Master Project Business Analytics

Energy Poverty Prediction in the Netherlands with Machine Learning Models

Author: Nasrin Rastgoo (2732567)

1st supervisor:	Dr. Karine Miras	
daily supervisor:	Dr. Francesco Dalla Longa	(TNO)
2nd reader:	Dr. Alessandro Zocca	

A thesis submitted in fulfilment of the requirements for the Vrije Universiteit Amsterdam Master of Science degree in Business Analytics-Computational Intelligence.

August 8, 2023

Preface

This thesis aims to fulfill the requirements of the Master of Business Analytics program at Vrije Universiteit Amsterdam. The Business Analytics program introduces a combination of computer science, mathematics, and business management techniques to aid in identifying and resolving business issues. This thesis aims to merge academic research with practical solutions to help the internship company and develop my expertise in this study's field. This graduation internship took place at TNO from February 2023 till August 2023. This research mainly focuses on predicting energy poverty in the Netherlands with machine learning models.

Abstract

The concept of energy poverty contains multiple dimensions that make households susceptible to high energy costs. It involves assessing affordability, energetic quality, and the capacity to invest in improving the quality of accommodation. This thesis endeavors to assess the predictive accuracy of machine learning models for energy poverty and identify the most influential features for effective prediction. By accurately predicting energy-poor households, policymakers can allocate budgets more efficiently and conduct scenario analyses based on diverse features.

To achieve this objective, data from CBS Microdata for the years 2019 and 2020 were utilized. A range of machine learning models, such as Decision Tree, Random Forest, Extreme Gradient Boosting, Logistic Regression, and Single-layer Network (Perceptron), were employed in the study. Special attention was given to addressing imbalanced data issues, and appropriate techniques were implemented to ensure robustness in the model performance.

By employing these methodologies, this thesis aims to enhance our understanding of energy poverty and provide valuable insights for policy planning and resource allocation.

Contents

List of Figures

List of Tables

Glossary

AUPRC	Area Under Precision Recall Curve
CBS	Dutch Central Bureau of Statistics
CW	Class weight
DT	Decision Tree Model
EP	Energy Poverty
FS	Feature Selection
HH	Household
LIHELEK	Low income with high energy bill /or low energy quality
LR	Logistic Regression model
ML	Machine Learning
PRC	Precision-Recall Curve
RF	Random Forest model
RFE	Recursive Feature Elimination
XGBoost	Extreme Gradient Boosting model

1 Introduction

Following the implementation of the Paris Agreement [4], Energy Poverty (EP) has gained significant prominence as countries globally work towards an equitable transition to a low-carbon society. The recent escalation in gas prices due to the conflict between Ukraine and Russia has further heightened households' susceptibility to high energy expenses, impacting various European nations, including the Netherlands. Consequently, there is a growing need for suitable metrics and tools to assess, monitor and predict energy poverty effectively, enabling the formulation of appropriate policy measures to address this issue. To effectively tackle this issue, developing comprehensive metrics and instruments for evaluating and monitoring energy poverty is crucial.

Energy Poverty encompasses three key dimensions: energy affordability, the energy efficiency of households (such as house insulation), and the ability to invest in improving energy efficiency [5]. Research by Mulder et al. [5] at TNO suggests that while the number of households currently affected by high energy costs is relatively small, energy poverty is expected to increase due to rising energy prices and the growing demand for energy-efficient homes in the context of the energy transition. This problem highlights the need for policymakers to have accurate predictions of energy poverty and a clear understanding of the factors that contribute to it.

Given the multidimensionality of energy poverty, machine learning algorithms offer a possible tool. They are particularly suitable for handling large datasets [6] and capturing complex relationships [7]. Additionally, applying eXplainable Artificial Intelligence (XAI) methods allows for gaining valuable insights into the intricate connections between the inputs and outputs of machine learning models, thereby facilitating the understanding of complex systems [7].

Previously, two significant studies on energy poverty have been conducted at TNO. In one study, Dalla Longa et al.[8] developed a system that utilizes machine learning techniques to assess the likelihood of energy poverty, taking into account factors such as income and energy expenditure. Although their datasets were limited, the use of a machine learning algorithm, specifically gradient boosting, provided valuable insights. Another study by Mulder et al.[5] focused on building a systematic framework for quantifying energy poverty through multidimensional indicators. Based on their findings [5, 9], the Ministry of Economic Affairs and Climate Policy (EZK) commissioned the Central Bureau of Statistics (CBS)[1] to develop an annual Energy Poverty Monitor [10]. In summary, the first study utilized a machine learning algorithm for energy poverty prediction. Still, it had limitations due to a restricted dataset and the absence of an officially accepted definition. On the reverse side, the second study developed a framework for energy poverty using microdata, but it did not incorporate machine learning models. During this internship and thesis, we aimed to bridge the gap between these two approaches by applying machine learning to microdata while adopting the new energy poverty definition.

Our primary research objective in this study is to predict energy poverty using machine learning models. We focus on utilizing the socio-economic features of households from microdata and consider the new definitions related to energy poverty. The study is centered around two prediction scenarios: same-year and next-year. In the same-year predictions, the models are trained separately on data from 2019 and 2020, and the goal is to predict energy poverty for the same year in which the training data was collected. On the other hand, in the next-year prediction scenario, the models are trained on data from the year 2019 and then used to predict energy poverty for the year 2020. The aim is to leverage the power of machine learning algorithms to predict energy poverty based on relevant socio-economic variables accurately.

The following research questions will be addressed in this study to achieve the mentioned objective:

RQ. 1. How accurate are machine learning models for predicting energy poverty in the Netherlands?

RQ. 2. Which features have high predictive power for energy poverty when using machine learning algorithms?

By addressing these research questions, the study aims to provide insights into the effectiveness of machine learning models for predicting energy poverty and contribute to the existing knowledge in the field.

