemble and layers of decision trees) | [21, 25, 26] |

## Chosen performance metrics

The nine selected performance metrics are outlined below. An outline of the reasoning for these performance metrics can be seen in the [1].

### 1. Accuracy [%]

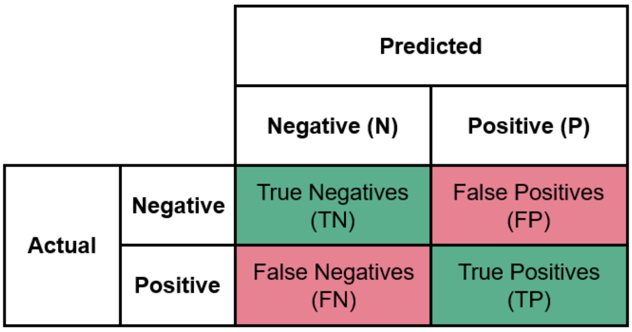
Calculated as the count of correct predictions over total number of predictions [27], see **Error! Reference source not found.**:

$$\boldsymbol{Accuracy = \frac{1}{n_{samples}} \sum_{i=0}^{n_{samples}-1} 1(\hat{y}_i = y_i)}$$

$$\mathrm{y} = \text{Ground truth occupancy}$$
$$\hat{\mathrm{y}} = \text{Predicted occupancy label}$$

### 2. Confusion matrix

The confusion matrix function evaluates classification accuracy by computing the confusion matrix with each row corresponding to the true class [28]. Figure 2 shows the confusion matrix and the classification of the predicted and actual classes.



| | | Predicted | |
|---|---|---|---|
| | | **Negative (N)** | **Positive (P)** |
| **Actual** | **Negative** | True Negatives (TN) | False Positives (FP) |
| | **Positive** | False Negatives (FN) | True Positives (TP) |

**Figure 2: Confusion matrix example.**

TP = The number of cases correctly predicted as positive

TN = The number of cases correctly predicted as negative

FP = The number of negative cases incorrectly predicted as positive

FN = The number of positive cases incorrectly predicted as negative

3. **Balanced accuracy [%]**

Calculated as the arithmetic mean of sensitivity (true positive rate) and specificity (true negative rate) [29, 30], see **Error! Reference source not found.**:

$$\boldsymbol{Balanced\ accuracy} = \frac{\mathbf{1}}{\mathbf{2}} * (\frac{\boldsymbol{TP}}{\boldsymbol{TP}+\boldsymbol{FN}} + \frac{\boldsymbol{TN}}{\boldsymbol{TN}+\boldsymbol{FP}})$$

4. **Precision [%]**

Calculated as the true positives divided by the true positives and the false positives [31], see **Error! Reference source not found.**:

$$\boldsymbol{Precision} = \frac{\boldsymbol{TP}}{\boldsymbol{TP}+\boldsymbol{FP}}$$

5. **Recall [%]**

Calculated as the true positives divided by the true positives and the false negatives [32], see **Error! Reference source not found.**:

$$\boldsymbol{Recall} = \frac{\boldsymbol{TP}}{\boldsymbol{TP}+\boldsymbol{FN}}$$

6. **F1-score [0-1]**

The F1 score can be interpreted as a harmonic mean of the precision and recall [33], see **Error! Reference source not found.**:

$$\boldsymbol{F1} = \mathbf{2} * \frac{(\boldsymbol{Precision} * \boldsymbol{Recall})}{(\boldsymbol{Percision} + \boldsymbol{Recall})}$$

7. **Matthew’s correlation coefficient (MCC) [(-1) – (+1)]**

The MCC is in essence a correlation coefficient value between -1 and +1. A coefficient of +1 represents a perfect prediction, 0 an average random prediction and -1 an inverse prediction [34, 35], see **Error! Reference source not found.**:

$$\boldsymbol{MCC} = \frac{(\boldsymbol{TP} * \boldsymbol{TN}) - (\boldsymbol{FP} * \boldsymbol{FN})}{\sqrt{(\boldsymbol{TP}+\boldsymbol{FP})(\boldsymbol{TP}+\boldsymbol{FN})(\boldsymbol{TN}+\boldsymbol{FP})(\boldsymbol{TN}+\boldsymbol{FN})}}$$

### 8. Area Under the Receiver Operating Characteristic Curve (AUROC) [0-1]

A receiver operating characteristic (ROC), or simply ROC curve, is a graphical plot that illustrates the performance of a binary classifier system as its discrimination threshold is varied. The ROC curve is plotted with the True Positive Rate (TPR), or Sensitivity, against the False Positive Rate (FPR) or 1 minus the Specificity, where TPR is on the y-axis and FPR is on the x-axis. AUC, or Area Under the Curve, is the area under the ROC curve. It represents the degree or measure of separability, indicating how well the model can distinguish between classes. [36, 37].

### 9. Briar-score loss [0-1]

The Brier score measures the mean squared difference between the predicted probability and the actual outcome [38], see Equation 8:

$$\boldsymbol{Brier - score\ loss} = \frac{\mathbf{1}}{\boldsymbol{nsamples}} * \sum_{\boldsymbol{i=0}}^{\boldsymbol{nsamples^{-1}}} * \ (\boldsymbol{yi - pi})^{\mathbf{2}}$$

$$yi = \text{ actual outcome}$$
$$pi = \text{ predicted probability estimate}$$

## Modeling framework

This modeling framework consists of a basic application of each algorithm in Python 3.11 on SPYDER 5 without any parameter or hyperparameter optimization. The Scikit-learn library was used for these model implementations [39]. The computer specifications for running the models were the following: Intel Core i7-10510U CPU and 16GB of RAM.

Two different modeling approaches are applied. One **generalized model** consisting of the following 12 parameters:

1. $CO_2$ concentration [ppm]
2. Indoor air temperature [°C]
3. Relative humidity [%]
4. Room type, one-hot-encoded (5 rooms)
5. Day of the week (cycled encoded) (two encodings)
6. Hour of the day (cycled encoded) (two encodings)

Each **room-based model** consists of the following 7 parameters:

1. $CO_2$ concentration [ppm]
2. Indoor air temperature [°C]
3. Relative humidity [%]
4. Day of the week (cycled encoded) (two encodings)
5. Hour of the day (cycled encoded) (two encodings)

For the generalized model, a 10-fold grouped shuffle split cross-validation [40] was conducted for all models, grouping days consistently within different folds across all models. Whereas for each room-based model, a 5-fold stratified grouped shuffle split cross-validation [41] was conducted with the aim to have balanced folds across the splits and also due to the smaller amount of data points and variations of data characteristics of the room types.