[1]Monitor Energiearmoede 2020

This study was carried out at the Energy Transition Studies group of TNO. TNO is Netherlands Organisation for Applied Scientific Research[1] an independent research organisation focusing on applied science. Energy Transition Studies group develop non-technological knowledge, methods, and tools, in the techno-economic, socio-economic, and social science fields to accelerate the energy transition with knowledge institutions, companies and the government. They also develop and use quantitative methods and models to support the analyses in this sector. For example, Energy Transition Studies has integrated energy models on different geographical scales (the world, Europe, the Netherlands) and models for different sectors (industry, electricity, the built environment, transport).

The structure of this study is described below.

Chapter 2 introduces the concepts of energy poverty, including its definition and the key dimensions contributing to it. It also provides an overview of the machine learning models used in the study and the feature selection method employed. This chapter serves as a foundation for understanding the subsequent chapters.

Chapter 3 presents a comprehensive review of related work conducted in the field of energy poverty. It explores existing research and studies that have examined various aspects of energy poverty, providing a context for the current research and highlighting the gaps and opportunities for further investigation.

Chapter 4 outlines the implementation approaches adopted in the study. It begins with explaining the dataset preparation, including data collection and preprocessing steps. The chapter then describes the experimental setup used for the machine learning models and the evaluation metrics employed to assess their performance.

Chapter 5 discusses the performance of five different types of models across different predictions, providing insights into their effectiveness in predicting energy poverty. Then focuses on the statistical analysis of the features that derived from the feature selection.

Chapter 6 concludes the thesis, summarizing the findings and insights drawn from the previous chapters.

[1]Nederlandse Organisatie Voor Toegepast Natuurwetenschappelijk Onderzoek

2

Background

2.1 Definition and measurement of energy poveryt

The energy poverty measurements, data and analysis in this study are in line with the Energy Poverty Monitor of the Central Bureau of Statistics (CBS)[1]. CBS provides the datasets for 2019 and 2020.

Regarding the previous studies of energy poverty in the Netherlands [8], [7], earlier researches in TNO [11], [9], [5], and based on the Policy Report of the European Commission "Energy poverty and vulnerable consumers in the energy sector across the EU: analysis of policies and measures" (2015) [12], the energy poverty has three main dimensions:

1. The affordability of energy;
2. The energetic quality of the house;
3. The ability to invest in the energy quality of the house.

Therefore, energy poverty is measured via income situation, energy consumption and the energy quality of homes for households in the Netherlands [11]. "In 2022, the Ministry of Economic Affairs and Climate Policy (EZK) commissioned CBS to develop an annual Energy Poverty Monitor, based on tno research (2021) [10]." In this study, Energy Poverty Monitor datasets from 2019 and 2020 are used. Regarding dimensions of energy poverty, table 2.1 shows all the indicators with some variations and combinations.

In this thesis, the indicator chosen to measure energy poverty is "LIHELEK". This specific indicator was selected to ensure convenience and consistency in predictions and analyses across all datasets and models. LIHELEK, which stands for "Low income with high energy bill &/or low energy quality" is a combination of the indicators LIHE and LILEK. It encompasses the presence of both low income (LI) with high energy costs (HE)

[1]Monitor Energiearmoede 2020

Table 2.1: All energy poverty indicators

Indicator	**Definition**	**Household 2019**	**Household 2020**
LI	Low income	15.80%	15.18%
WI	Few investment opportunities	48.29%	46.10%
HE	High enrgy bill	47.51%	30.22%
LEK	Low energy quality home	50.21%	46.60%
ZLEK	LEK-very	15.47%	15.41%
LIHE	Low income with high energy bill	6.06%	3.20%
LILEK	Low income with low energy quality	5.44%	5.04%
LIZLEK	LILEK-very	0.62%	0.69%
LEKWI	Low energy quality with few investment	19.75%	17.61%
ZLEKWI	LEKWI-very	3.02%	2.99%
LIHELEK	Low income with high energy bill &/or low energy quality	8.40%	6.43%
LIHEZLEK	LIHELEK-very	6.17%	3.43%

and/or low energy quality housing (LEK). As such, LIHELEK represents a multidimensional concept in this study. The areas representing LIHELEK are depicted as shaded regions in Figure 2.1.

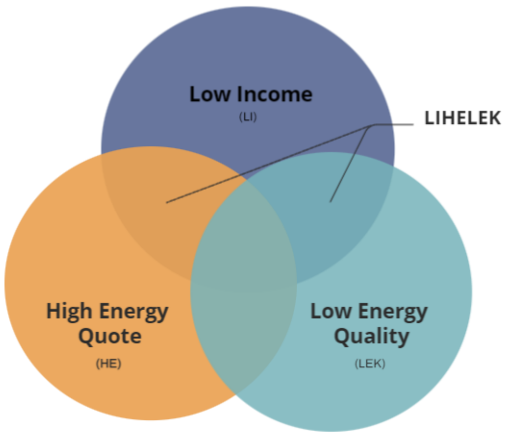


Figure 2.1: Venn diagram of Energy poverty character

2.2 Machine learning models

To select the most suitable machine learning model for the classification task in this study, various models are being tried. With regard to David H. Wolpert's "No free lunch theorem," [13] no single machine learning algorithm is universally superior for all possible problem domains, and its suitability largely depends on the problem domain and dataset. It means that there are limitations and trade-offs inherent in any learning algorithm. Therefore, to identify the appropriate model for predicting energy poverty in households, the performance of various learning algorithms is evaluated. The models differ in terms of learning time, computation cost, complexity, and the number of features.

This thesis employs two primary categories of machine learning models: tree-based models and linear models. The tree-based models consist of a decision tree, a random forest (an ensemble model), and Extreme Gradient Boosting (a boosting model). On the other hand, the linear models comprise logistic regression and a single-layer network (perceptron).

2.2.1 Decision Tree (DT)

The decision tree classifier is the first machine learning model, which is trained in this study to predict energy poverty. The decision tree model has a good predictive performance and interpretability for classification problems [14]. Decision trees mostly use for grouping data, and conceptually rules are easier to construct than weights and architecture in neural networks [15]. Each tree contains nodes and branches. Each node represents a feature which is supposed to be classified and each subset defines a value that can be taken by the node [15].

To use the DT, it starts at the tree root and splits the data on the feature which can result in the largest information gain (IG), then by repeating, this splitting procedure is repeated at each child node until the leaves are prune [3]. Prune means that at each node then, the training examples belong to the same class [3]. The objective function to optimize the tree learning algorithm defines as follows [3]:

$$IG(D_p, f) = I(D_p) - \sum_{j=1}^{m} \frac{N_j}{N_p} I(D_j) \tag{2.1}$$

f is the feature to perform the split, D_p and D_j are the dataset of the parent and jth child node, I is impurity measure, N_p is the total number of training examples at the parent node and N_j is the number of examples in the jth child node. Then, information gain is the difference between the impurity of the parent node and the sum of the child node impurities [3]. Therefore, in the scikit-learn library, binary decision tree is implemented in a simpler way. For simplicity, each parent node splits into two child nodes, D_{left} and D_{right} as follows [3]:

$$IG(D_p, f) = I(D_p) - \frac{N_{left}}{N_p} I(D_{right}) \tag{2.2}$$

The DT algorithm is used for supervised learning to build a model based on training data to predict target classes. As mentioned before, it is simple to comprehend and can classify both categorical and numerical outcomes [15].

On the other side, the optimal decision-making mechanism can be deterred and leads to incorrect decisions [15]. DT induction algorithms have several other advantages over many ML algorithms, such as robustness to noise, tolerance against missing information, handling of irrelevant and redundant predictive attribute values, and low computational cost [15].

Tuning hyperparameters

Since the hyperparameter tuning is computationally expensive, some of the hyperparameters which have the most effect on the model's performance are selected. Besides, Nowozin et al.[16] mentioned that the key parameters are the maximum depth of the tree, the minimum number of samples to keep growing, and the type and number of splits. Therefore, Maximum depth and to have more control of the model's performance, Minimum Samples Leaf and Criterion are tuned in this study as well. The selected hyperparameters for tuning include:

1. Maximum Depth: The longest path from the root of the tree to any of its leaves, and it is a good time measurement which is needed to have a classification [17].

2. Samples Leaves: The minimum number of instances in nodes/leaves used for splitting [14]. The number of leaf nodes in the tree is a complexity metric [17].

3. Criterion: Splitting criteria or impurity that are common in binary classification are Gini impurity (I_G) and entropy (I_H). The definition of entropy is following [3]:

$$I_H(t) = -\sum_{i=1}^{c} p(i|t) log_2 p(i|t) \tag{2.3}$$

$p(i|t)$ is the proportion of the example that belong to calss i for a specific node, t. The Gini impurity is a criterion to minimize the probability of misclassification as following [3]:

$$I_G(t) = \sum_{i=1}^{c} p(i|t)(1 - p(i|t)) = 1 - \sum_{i=1}^{c} p(i|t)^2 \tag{2.4}$$

Entropy and Gini impurity are maximal if the classes are perfectly mixed [3].

2.2.2 Extreme Gradient Boosting (XGBoost)

Boosting is a technique for reducing bias in model training, So when training for another model, it adjusts the weight on the error in the prior model [1]. As a result, it is capable of reducing bias in data training [1].

Extreme Gradient boosting (XGBoost) is in the category of ensemble learning models, which is used as a gradient boosting system [18]. Gradient boosting is the use of boosting to a decision tree in order to reduce bias in the model and hence increase accuracy [1]. Figure 2.2 shows how by increasing the iterations, the error of the gradient boosting algorithm is decreasing.

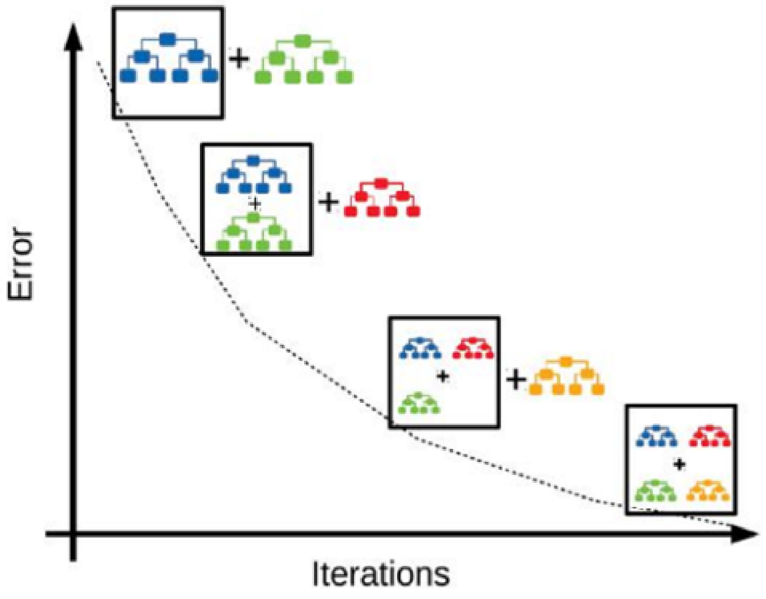


Figure 2.2: Gradient Boosting algorithm [1]

Regarding the algorithm, overfitting is possible to happen [1]. XGBoost consists of two regularizations which is based on GBDT, and make it a better algorithm to avoid overfitting [18].

XGBoost models have some advantages like high-computation speed, high prediction performance, and suitable robustness [18].

Tuning hyperparameters

The hyperparameters of XGBoost, which make it a promising model, are described as follows:

1. Number of estimator: The maximum number of weak learners [18]. This is a tricky hyperparameter. If a small value is chosen, underfitting will happen, and by selecting a large value, overfitting will happen [18]. For overfitting control, two approaches are used. First, the model complexity and adding randomness. To control the model complexity directly, *max_ depth* and *min_ child_ weight* will be tuned [19].

2. Maximum depth: The maximum depth of a tree [18].

3. Minimum child Weight: The minimum sum of the instance weight needed in a child [18]. To add randomness for making the training more robust to noise, the following hyperparameters tuned [19].