The split of the model selection and model evaluation is 80 % and 20 %, respectively. This cross-validation type was performed as the aim is to perform hyperparameter optimization after the evaluation of the algorithms. The reproducibility seed for all models is denoted as random_state = 42.

Figure 3 shows the overview of the cross-validation procedure.

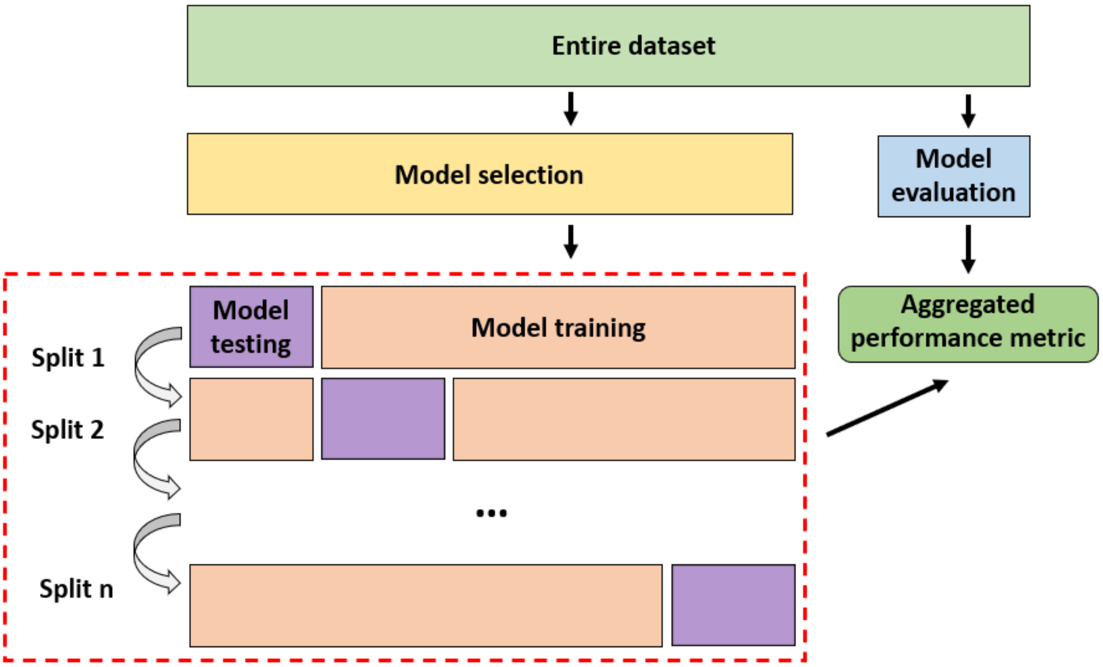


Figure 3: Cross-validation procedure for the modeling.

## Generalized model performance

The generalized model's algorithm modeling results can be found in Table 2 to Table 4, where algorithms are arranged from left to right based on their performance. Due to the imbalance of the majority/minority class of the dataset, particular emphasis was placed on balanced accuracy, MCC, precision, recall, and the F1-score to ensure a more robust evaluation of model performance. The second line under each result is the standard deviation of the 10-fold.

1. **$CO_2$ concentration, air temperature, and relative humidity**

Table 2: Results of the occupant detection modeling performance metrics for the generalized model using CO2 concentration, air temperature, and relative humidity (average performance of 10-fold grouped shuffle split). The standard deviation is placed below the performance metric.

| **Model / perform. And evaluat. metric** | **Support Vector Machine** | **k-Nearest Neighbor** | **Naïve Bayes** | **Logistic Regression** | **XGBoost** | **Random Forest** |
|---|---|---|---|---|---|---|
| **Accuracy** | 75 %<br>2.7 % | 78 %<br>0.020 | 80 %<br>2.1 % | 81 %<br>2.1 % | 88 %<br>0.9 % | 90 %<br>0.9 % |
| **Balanced accuracy** | 54 %<br>2.2 % | 69 %<br>0.031 | 70 %<br>3.8 % | 68 %<br>6.9 % | 81 %<br>2.5 % | 83 %<br>2.8 % |
| **Confusion matrix** | [1593 27]<br>[533 59] | [1440 179]<br>299 293] | [1475 145]<br>[306 286] | [1534 86]<br>[341 250] | [1535 84]<br>[191 401] | [1562 57]<br>[174 417] |
| **Precision** | 75 %<br>18 % | 62 %<br>4.9 % | 67 %<br>6.8 % | 76 %<br>6.8 % | 82 %<br>2.7 % | 88 %<br>2.2 % |
| **Recall** | 10 %<br>5 % | 49 %<br>8.5 % | 47 %<br>10 % | 41 %<br>10 % | 67 %<br>6.2 % | 69 %<br>6.3 % |
| **F1-score** | 0.16<br>0.077 | 0.54<br>0.058 | 0.55<br>0.066 | 0.52<br>0.066 | 0.74<br>0.036 | 0.77<br>0.039 |
| **MCC** | 0.19<br>0.070 | 0.41<br>0.051 | 0.44<br>0.051 | 0.45<br>0.051 | 0.66<br>0.032 | 0.72<br>0.035 |
| **AUC-ROC** | 0.72<br>0.050 | 0.76<br>0.022 | 0.77<br>0.033 | 0.78<br>0.033 | 0.92<br>0.016 | 0.93<br>0.017 |
| **Brier score loss** | 0.17<br>0.012 | 0.16<br>0.012 | 0.17<br>0.015 | 0.15<br>0.015 | 0.09<br>0.007 | 0.08<br>0.005 |
| **Comp. time** | 249 seconds | 1.5 seconds | 0.6 seconds | 1.6 seconds | 5.7 seconds | 6.9 seconds |