4. Colsample bylevel: The subsample of the columns at each level [18].

5. Subsample ratio: The subsample ratio of the training instance [18].

2.2.3 Random Forest (RF)

Bagging, which stands for Bootstrap Aggregating, is a model averaging approach aimed at reducing variance and improving the generalization performance of a predictive model [1]. It works by training different parts of the data to fit the model and then averaging the results of all the models to get the best result [1].

Random forest is a nonlinear classification method based on building an ensemble of decision trees and predicted output is the majority voting of the individual trees [20]. This approach combines them to achieve enhanced prediction accuracy and stability [21]. The process in the random forest model is shown in Figure 2.3.

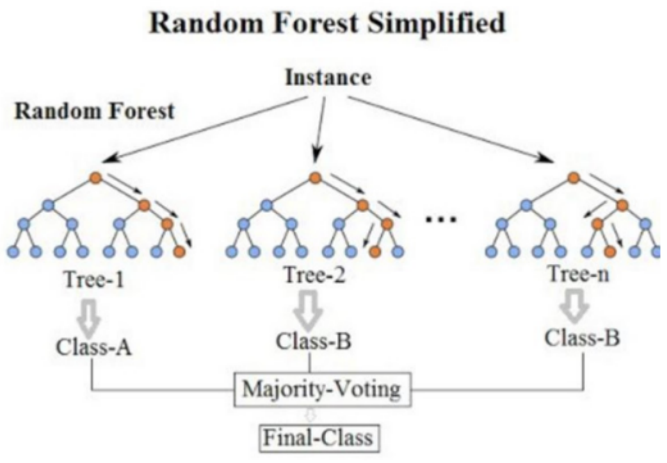


Figure 2.3: Random forest model [1]

The random forest algorithm (RF), originally introduced by Breiman in 2001, has become a widely used non-parametric method for classification and regression tasks [22]. It is capable of constructing prediction rules by considering a variety of predictor variables without any predetermined assumptions about their relationship with the response variable [22].

Tuning hyperparameters

The random forest model has several hyperparameters to control the architecture of each tree, the structure and size of the tree, and the level of randomness. Features that are tuned in this study are the following:

1. Max_ depth: The maximum depth of the tree [23].

2. Max_features: The number of features to consider when looking for the best split [23].

3. Criterion: The function to measure the quality of a split [23].

4. n_ estimators: The number of trees in a forest [23].

2.2.4 Logistic Regression (LR)

Logistic regression is like linear regression and it uses with binomial variable [24]. The Logistic Regression Classifier calculates a weighted sum of the input features and applies a logistic transformation to produce an output between 0 and 1 [21]. The logistic transformation is achieved using a sigmoid function [21].

A logistic regression model predicts the probability of an outcome based on individual attributes; besides, the chance is a ratio [24]. The logarithm of the possibility is [24]:

$$log(\frac{\pi}{1-\pi}) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_m X_m \tag{2.5}$$

Here, π is the probability of an event and β_i are the regression coefficients related to the reference cluster, and X_i are explanatory variables. Individuals in the reference cluster, denoted by β_0, present the reference level of each variable $X_{1..m}$ [24].

Tuning hyperparameters

The hyperparameters which are tuned in this study are:

1. Solvers: The algorithm to use in the optimization problem [23].

2. Penalty: Specify the norm of the penalty [23].

3. C: Inverse of regularization strength. Smaller values specify stronger regularization [23].

2.2.5 Single-layer network (Perceptron)

The perceptron is a linear supervised machine learning model for binary classification. It categorises input data into one of two separate states based on a training procedure carried out on prior input data. A perceptron is a single-layer neural network, which is shown in Fig. 2.4.

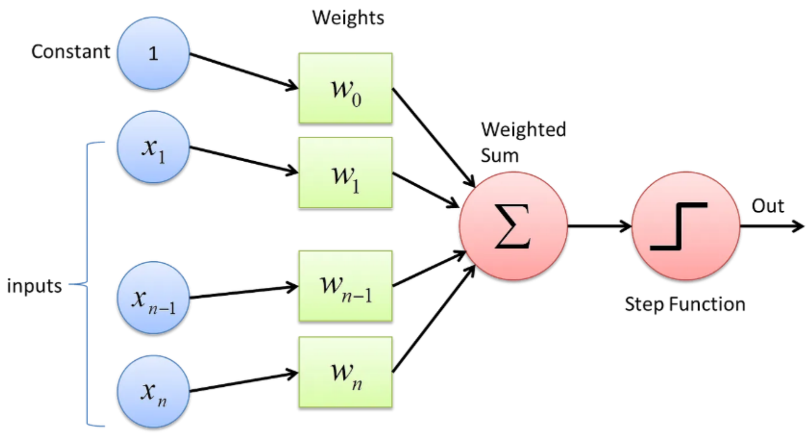


Figure 2.4: Perceptron architecture [2]

To make a prediction, the model calculates the weighted sum of the inputs and a bias (set to 1), which is known as activation. If the activation is greater than 0.0, the model outputs 1.0; otherwise, it outputs 0.0. This binary output makes the Perceptron suitable for two-class classification tasks[25].

$$Activation = (Weights * Inputs) + Bias \tag{2.6}$$

The Perceptron's key feature is its ability to learn a decision boundary that separates two classes using a straight line (hyperplane) in the feature space [25]. This decision boundary helps the model classify new data points into one of the two classes based on their feature values [25].

Tuning hyperparameters

Many hyperparameters could be optimized, but two hyperparameters which will probably have the most effect on the learning are:

1. Learning rate (eta0): The constant by which the updates are multiplied [23]. The learning rate is also critical for obtaining the optimum. A slow learning rate necessitates frequent updates, whereas a large learning rate results in divergent behavior [1].

2. Epoch (max_ iter): The maximum number of passes over the training data (aka epochs) [23].

2.2.6 Ensemble Modeling

Ensemble modeling is a powerful technique that involves creating multiple diverse models to predict an outcome. These models can be generated using different algorithms or by using different training datasets. The ensemble modeling combines the predictions of each base model to generate a final prediction for unseen data. The main purpose of using ensemble modeling is to reduce the generalization error of the predictions [26]. The effectiveness of ensemble modeling relies on the diversity and independence of the base models [26]. When the base models are diverse and independent, the ensemble approach helps decrease the prediction error, resulting in more accurate predictions [26]. It is well known that by combining classifiers, one can improve prediction accuracy. It is effective to embed the data level approaches in boosting procedure which is one of the most popular combination techniques [27]. It has proven to be successful in improving the accuracy and robustness of prediction models, making it a valuable tool in various applications [26].

In this thesis, ensemble modeling adopts the hard voting technique, where the final classification decision is determined by the majority consensus among individual models [21].

2.3 Feature Selection methods

This section will explain the method for the feature selection process to shorten the list of features.

Given the extensive size of the datasets, exceeding 6 million raw entries, the reduction of noise in the training data is crucial. Thus, in this study, feature selection was identified as a fundamental component of the entire pipeline. It is noteworthy that the results obtained from feature selection hold significant value for both the company and policymakers in subsequent stages.

In order to examine the significance of each feature in predicting whether a household is experiencing energy poverty or not, a comprehensive ranking of all the features is conducted. One commonly used feature selection and ranking method is regarding the variable importance. Feature importance is the term to show how important the feature is for the classification performance of the model [20]. It is used to measure each feature's contribution to the classifier model's performance, and it would be different for various machine learning models [20], [28]. The variable relevance is computed by calculating the incremental improvement in performance attributable to each application of a feature inside the model and aggregating this data over the whole model [28]. One advantage of this approach is its ability to capture the information of one feature if two features are highly correlated [3].

The relation for the importance of each node j in an individual decision tree is the following [29]:

$$ni_j = w_j C_j - w_{left(j)} C_{left(j)} - w_{right(j)} C_{right(j)} \tag{2.7}$$

Which ni_j is the importance of node j, w is the weighted number of samples reaching node j, C is the impurity value of node j, $left(j)$, $right(j)$ are child nodes from left and right split on node j, respectively. Therefore, the importance score of feature i is [29]:

$$fi_i = \frac{{}_{j:node\ j\ splits\ on\ feature} ni_j}{\sum_{j \in all\ nodes} ni_j} \tag{2.8}$$

For random forest and gradient boosting, the feature importance is the mean over all the trees [29]. In other words, by using a random forest, the feature importance is as the averaged impurity decrease computed from all decision trees in the forest without making any assumptions about whether our data is linearly separable or not[3].

To assess the impact of incorporating features based on their rankings, each feature can be incrementally added to the model. Subsequently, the model is trained using the training set, and its performance in predicting the validation set is measured and plotted with the cumulative inclusion of features. This analysis aims to evaluate the influence of feature addition on the predictive capabilities of the model.

As it will be shown in the chapter4, after reaching its peak, the performance of the models tends to plateau or exhibit minimal change when additional features are added. This observation suggests that further feature additions do not lead to significant improvements. However, it is worth noting that while the inclusion of certain variables may not immediately impact performance, the addition of another feature in subsequent steps can alter the behaviour of the performance line. Consequently, relying solely on feature ranking based on importance scores is insufficient for the feature selection process, and further investigation is needed.

To identify the most relevant features, two methods are used, and their performance on the validation set is evaluated. The model that demonstrates the best performance among these methods is selected as the preferred choice. The machine learning model for the evaluation of the methods is the decision tree, as it is considerably computationally fast.

There are three feature selection (FS) approaches: filter, wrapper and embedded [30].

I. Select K-Best. This method belongs to the filter category of feature selection methods, and it uses features independently of the classifier without involving the learning algorithm [30]. The filter methods consist of univariate and multivariate methods [30]. This method is one of the univariate feature selections, which is based on univariate statistical tests. SelectKBest keeps the k highest scoring features and removes all others [23]. The univariate algorithms' features are examined one by one, regardless of their influence on the other features [30].

In the filter approach, each feature has its sore after evaluating its performance [30]. Considering the numerical and categorical features in this study, different evaluation methods are used. ANOVA F-value is employed for the numerical features, while the chi-squared statistic is used for the categorical variables.

II. Recursive Feature Elimination (RFE): RFE is a wrapper feature selection method which depends on the used classifier, which means that finding the relevant features is based on the learning algorithm [30]. In theory, this approach outperforms the previous approach [30], which is examined in this study. RFE is a feature selection method that is based on the idea of iteratively removing features that are not important for the classification task. In brief, RFE starts with all of the features and then, at each step, removes the feature that has the least impact on the classification performance. This process is repeated until a desired number of features have been selected. However, RFE regarding the complexity of the models is time-consuming [30].

3

Literature Review

Due to the complex nature of energy poverty, studies have utilized various indicators to assess, comprehend, and monitor this phenomenon. Recognizing its multidimensional nature, a comprehensive set of indicators has been employed to capture its social, economic, and technical dimensions effectively. This thesis specifically focuses on related works that had the Netherlands as geographical scope and studies that have used machine ML models to analyze energy poverty. This research can take advantage of insights and results that are directly applicable to the local context by focusing specifically on the Netherlands and by utilizing the strength of machine learning techniques for analysis and prediction.

3.1 Energy Poverty in the Netherlands

In 2021, Dalla Longa et al., [8] presented an approach that uses machine learning techniques to estimate the probability of energy poverty. The energy poverty metric that they used was the "Low Income, High-Cost". The authors propose implementing a machine learning classifier to forecast the possibility of energy poverty based on a variety of socioeconomic indicators such as housing value, ownership and age, household size, and average population density. While income remains the most important predictor, including these extra socioeconomic characteristics is critical for making accurate predictions. The study utilized two datasets: one based on neighborhood-level average data, the other based on single household data but covering only about 1% of Dutch households. The authors focus on training gradient boosting decision tree models using XGBoost.

The research shows the performance of three XGBoost models trained on KWB data [31]. Model A just includes income as an input feature, Model B includes the five characteristics stated in Section 3.1, and Model C combines income and these five factors. The probability of energy poverty can be predicted using the XGBoost algorithm, which is a gradient boosting decision-tree approach, based on specified socioeconomic features accessible at either the neighborhood-level averages or the individual family level. The authors provide insights into the factors contributing to energy poverty and emphasize the importance of these features for accurate risk assessment.