## 2. $CO_2$ concentration and air temperature

**Table 3: Results of the occupant detection modeling performance metrics for the generalized model using $CO_2$ concentration and air temperature (average performance of 10-fold grouped shuffle split). The standard deviation is placed below the performance metric.**

| Model / perform. and evaluat. metric | Support Vector Machine | k-Nearest Neighbor | Naïve Bayes | Logistic Regression | XGBoost | Random Forest |
|---|---|---|---|---|---|---|
| **Accuracy** | 75 %<br>2.6 % | 82 %<br>0.016 | 80 %<br>1.9 % | 81 %<br>2.3 % | 88 %<br>1.2 % | 89 %<br>0.8 % |
| **Balanced accuracy** | 54 %<br>2.1 % | 73 %<br>0.035 | 70 %<br>3.9 % | 70 %<br>3.5 % | 83 %<br>3 % | 83 %<br>2.2 % |
| **Confusion matrix** | [1594 26]<br>[534 58] | [1485 134]<br>[267 325] | [1478 142]<br>[309 283] | [1543 77]<br>[333 259] | [1526 94]<br>[169 423] | [1554 66]<br>[174 418] |
| **Precision** | 0.75<br>0.184 | 0.70<br>0.043 | 0.68<br>0.072 | 0.77<br>0.086 | 0.81<br>0.040 | 0.86<br>0.022 |
| **Recall** | 0.09<br>0.049 | 0.54<br>0.086 | 0.47<br>0.112 | 0.43<br>0.075 | 0.70<br>0.065 | 0.70<br>0.05 |
| **F1-score** | 0.16<br>0.075 | 0.61<br>0.062 | 0.54<br>0.068 | 0.55<br>0.073 | 0.76<br>0.046 | 0.77<br>0.030 |
| **MCC** | 0.19<br>0.069 | 0.50<br>0.056 | 0.44<br>0.050 | 0.47<br>0.072 | 0.68<br>0.048 | 0.71<br>0.026 |
| **AUC-ROC** | 0.72<br>0.056 | 0.81<br>0.025 | 0.77<br>0.034 | 0.78<br>0.042 | 0.93<br>0.013 | 0.93<br>0.019 |
| **Brier score loss** | 0.17<br>0.012 | 0.14<br>0.011 | 0.17<br>0.015 | 0.14<br>0.011 | 0.088<br>0.007 | 0.08<br>0.006 |
| **Comp. time** | 181.45 seconds | 1.42 seconds | 0.47 seconds | 1.60 seconds | 7.43 seconds | 6.48 seconds |

## 3. $CO_2$ concentration and relative humidity

Table 4: Results of the occupant detection modeling performance metrics for the generalized model using $CO_2$ concentration and relative humidity (average performance of 10-fold grouped shuffle split). The standard deviation is placed below the performance metric.

| Model / perform. and evaluat. metric | Support Vector Machine | k-Nearest Neighbor | Naïve Bayes | Logistic Regression | XGBoost | Random Forest |
|---|---|---|---|---|---|---|
| Accuracy | 75 %<br>2.6 % | 78 %<br>0.019 | 80 %<br>2.1 % | 80 %<br>1.7 % | 88 %<br>1.4 % | 89 %<br>1.1 % |
| Balanced accuracy | 54 %<br>2.2 % | 68 %<br>2.6 % | 70 %<br>4 % | 68 %<br>4.6 % | 81 %<br>2.6 % | 82 %<br>2.5 % |
| Confusion matrix | [1593 27]<br>[533 60] | [1434 182]<br>[306 285] | [1478 142]<br>[298 294] | [1521 100]<br>[334 258] | [1542 78]<br>[192 400] | [1554 66]<br>[184 408] |
| Precision | 0.74<br>0.183 | 0.61<br>0.050 | 0.68<br>0.069 | 0.73<br>0.074 | 0.83<br>0.023 | 0.85<br>0.020 |
| Recall | 0.09<br>0.050 | 0.47<br>0.075 | 0.48<br>0.110 | 0.42<br>0.118 | 0.67<br>0.058 | 0.68<br>0.053 |
| F1-score | 0.17<br>0.077 | 0.53<br>0.051 | 0.56<br>0.070 | 0.53<br>0.091 | 0.74<br>0.035 | 0.76<br>0.036 |
| MCC | 0.19<br>0.070 | 0.39<br>0.044 | 0.45<br>0.054 | 0.44<br>0.066 | 0.67<br>0.034 | 0.70<br>0.035 |
| AUC-ROC | 0.72<br>0.056 | 0.76<br>0.021 | 0.77<br>0.033 | 0.80<br>0.033 | 0.93<br>0.011 | 0.92<br>0.019 |
| Brier score loss | 0.17<br>0.012 | 0.16<br>0.011 | 0.17<br>0.017 | 0.14<br>0.010 | 0.093<br>0.009 | 0.090<br>0.006 |
| Comp. time | 227.84 seconds | 1.48 seconds | 0.44 seconds | 1.11 seconds | 5.49 seconds | 11.20 seconds |

## Room-based model performance

The results from the algorithm modeling for the room-based models can be found in Table 5 to Table 9, where algorithms are arranged from left to right based on their performance. Due to the imbalance of the majority/minority class of the dataset, particular emphasis was placed on balanced accuracy, MCC, precision, recall, and the F1-score to ensure a more robust evaluation of model performance. The second line under each result is the standard deviation of the 10-fold.

### 1. Bedroom model performance

Table 5: Results of the occupant detection modeling for the bedroom model performance metrics (average performance of 10-fold grouped shuffle split). The standard deviation is placed below the performance metric.

| Model / perform. and evaluat. metric | Support Vector Machine | k-Nearest Neighbor | Naïve Bayes | Logistic Regression | XGBoost | Random Forest |
|---|---|---|---|---|---|---|
| **Accuracy** | 78 %<br>3.5 % | 78 %<br>3.8 % | 89 %<br>3.3 % | 92 %<br>1 % | 93 %<br>1.5 % | 94 %<br>1.5 % |
| **Balanced accuracy** | 70 %<br>5.3 % | 74 %<br>5.2 % | 86 %<br>4.5 % | 91 %<br>1.6 % | 92 %<br>1.8 % | 92 %<br>2 % |
| **Confusion matrix** | [428 12]<br>[130 103] | [383 57]<br>[91 142] | [427 13]<br>[58 176] | [415 25]<br>[28 206] | [420 14]<br>[31 203] | [427 14]<br>[29 204] |
| **Precision** | 0.89<br>0.052 | 0.71<br>0.059 | 0.93<br>0.025 | 0.89<br>0.035 | 0.93<br>0.029 | 0.93<br>0.029 |
| **Recall** | 0.43<br>0.111 | 0.60<br>0.101 | 0.75<br>0.097 | 0.88<br>0.046 | 0.87<br>0.040 | 0.87<br>0.047 |
| **F1-score** | 0.58<br>0.109 | 0.65<br>0.080 | 0.83<br>0.065 | 0.88<br>0.013 | 0.90<br>0.018 | 0.90<br>0.030 |
| **MCC** | 0.52<br>0.091 | 0.49<br>0.096 | 0.77<br>0.067 | 0.83<br>0.020 | 0.85<br>0.029 | 0.86<br>0.031 |
| **AUC-ROC** | 0.79<br>0.064 | 0.80<br>0.058 | 0.97<br>0.006 | 0.97<br>0.004 | 0.98<br>0.006 | 0.98<br>0.006 |
| **Brier score loss** | 0.156<br>0.026 | 0.17<br>0.033 | 0.075<br>0.017 | 0.056<br>0.008 | 0.054<br>0.012 | 0.051<br>0.009 |
| **Comp. time** | 16.74 seconds | 0.74 seconds | 1.43 seconds | 1.34 seconds | 4.81 seconds | 3.47 seconds |

## 2. Kitchen / Living room model performance

Table 6: Results of the occupant detection modeling for the kitchen / living room performance metrics (average performance of 10-fold grouped shuffle split). The standard deviation is placed below the performance metric.

| Model / perform. and evaluat. metric | Support Vector Machine | k-Nearest Neighbor | Naïve Bayes | Logistic Regression | XGBoost | Random Forest |
|---|---|---|---|---|---|---|
| **Accuracy** | 67 %<br>7.3 % | 62 %<br>4.7 % | 78 %<br>3 % | 76 %<br>4.4 % | 84 %<br>2.4 % | 85 %<br>3 % |
| **Balanced accuracy** | 65 %<br>6 % | 60 %<br>3.7 % | 78 %<br>3.3 % | 76 %<br>4.6 % | 84 %<br>1.9 % | 85 %<br>2.5 % |
| **Confusion matrix** | [63 58]<br>[36 131] | [64 57]<br>[53 114] | [94 28]<br>[34 133] | [88 33]<br>[34 133] | [110 21]<br>[24 143] | [110 21]<br>[21 146] |
| **Precision** | 0.70<br>0.079 | 0.66<br>0.054 | 0.83<br>0.055 | 0.80<br>0.068 | 0.87<br>0.045 | 0.87<br>0.043 |
| **Recall** | 0.78<br>0.156 | 0.68<br>0.127 | 0.79<br>0.083 | 0.79<br>0.055 | 0.86<br>0.066 | 0.87<br>0.077 |
| **F1-score** | 0.72<br>0.093 | 0.66<br>0.081 | 0.81<br>0.032 | 0.79<br>0.051 | 0.86<br>0.027 | 0.87<br>0.036 |
| **MCC** | 0.33<br>0.135 | 0.21<br>0.075 | 0.56<br>0.069 | 0.52<br>0.090 | 0.70<br>0.045 | 0.71<br>0.055 |
| **AUC-ROC** | 0.71<br>0.061 | 0.66<br>0.040 | 0.87<br>0.040 | 0.86<br>0.040 | 0.92<br>0.025 | 0.93<br>0.022 |
| **Brier score loss** | 0.21<br>0.022 | 0.24<br>0.032 | 0.15<br>0.019 | 0.15<br>0.020 | 0.12<br>0.025 | 0.10<br>0.014 |
| **Comp. time** | 6 seconds | 0.97 seconds | 0.35 seconds | 1.68 seconds | 2.17 seconds | 2.96 seconds |

## 3. Living room model performance

**Table 7: Results of the occupant detection modeling for the living room performance metrics (average performance of 10-fold grouped shuffle split). The standard deviation is placed below the performance metric.**

| Model / perform. and evaluat. metric | Support Vector Machine | k-Nearest Neighbor | Naïve Bayes | Logistic Regression | Random Forest | XGBoost |
|---|---|---|---|---|---|---|
| **Accuracy** | 79 %<br>3.9 % | 78 %<br>2.9 % | 83 %<br>2.9 % | 83 %<br>2.9 % | 86 %<br>3.8 % | 86 %<br>3.2 % |
| **Balanced accuracy** | 50 %<br>0.3 % | 61 %<br>2.9 % | 66 %<br>7.5 % | 67 %<br>6.2 % | 73 %<br>5.1 % | 77 %<br>5.1 % |
| **Confusion matrix** | [381 0]<br>[99 0] | [345 36]<br>[67 33] | [359 22]<br>[58 41] | [354 27]<br>[55 44] | [361 20]<br>[48 51] | [352 29]<br>[37 63] |
| **Precision** | 0.2<br>0.400 | 0.47<br>0.103 | 0.69<br>0.122 | 0.64<br>0.096 | 0.72<br>0.088 | 0.69<br>0.071 |
| **Recall** | 0 | 0.32<br>0.075 | 0.38<br>0.183 | 0.42<br>0.159 | 0.51<br>0.104 | 0.62<br>0.117 |
| **F1-score** | 0 | 0.37<br>0.074 | 0.45<br>0.174 | 0.48<br>0.120 | 0.60<br>0.088 | 0.64<br>0.071 |
| **MCC** | 0 | 0.26<br>0.066 | 0.40<br>0.122 | 0.41<br>0.082 | 0.52<br>0.094 | 0.57<br>0.070 |
| **AUC-ROC** | 0.57<br>0.079 | 0.70<br>0.052 | 0.85<br>0.050 | 0.85<br>0.038 | 0.87<br>0.035 | 0.88<br>0.034 |
| **Brier score loss** | 0.15<br>0.001 | 0.16<br>0.016 | 0.11<br>0.018 | 0.11<br>0.015 | 0.10<br>0.021 | 0.11<br>0.028 |
| **Comp. time** | 7.09 seconds | 0.99 seconds | 0.74 seconds | 1.13 seconds | 3.66 seconds | 2.37 seconds |