In 2023, Mulder et al. [5] provided an innovative approach to characterize energy poverty in the Netherlands from a multifaceted and spatial perspective. The goal is to create a nationwide energy poverty monitor based on comprehensive spatial analysis. The study includes georeferenced microdata at the household level, which covers about 80% of Dutch households. The authors propose a series of novel indicators that highlight three aspects of the energy poverty problem: energy affordability, housing quality in terms of energy efficiency, and households' ability to participate in the energy transition. Taking these factors into account, the authors discover that around 7% of Dutch homes suffer a combination of high energy costs, inadequate insulation, and low income as of 2019 energy prices. They mentioned that, unlike many other North-West European countries, the Netherlands has historically ignored the issue of energy poverty in its national policy. As a result, there has been no regular national monitoring of energy poverty and high energy bills have traditionally been handled in the context of income poverty and income programs. According to Mulder et al. [5], while the number of households currently affected by high energy costs is relatively small, energy poverty has the potential to increase significantly due to factors such as rising energy prices and growing demand for energy-efficient homes in the context of the energy transition. The report emphasizes the importance of treating energy poverty and putting it into national policy frameworks in order to reduce its impact.

In a follow-up study published in 2023 [11], Mulder and coauthors provided an updated estimate of the national and local levels of energy poverty in the Netherlands. Following [5], CBS has established a national EP monitor at the request of the Ministry of Economic Affairs, and in [11], the authors analyzed the data in the monitor. The study addresses four major issues: the number of energy-poor households, their energy costs, the characteristics of energy-poor households, and the geographic distribution of houses with low-energy efficiency. The estimates imply a 90,000 rise in energy-poor homes between 2020 and 2022, totaling 602,000 households (7.4% of the total). The authors showed evidence that the utilization of compensation measures has prevented a significant rise in energy poverty levels, even with the increase in energy prices. Energy savings also help to reduce energy poverty slightly. The study found that families with the lowest energy quality, primarily energy labels G and F, have the highest prevalence of energy poverty.

houses with low-energy efficiency are disproportionately composed of single-person households and single-parent families. The study finds a rise in energy poverty in various urban regions across the country, as well as an increase in energy-poor households in homes with poor energy quality.

In summary, existing research in the Netherlands has explored energy poverty using machine learning approaches, but the data used in these studies has been limited. Additionally, there is a lack of machine learning studies that focus on large-scale household data. Therefore, there is a clear need to further investigate energy poverty using machine learning methods on more extensive datasets of households.

3.2 Machine learning studies

In 2020, Rajic et al. [32] introduced a new application of neural networks in energy systems and energy resource planning activities. The primary objective of the study is to analyze energy poverty using real socio-economic data within a specific country. The model employed in the research consists of a neural network with one hidden layer. The data used in this analysis are for the Republic of Serbia and consist of monthly level data for the last 27 years. The dataset consists of 15 features. The authors provided a global classification of energy poverty indicators, encompassing 178 different indicators. These indicators are classified into several categories, including income/expenditure (33%), physical infrastructure (29%), policy-based (12%), outcomes (12%), demographics (8%), and energy demand (6%). Additionally, the report introduces expenditure-based indicators for the energy poverty approach, which can be classified into three types: a high share of energy costs, low available income, and low energy efficiency of households. Overall, the study offers a new approach to analyzing energy poverty by neural networks and real socio-economic data, with a focus on the specific country under examination.

In 2021, the study conducted by Hong et al. [33] aims to develop a series of models for predicting energy poverty and analyze the relative importance and partial dependencies of indicators by using machine learning techniques. Their data is from the Korea 2016 Household Income and Expenditure Survey conducted by the National Statistics Office. The machine learning models in the study are Decision tree, Artificial neural network, Bagging, Random Forest, XGBoost, and SVM. The major factors contributing to energy poverty identified in the study include household characteristics, the reference person of household characteristics, consumption characteristics, and residential characteristics. The results indicate that the Random Forest model outperforms other models. The indicator used for energy poverty is CEPI, defined as having an income below 60% of the median income and energy expenditure exceeding the household income median by more than 10%. It can reflect both the low-income level and the low fuel expense of energy-poor households.

In 2022, Abbas et al. [34] calculated the depth, intensity, and degrees of energy poverty in developing countries using a multidimensional approach. It also employed machine learning algorithms to identify the most pertinent socioeconomic determinants of extreme multidimensional energy poverty. These findings have policy significance in eradicating severe energy poverty. They used survey data from 59 countries in Asia and Africa with socio-economic variables, which are the accumulated wealth of a household, size, and ownership status of a house, marital status, and place of residence. This study employs Multidimensional Energy Poverty Index (MEPI) to calculate the depth and degrees of energy poverty across multiple dimensions of household energy services. For the machine learning model, they implemented a Multilayer Perceptron Artificial Neural Network model with two hidden layers.

In their research, in 2022, van Hove et al. [7] employed machine learning techniques to identify the underlying factors driving energy poverty in Europe. Based on a household-level survey in 11 European countries with various economies, cultures, and climates, they used a "low income, high expenditure" framework to classify households as energy poor. The authors get successful results by training a gradient-boosting classifier on a collection of socio-economic features which are thought to be predictive of energy poverty. The energy poverty classification methodology classified each household into four risk categories. This paradigm is based on the income vs. energy expenditure grid, which is divided into four quadrants using two thresholds. In their study, the energy poverty risk classifier identifies income, floor space, and household size as highly critical factors. These three characteristics are thought to have universal predictive potential across the European continent.

In 2022, Lopez-Vargas et al. [35] did a thorough evaluation of the literature on the use of Artificial Intelligence (AI) in the context of Energy Poverty. The research looked into the methodology, data sources, and applications of AI in tackling multidimensional energy poverty; low income, high energy prices, and low building energy efficiency. The review focused on publications over the previous seven years. The review's findings revealed that Artificial Neural Networks (ANNs) and Decision Trees were the most used AI algorithms in the field of energy poverty. The AI algorithms used works were categorized into two main groups: those utilizing ANNs-based algorithms and those employing Decision Trees. Both ANNs-based approaches, including deep learning techniques, and regression algorithms were commonly utilized in the categorization of low-income people using AI. Furthermore, Random Forest (RF), Support Vector Machines (SVMs), and Gradient Boosting Machines were preferred algorithms in low-income investigations. ANN-based algorithms were the most widely utilized AI technique for energy cost analysis, while SVM-based algorithms were commonly used for energy consumption-related problems. Deep learning approaches were found to be effective at detecting energy billing discrepancies and unpaid energy bills. ANN-based algorithms were the most commonly used AI tool for investigations involving poor energy efficiency, followed by RF and deep learning methods. The review, however, indicated a gap in the literature, as there were few studies focusing on the application of AI. They highlighted the need for further exploration of AI approaches to comprehensively address the multidimensional nature of energy poverty.

3.3 Differences from Previous Studies: Addressing Big Data Challenges in Energy Poverty Prediction

In contrast to previous studies on energy poverty in the Netherlands, this thesis aims to address several key differences and gaps. Firstly, the challenge of dealing with big data at the household level is a significant focus of this research. By utilizing machine learning methods on extensive datasets, this study aims to overcome this challenge and provide valuable insights into energy poverty prediction.

Secondly, while previous studies have explored machine learning approaches for energy poverty prediction, they have not specifically focused on identifying the most important features for prediction. This thesis aims to fill this gap by using various machine learning models and determining the common features with the highest predictive power. This information can be crucial for policymakers to make informed decisions and gain a comprehensive understanding of the contributing factors to energy poverty.

Finally, the investigation of different machine learning methods and their impacts on energy poverty prediction is a central aspect of this study. By exploring various algorithms, this research seeks to identify the most effective methods for predicting energy poverty as a multidimensional concept, known as LIHELEK.

In conclusion, this study's main focus is on investigating the predictive power of machine learning methods for energy poverty prediction, addressing the challenges of big data, identifying important features, and exploring different machine learning approaches. By addressing these gaps, this thesis aims to contribute valuable insights and support policymakers in tackling the issue of energy poverty effectively.

4

Methodology

In this chapter, we begin by providing an overview and understanding of the dataset that was used for our study. We then outline the step-by-step process that was undertaken to address the research questions stated in chapter 1.

4.1 Dataset

This thesis uses various datasets to make predictions for energy poverty in the Netherlands. Based on the previous studies on energy poverty in TNO [5], [9], [11], CBS has developed the indicators and has provided datasets on household level for energy poverty, Microdata Monitor Energy Poverty [10]. These datasets provide a complete profile of households, their residential, and geographical situation, and the household reference person (which we will refer to as "householder" in the remainder of this thesis). Therefore, the main datasets are CBS Microdata for energy poverty in 2019 and 2020, published in February 2023. These anonymized datasets are accessible through the CBS Microdata services [10], and contain 6,963,830 and 7,037,415 records, respectively, of households living in the Netherlands.

In order to have more insights, datasets on additional socioeconomic and geographical features were merged with the energy poverty tables. Table 4.1 mentions all the used datasets and their source in the CBS database.

Table 4.1: Datasets

Datasets	**Features**	**Source**
CBS Microdata	Household and accommodation features	Energiemoede (2019, 2020) [10]
Household reference person	Demographic features	GBAPERSOON (2019, 2020) [36]

Table 4.1: Datasets

Datasets	**Features**	**Source**
Accommodation	Geographical features	kwb 2022 [31]

All the features are categorized into four main categories:

- Household features
- Household reference person features
- Accommodation features
- Geographical features

Table 4.2 shows all the features, their definitions, and their related category.

Table 4.2: All the features

Category	**Feature Name**	**Definintion**
EP Indicator	LIHELEK	Households with a low income and a high energy bill and/or low energy quality home
Household Features	Starting Year	Staring year of household composition
	Type	Type of household
	Financial assets	Total value of a household's financial assets
	Mortgage Debt	Mortgage debt related to a household's owner-occupied home
	Size	Number of persons that from the household
	Population	Population delimitation of households with observed income in the dwelling (housing base)
	Income source	Main source of household income
	Disposable income	Household disposable income
	Income percentage	Income relative to the low-income threshold
	Standardized income	the disposable income of a household corrected for the size and composition of a household
	Payment budget	The disposable income of a household, excluding the included expenditure components

Table 4.2: All the features

Category	**Feature Name**	**Definintion**
	Payment Reserve	Value of the total assets of a household, excluding the value of the owner-occupied home and negatively valued components in the survey year
	Gas Usage	Gas consumption in m3 in the year under review
	Electricity usage	Electricity consumption in kWh in the year under review
	Solar panel usage	The volume of the feed-in of electricity by solar panels.
	City heat usage	Estimated heat consumption in city in the year under review
	Allowance	Indicator of whether a household is entitled to an energy allowance
	Energy bill	Energy amount in the year under review (euro)
Household Reference Person Features	Age	Age of the reference person of the household (at the beginning of the year)
	Date of birth	Date of birth of the reference person of the household
	Education	Education of the reference person of the household
	Gender	Gender of the reference person of the household
	Migration background	Migration background of the reference person of the household
Accommodation Features	House value	Value of a house owned by a household
	Residential type	Type of the accommodation with residential function
	Construction year	Year of the building constructed
	Surface area	Usable area of the accommodation in square metres
	Residential type	Permitted functional use of the accommodation for living
	Value	Value of immovable property by a municipality in term of real states
	Ownership	Ownership status of a house
	Energy label	Registered energy label
Geographical Features	Municipality code	Municipality code of the accommodation

Table 4.2: All the features

Category	Feature Name	Definintion
	District code	District code of the accommodation
	Neighbourhood code	Neighbourhood code of the accommodation

More details about the variables, their description, and their names in the CBS documents are available in Appendix A.

In this study, the primary objective is to predict energy poverty using variables that are not directly related to energy poverty indicators and are not utilized in calculating household energy poverty.