## 4. Kitchen model performance

**Table 8: Results of the occupant detection modeling of the kitchen performance metrics (average performance of 10-fold grouped shuffle split). The standard deviation is placed below the performance metric.**

| Model / perform. and evaluat. metric | Support Vector Machine | k-Nearest Neighbor | Naïve Bayes | Logistic Regression | Random Forest | XGBoost |
|---|---|---|---|---|---|---|
| Accuracy | 81 %<br>4.6 % | 77 %<br>4.9 % | 78 %<br>4.1 % | 80 %<br>4.4 % | 79 %<br>4 % | 78 %<br>3.5 % |
| Balanced accuracy | 50 %<br>0 | 51 %<br>2.5 % | 53 %<br>5 % | 52 %<br>5.3 % | 52 %<br>1.9 % | 54 %<br>2.4 % |
| Confusion matrix | [391 0]<br>[90 0] | [366 25]<br>[82 8] | [365 27]<br>[78 12] | [380 11]<br>[86 1] | [376 16]<br>[83 7] | [365 26]<br>[77 13] |
| Precision | 0 | 0.23<br>0.085 | 0.28<br>0.220 | 0.14<br>0.222 | 0.30<br>0.142 | 0.33<br>0.147 |
| Recall | 0 | 0.09<br>0.052 | 0.13<br>0.097 | 0.06<br>0.122 | 0.07<br>0.048 | 0.14<br>0.062 |
| F1-score | 0 | 0.13<br>0.058 | 0.17<br>0.127 | 0.07<br>0.147 | 0.10<br>0.064 | 0.20<br>0.072 |
| MCC | 0 | 0.03<br>0.071 | 0.09<br>0.138 | 0.05<br>0.128 | 0.06<br>0.065 | 0.11<br>0.070 |
| AUC-ROC | 0.57<br>0 | 0.57<br>0.040 | 0.74<br>0.062 | 0.71<br>0.067 | 0.71<br>0.050 | 0.68<br>0.042 |
| Brier score loss | 0.16<br>0.031 | 0.17<br>0.034 | 0.14<br>0.023 | 0.14<br>0.022 | 0.14<br>0.019 | 0.17<br>0.027 |
| Comp. time | 4.56 seconds | 1.08 seconds | 0.33 seconds | 1.23 seconds | 3.78 seconds | 2.39 seconds |

## 5. Office model performance

Table 9: Results of the occupant detection modeling of the office performance metrics (average performance of 10-fold grouped shuffle split). The standard deviation is placed below the performance metric.

| Model / perform. and evaluat. metric | Support Vector Machine | k-Nearest Neighbor | Naïve Bayes | Logistic Regression | XGBoost | Random Forest |
|---|---|---|---|---|---|---|
| Accuracy | 96 %<br>2.2 % | 95 %<br>2.7 % | 95 %<br>3.4 % | 96 %<br>2.3 % | 94 %<br>2.8 % | 96 %<br>2.3 % |
| Balanced accuracy | 50 %<br>0 % | 50 %<br>1 % | 56 %<br>8.3 % | 50 %<br>0.1 % | 49 %<br>1.8 % | 53 %<br>4.3 % |
| Confusion matrix | [464 0]<br>[2 0] | [456 0]<br>[2 0] | [455 2]<br>[2 1] | [464 0]<br>[2 0] | [462 0]<br>[2 0] | [460 1]<br>[2 1] |
| Precision | 0 | 0.11<br>0.190 | 0.16<br>0.175 | 0 | 0.01<br>0.046 | 0 |
| Recall | 0 | 0.013<br>0.020 | 0.15<br>0.182 | 0 | 0.01<br>0.035 | 0 |
| F1-score | 0 | 0.022<br>0.034 | 0.15<br>0.170 | 0 | 0.01<br>0.039 | 0 |
| MCC | 0 | 0.012<br>0.059 | 0.13<br>0.160 | 0 | 0 | 0.15<br>0.17 |
| AUC-ROC | 0.45<br>0.206 | 0.58<br>0.078 | 0.84<br>0.095 | 0.83<br>0.105 | 0.53<br>0.202 | 0.82<br>0.112 |
| Brier score loss | 0.040<br>0.020 | 0.047<br>0.022 | 0.040<br>0.024 | 0.038<br>0.021 | 0.051<br>0.026 | 0.040<br>0.022 |
| Comp. time | 2.22 seconds | 0.71 seconds | 0.37 seconds | 0.67 seconds | 1.07 seconds | 2 seconds |

# Reflections

This technical report describes the evaluation of various algorithms' performance used for supervised binary classification for occupant detection using a dataset from a residential building in the North of Denmark. The following models were tested: Logistic Regression, Support Vector Machine, Random Forest, k-Nearest Neighbor, Naive Bayes, and XGBoost. Some reflections from the results of this technical report are presented below.

1. Algorithm performance and characteristics:

All models have varying performance, indicating that some algorithms do not have suitable characteristics or parameters for the chosen dataset. Furthermore, since there is no hyperparameter optimization, it makes the algorithm comparison less optimal but provides an indication of performance. However, some algorithms, such as Random Forest, are not directly sensitive to hyperparameters. Conversely, Support Vector Machine is very sensitive to hyperparameters, which can be reflected in the longer computational time (optimization of the kernel might be necessary).

2. Generalized model performance:
    - Three input variations were performed (solely $CO_2$ concentration, $CO_2$ concentration and relative humidity, and $CO_2$ concentration and air temperature). All three models have similar performance in all the various algorithms, varying from 54 % to 82 % balanced accuracy and from 0.19 to 0.72 MCC score.
    - The XGBoost and Random Forest models have the highest performance based on the performance metrics for all variations, and it is assumed that both algorithms are suitable for further modeling and exploration.

3. Room-based model performance:
    - There is a considerable variation in performance across the various algorithms. Generally, the models perform around 50 % to 93 % balanced accuracy and from 0 to 0.86 MCC score.
    - The bedroom room type-based model has the highest performance, while the office room type-based model has the poorest performance. Proper tuning might be required in some room-based models due to the data characteristics (imbalance ratio and internal correlations).
    - XGBoost and Random Forest have the highest performance based on the performance metrics, and it is assumed that both algorithms are suitable for further application. However, XGBoost has a more extensive library for hyperparameter tuning than some

of the other models and thus could be a suitable algorithm due to the varying room-based model performance.

# References

[1] K.H. Andersen, A. Marshal-Pomianowska, W. O'Brien, H. Johra, M. Shaffer, H. Knudsen, P.K. Heiselberg, Exploring Occupant Detection Model Generalizability for Residential Buildings Using Supervised Learning with IEQ Sensors.

[2] Kamilla Heimar Andersen, Simon Pommerencke Melgaard, GitHub Repository (2021), URL: https://github.com/aauphd2024/occupant_detection_modeling

[3] Kamilla Heimar Andersen, A. Marszal-Pomianowska, H. Knudsen, H. Johra, S.P. Melgaard, M.Z. Dahl, P.A. Hundevad, P.K. Heiselberg, Room-based Indoor Environment Measurements and Occupancy Ground Truth Datasets from Five Residential Apartments in a Nordic Climate (2023) https://vbn.aau.dk/da/publications/room-based-indoor-environment-measurements-and-occupancy-ground-t

[4] What Is Machine Learning? How it works, why it matters, and getting started, URL: https://se.mathworks.com/discovery/machine-learning.html

[5] I.H. Sarker, Machine Learning: Algorithms, Real-World Applications and Research Directions, SN COMPUT. SCI. 2 (2021) 16010.1007/s42979-021-00592-x.

[6] L. Rueda, K. Agbossou, A. Cardenas, N. Henao, S. Kelouwani, A comprehensive review of approaches to building occupancy detection, Building and environment. 180 (2020) 10696610.1016/j.buildenv.2020.106966.

[7] J. Xie, H. Li, C. Li, J. Zhang, M. Luo, Review on occupant-centric thermal comfort sensing, predicting, and controlling, Energy and buildings. 226 (2020) 11039210.1016/j.enbuild.2020.110392.

[8] A.N. Sayed, Y. Himeur, F. Bensaali, Deep and transfer learning for building occupancy detection: A review and comparative analysis, Engineering applications of artificial intelligence. 115 (2022) 10525410.1016/j.engappai.2022.105254.

[9] Dipti Trivedi, V. Badarla, Occupancy detection systems for indoor environments: A survey of approaches and methods, Indoor &#43; built environment. 29 (2020) 1053-1069 https://journals.sagepub.com/doi/full/10.1177/1420326X19875621

[10] Z. Chen, C. Jiang, L. Xie, Building occupancy estimation and detection: A review, Energy and buildings. 169 (2018) 260-27010.1016/j.enbuild.2018.03.084.

[11] H. Saha, A.R. Florita, G.P. Henze, S. Sarkar, Occupancy sensing in buildings: A review of data analytics approaches, Energy and buildings. 188-189 (2019) 278-28510.1016/j.enbuild.2019.02.030.

[12] Y. Jin, D. Yan, A. Chong, B. Dong, J. An, Building occupancy forecasting: A systematical and critical review, Energy and buildings. 251 (2021) 11134510.1016/j.enbuild.2021.111345.

[13] Muhammad Tirta Mulia, S.H. Supangkat, N. Hariyanto, A review on building occupancy estimation methods, ICTSS (Sep 2017) 1-7 https://ieeexplore.ieee.org/document/828887810.1109/ICTSS.2017.8288878

[14] Introduction to machine learning second edition10.1007/978-1-62703-748-8_7.

[15] Trevor Hastie, R. Tibshirani, J. Friedman, The Elements of Statistical Learning, Data Mining, Inference, and Prediction (2009), URL: https://hastie.su.domains/ElemStatLearn/

[16] A. Narassiguin, M. Bibimoune, H. Elghazel, A. Aussem, An extensive empirical comparison of ensemble learning methods for binary classification, Pattern Anal Applic. 19 (2016) 1093-112810.1007/s10044-016-0553-z.

[17] F. Colas, P. Brazdil, Comparison of SVM and Some Older Classification Algorithms in Text Classification Tasks, Springer, Boston, MA, 2006 https://doi.org/10.1007/978-0-387-34747-9_18

[18] N. Naik, S. Purohit, Comparative Study of Binary Classification Methods to Analyze a Massive Dataset on Virtual Machine, Procedia Computer Science. 112 (2017) 1863-187010.1016/j.procs.2017.08.232.

[19] Differentiate between Support Vector Machine and Logistic Regression, URL: https://www.geeksforgeeks.org/differentiate-between-support-vector-machine-and-logistic-regression/
[20] Logistic Regression Vs Support Vector Machines (SVM), URL: https://medium.com/axum-labs/logistic-regression-vs-support-vector-machines-svm-c335610a3d16
[21] The battle between Logistic Regression, Random Forest Classifier, XG Boost and Support Vector Machine has been concluded!, URL: https://medium.com/@nischitasadananda/the-battle-between-logistic-regression-random-forest-classifier-xg-boost-and-support-vector-46d773c70f41
[22] A. Narassiguin, M. Bibimoune, H. Elghazel, A. Aussem, An extensive empirical comparison of ensemble learning methods for binary classification, Pattern Anal Applic. 19 (2016) 1093-112810.1007/s10044-016-0553-z.
[23] Pros And Cons Of The K-Nearest Neighbors (KNN) Algorithm, URL: https://roboticsbiz.com/pros-and-cons-of-the-k-nearest-neighbors-knn-algorithm/
[24] Naive Bayes in Machine Learning [Examples, Models, Types], URL: https://www.knowledgehut.com/blog/data-science/naive-bayes-in-machine-learning
[25] Gradient Boosting Machines, URL: http://uc-r.github.io/gbm_regression
[26] Learn about Gradient Boosting, URL: https://datascience.fm/learn-beginner-gradient-boosting/
[27] Accuracy score, URL: https://scikit-learn.org/stable/modules/model_evaluation.html#accuracy-score
[28] Confusion matrix, URL: https://scikit-learn.org/stable/modules/model_evaluation.html#confusion-matrix
[29] Sklearn balanced accuracy - Python, URL: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.balanced_accuracy_score.html#sklearn.metrics.balanced_accuracy_score
[30] What is Balanced Accuracy? (Definition & Example), URL: https://www.statology.org/balanced-accuracy/
[31] Sklean Precision - Python, URL: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.precision_score.html#sklearn.metrics.precision_score
[32] Sklearn Recall - Python, URL: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.recall_score.html#sklearn.metrics.recall_score
[33] Sklearn F1-score - Python, URL: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html
[34] Davide Chicco, G. Jurman, The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation, BMC Genomics. 21 (2020).
[35] Sklean MCC - Python, URL: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.matthews_corrcoef.html
[36] Understanding AUC - ROC Curve, URL: https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5
[37] Classification: ROC Curve and AUC, URL: https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc
[38] Brier score loss, URL: https://scikit-learn.org/stable/modules/model_evaluation.html#brier-score-loss
[39] Machine Learning in Python, URL: https://scikit-learn.org/stable/
[40] Grouped Shuffle Split, URL: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GroupShuffleSplit.html
[41] Stratified Shuffle Split, URL: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.StratifiedShuffleSplit.html

[42] A. Mohammadabadi, S. Rahnama, A. Afshari, Indoor Occupancy Detection Based on Environmental Data Using CNN-XGboost Model: Experimental Validation in a Residential Building, Sustainability (Basel, Switzerland). 14 (2022) 1464410.3390/su142114644.

[43] S.Y. Tan, M. Jacoby, H. Saha, A. Florita, G. Henze, S. Sarkar, Multimodal sensor fusion framework for residential building occupancy detection, Energy and buildings. 258 (2022) 11182810.1016/j.enbuild.2021.111828.

[44] Margarite Jacoby, S.Y. Tan, G. Henze, S. Sarkar, A high-fidelity residential building occupancy detection dataset, Sci Data. 8 (2021).

[45] Christian Beckel, W. Kleiminger, R. Cicchetti, T. Staake, S. Santini, The ECO data set and the performance of non-intrusive load monitoring algorithms, Proceedings of the 1st ACM Conference on embedded systems for energy-efficient buildings (Nov 03, 2014) 80-89 http://dl.acm.org/citation.cfm?id&#61;267406410.1145/2674061.2674064

[46] L.M. Candanedo, V. Feldheim, Accurate occupancy detection of an office room from light, temperature, humidity and CO2 measurements using statistical learning models, Energy and buildings. 112 (2016) 28-3910.1016/j.enbuild.2015.11.071.

[47] Z. Li, B. Dong, A new modeling approach for short-term prediction of occupancy in residential buildings, Building and environment. 121 (2017) 277-29010.1016/j.buildenv.2017.05.005.

[48] Guoming Tang, K. Wu, J. Lei, W. Xiao, The meter tells you are at home! Non-intrusive occupancy detection via load curve data, 2015 IEEE International Conference on Smart Grid Communications (SmartGridComm) (2015-11) 897-90210.1109/SmartGridComm.2015.7436415.

[49] M. Jin, R. Jia, C.J. Spanos, Virtual Occupancy Sensing: Using Smart Meters to Indicate Your Presence, IEEE Transactions on Mobile Computing. 16 (2017) 3264-327710.1109/TMC.2017.2684806.

[50] Dong Chen, S. Barker, A. Subbaswamy, D. Irwin, P. Shenoy, Non-Intrusive Occupancy Monitoring using Smart Meters, Proceedings of the 5th ACM Workshop on embedded systems for energy-efficient buildings (Nov 11, 2013) 1-8 http://dl.acm.org/citation.cfm?id&#61;252829410.1145/2528282.2528294

[51] R. Razavi, A. Gharipour, M. Fleury, I.J. Akpan, Occupancy detection of residential buildings using smart meter data: A large-scale study, Energy and buildings. 183 (2019) 195-20810.1016/j.enbuild.2018.11.025.

[52] B. Huchuk, S. Sanner, W. O&#39;Brien, Comparison of machine learning models for occupancy prediction in residential buildings using connected thermostat data, Building and environment. 160 (2019) 10617710.1016/j.buildenv.2019.106177.

[53] H. Zhou, J. Yu, Y. Zhao, C. Chang, J. Li, B. Lin, Recognizing occupant presence status in residential buildings from environment sensing data by data mining approach, Energy and buildings. 252 (2021) 11143210.1016/j.enbuild.2021.111432.

[54] Wilhelm Kleiminger, C. Beckel, T. Staake, S. Santini, Occupancy Detection from Electricity Consumption Data, Proceedings of the 5th ACM Workshop on embedded systems for energy-efficient buildings (Nov 11, 2013) 1-8 http://dl.acm.org/citation.cfm?id&#61;252829510.1145/2528282.2528295

[55] L.M. Candanedo, V. Feldheim, D. Deramaix, A methodology based on Hidden Markov Models for occupancy detection and a case study in a low energy residential building, Energy and buildings. 148 (2017) 327-34110.1016/j.enbuild.2017.05.031.

[56] C. Wang, J. Jiang, T. Roth, C. Nguyen, Y. Liu, H. Lee, Integrated sensor data processing for occupancy detection in residential buildings, Energy and buildings. 237 (2021) 11081010.1016/j.enbuild.2021.110810.

[57] D. Calì, P. Matthes, K. Huchtemann, R. Streblow, D. Müller, CO2 based occupancy detection algorithm: Experimental analysis and validation for office and residential buildings, Building and environment. 86 (2015) 39-4910.1016/j.buildenv.2014.12.011.

[58] S.I. Kampezidou, A.T. Ray, S. Duncan, M.G. Balchanos, D.N. Mavris, Real-time occupancy detection with physics-informed pattern-recognition machines based on limited CO2 and temperature sensors, Energy and buildings. 242 (2021) 11086310.1016/j.enbuild.2021.110863.

[59] Y. Jeon, C. Cho, J. Seo, K. Kwon, H. Park, S. Oh, I. Chung, IoT-based occupancy detection system in indoor residential environments, Building and environment. 132 (2018) 181-20410.1016/j.buildenv.2018.01.043.

[60] J. Jiang, C. Wang, T. Roth, C. Nguyen, P. Kamongi, H. Lee, Y. Liu, Residential House Occupancy Detection: Trust-Based Scheme Using Economic and Privacy-Aware Sensors, JIoT. 9 (2022) 1938-195010.1109/JIOT.2021.3091098.

[61] Tianyu Zhang, A. Al Zishan, O. Ardakanian, ODToolkit, Proceedings of the Tenth ACM International Conference on future energy systems (Jun 15, 2019) 35-46 http://dl.acm.org/citation.cfm?id&#61;332828010.1145/3307772.3328280

[62] Yan Gao, A. Schay, D. Hou, Occupancy Detection in Smart Housing Using Both Aggregated and Appliance-Specific Power Consumption Data, ICMLA (Dec 2018) 1296-1303 https://ieeexplore.ieee.org/document/861423510.1109/ICMLA.2018.00210

# Appendix: Literature review

Table 10: Found literature (Scopus-indexed) on occupant detection in residential buildings.

| Reference | Published | Sensor(s) | Scope / application | Learning approach | Type of ground truth data | Length of dataset | Data resolution | Model(s) | Dataset open access | Building level | Citations per August 2023 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| [42] | 2022 | CO2 concentration, air temperature and relative humidity | Develop a model | Supervised | Occupant survey | Two weeks | 5 min. | CNN-XGBoost, CNN, XGBoost, Logistic Regression, Decision Tree, K-Nearest Neighbor, Random Forest, SV regression, K-means clustering, Gradient boosting | No | Room-based | 2 |
| [43] | 2022 | Air temperature, light, relative humidity, audio and images | Develop a model | Supervised | Uses dataset from [44], [45] and [46] | Uses dataset from [44], [45] and [46] | 10 seconds to 8 kHz for audio | Occ.STPN, Random Forest, Few-shot model | [44] yes | Room-based | 9 |
| [47] | 2017 | Time User Survey data | Develop a model for occupancy forecast | Supervised | Motion sensor (PIR) | From 1 to four months | 5 min. | Markov Model, Artificial Neural Network (ANN) and Support Vector Regression (SVR) | No | Room-based | 73 |
| [48] | 2015 | Power use of appliances and human-activated switching | Develop a model | Unsupervised but tested supervised algorithms | Google+ GPS module | Two months | 10 seconds | Load curve data and readily-available appliance knowledge | No | Apartment-based | 15 |

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | Decision tree, SVM, KNN, Bayes | | | |
| [45] | 2014 | Power use | Development of a framework for non-intrusive monitoring | Semi-, unsupervised and supervised | Uses simulated, real and lab data (three diff. datasets) for the algorithms<br>ECO: Tablet + PIR | -<br>ECO: 8 months, 6 households | All datasets: 1 second | Parson, Baranski, Weiss and Kolter (HMM and clustering) | No<br>ECO: Yes | Apartment-based | 178 |
| [49] | 2017 | Power use | Develop a model | Unsupervised | Uses dataset from [45] | Uses dataset from [45] | 1 second | Base Learning (BL), Non-intrusive Learning (NL), and Transfer Learning (TL)<br>SVM-PCA, Random Forest | - | Apartment-based | 64 |
| [50] | 2013 | Electricity use | Development of a NIOM framework | Unsupervised | Occupant actions and GPS (30 seconds ground truth) | Two household<br>Not reported length? | Not reported? | Night as baseload, statistics | - | Apartment-based | 88 |
| [51] | 2019 | Electricity use | Test models with ECO, DRED and Smart* datasets | Supervised | See datasets | 18 months 500 households<br>Customer Behaviour Trials (CBT) in Ireland | 30 minutes | Random Forests, Gradient Boosting, ANN, SVM, KNN | - | Apartment-based | 91 |
| [52] | 2019 | Thermostat data from Eco Bee (Donate you Data)<br>Motion sensor (PIR) | Compare different algorithms | Semi- and supervised | Assumed GT with the thermostat data from Eco Bee (Donate you Data) | - | 30. min | Logistic Regression, Random Forest, Markov model (MM), the hidden Markov | No | House-based | 94 |

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | Motion sensor (PIR) | | | model (HMM), and the recurrent neural network (RNN) | | | |
| [53] | 2021 | Indoor environment data: CO2, air temperature, illuminance, relative humidity and noise | Information of occupant presence could be extracted from indoor environment data by appropriate data mining approach | Supervised | GPS, app on phone | Approx. 1 month | 5. minutes | Decision tree and curve description | No | Three bedrooms | 3 |
| [54] | 2013 | Electricity use | Develop a model | Unsupervised and supervised | PIR and Android application installed on phone | 8. months | 1 second | SVM, KNN, HMM, prior knowledge and threshold | No | House-based | 282 |
| [55] | 2017 | Air temperature, humidity, humidity ratio, CO2 and light time series data | Average occupancy schedules | Unsupervised | Not relevant | Uses dataset from [46] | Uses dataset from [46]<br>Varies the resolution of 5 min, 10 min, 20 and 30. min | Hidden Markov Models | Yes | Room-based | 107 |
| [56] | 2021 | Temperature and motion + human-activities: door handle touch, water usage, and motion near the door area | Develop a two-layer model | Supervised | Switches that people have to activate | 54 days in a living lab | Around seconds, cannot properly find | Random Forest, Decision Tree, K-Nearest Neighbor, and Support Vector Machine | No | Living lab | 29 |
| [57] | 2015 | CO2 concentration, fan power, room temperature, window opening | Develop a model | Unsupervised | Not applicable | Up to 6 days | 1. minute | Mass balance | No | a kitchen and a big sleeping/living room of a residential building without mechanical ventilation | 174 |

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| [58] | 2021 | CO2 and air temperature | Develop a model for real-time occupancy | Supervised | Occupants' logs | 7 days | 5. seconds | physics-informed pattern-recognition machine (PI-PRM) | No | Experimental setup (bedroom) | 17 |
| [59] | 2018 | Particulate matter + IE data | Develop a model | Unsupervised | Not applicable | 4 months | 1. minute | Point extraction algorithm is proposed to construct triangular shapes | No | Three studio type houses | 64 |
| [60] | 2022 | Occupant behavior (indoor handle temp., outdoor temp. handle) | Develop a model | Supervised/ unsupervised | PIR sensor | 40 days | Not applicable, event-based | Trustworthy sequence matching module | No | Living lab: living room, a kitchen, a bedroom, a bathroom and a mechanical room | 4 |
| [61] | 2019 | CO2, air temperature, relative humidity, humidity ratio, | Toolkit | Domain-adaptive and supervised | Uses five datasets to develop the toolkit, see paper. Uses mainly camera and app with GPS | From 7 days to 1 year | From 10 seconds to 15 minutes | HMM, Particle Filtering (PF), SVM, Random Forest, Sparse Non-negative Matrix Factorization (SNMF), ANN and RNN/LSTM | Yes | 35 various rooms | 7 |
| [62] | 2018 | Aggregated and appliance-specific power use data | Develop a model | Supervised | Uses the ECO data and e Clarkson smart housing dataset | Clarkson dataset: 12 days, GPS signal for GT | Clarkson dataset: 5 minutes, for el-use: unsure | Own app, prior-knowledge, SVM versions (PCA and SFS) | Yes | Apartment-based | 11 |

# Recent publications in the DCE Technical Report Series

ISSN 1901-726X
DCE Technical Report No. 319