4.2 Data Prepration

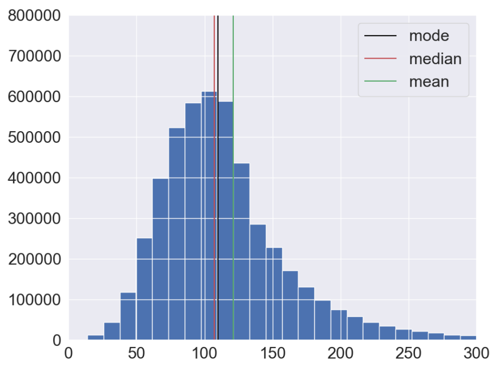
4.2.1 Handling missing values

It is important to address missing values in the dataset, as many machine learning algorithms struggle to handle them. For optimal performance, appropriate processing techniques need to be applied. These techniques aim to handle missing values in a way that allows the machine learning algorithms to effectively analyze the data. Fortunately, the amount of missing values in the CBS dataset was insignificant and could be handled with imputation.

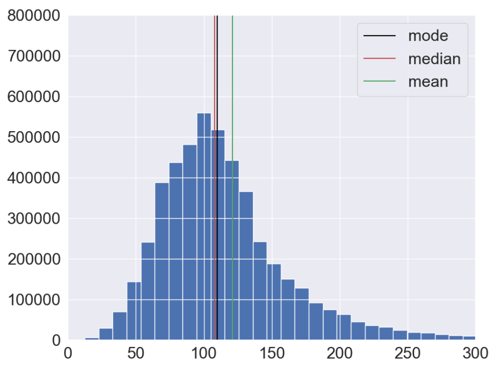
For numeric features with missing values, such as "accommodation construction year" and "accommodation area," the missing values are imputed by replacing them with the mean of their respective columns. The feature "accommodation construction year" contains missing values, which need to be addressed. In this study, the missing values are imputed with the mean value of the feature.

In the case of the "accommodation area" feature, the median value is utilized for imputation due to the presence of a long tail in the distribution. Using the median as an indicator for the central value of the data is considered more appropriate in such cases. Figure 4.1 illustrates the distribution of the accommodation area in the years 2019 and 2020, the y-axis shows the number of households. It demonstrates that the median and mean values are close, but the median value is slightly lower.

The features "city heat" and "solar panel" have missing values, respectively. The missing values for these features mean that the corresponding household does not use these energy sources. So, the missing values are replaced with 0. The "education level of householder"



(a) 2019



(b) 2020

Figure 4.1: Accommodation area distribution with mean and median

feature is excluded from the analysis due to a significant amount of missing data exceeding half of the feature's values.

It is essential to highlight that when handling missing values in the dataset, techniques are applied separately for the training and test sets. This approach is crucial to prevent data leakage. If we were to fill in missing values using information from both the training and test sets combined, it could lead to data leakage, as the model would inadvertently learn from the test set. By handling missing values independently for each set, we ensure the integrity of the model's performance on unseen data during testing.

4.2.2 Feature Engineering

To enhance the analysis and make certain features more suitable for further investigation, feature engineering steps are undertaken. These steps involve manipulating and transforming the existing features to derive new features or modify existing ones.

In the initial step of feature engineering, the categorical feature "accommodation energy label" is transformed into a numerical format. The mapping assigns numerical values to the labels A to G, with "unknown" being mapped to 1 and subsequent labels incrementing from there.

Additionally, a new binary feature, "high energy label," is created based on the energy label. If the energy label is greater than 3, indicating a higher energy label category, the value of the new feature is set to true.

Furthermore, two more binary features, "accommodation solar panel" and "accommodation city heat," are introduced. These features indicate whether households utilize solar panels or city heat sources of energy in their accommodation. These binary features are added to provide additional information about the energy sources utilized by households.

Another binary feature is "accommodation urban", which indicates if the accommodation is located in an urban area or not. If the "urban level" feature is equal to 1 or 2, the "accommodation urban" is true.

The features listed in Table 4.3 are ultimately used for the feature selection and training pipelines. These features are categorized into numerical and categorical variables, and their corresponding names in the dataset are provided.

Table 4.3: Features used in the pipelines

Type	Category - Feature Name	Feature Code
	Household - financial assets	hh_assets
	Household - Mortgage debt	hh_mortgageDebt
	Householder age	ref_age
	Household - size	hh_size
Numerical	Accommodation - construction year	acc_constructionYear
	Accommodation - surface area	acc_area
	Accommodation - value	acc_value
	Household - energy bill	energy_bill
	Household - Starting year	move_date_y
	Household - type	hh_type
	Household - population	hh_population
	Household - income source	hh_incomeSource
	Accommodation - residential type	residential_type
	Householder - gender	Gender
	Householder - migration background	migration_status
Categorical	Accommodation - energy label	acc_energyLabel_encoded
	Accommodation - ownership	acc_ownershipType
	Accommodation solar panel	acc_solarPanel
	Accommodation - city heat	acc_Heat
	Accommodation - urbanization level	acc_urban_level
	Accommodation - urban area	acc_urban
	Accommodation - high energy label	acc_high_energyLabel

4.3 Binary Classification

In machine learning, binary classification is a supervised learning problem in which the goal is to classify instances into one of two classes. The two classes are "positive" and "negative," "1" and "0", here "energy poor" and "not energy poor" households. The aim of binary classification in this study is to create a model that can predict the class label of previously unseen cases based on their input features.

The input data in binary classification here consists of a collection of household, householder and accommodation features (independent variables or predictors) that describe each household. These characteristics, as mentioned in previous sections, are numerical and categorical. The class or category to which the instance belongs is represented by the target variable (dependent variable or class label).

The machine learning model is trained on a labeled dataset with known class labels. During the training process, the model learns patterns and correlations in the input features in order to predict the class labels of previously unknown instances.

Once the model is trained, it can be used to categorize new instances by allocating them to one of two classes. The output of the model is the label of input. Then, the performance of the model in comparison to the true label is measured.

4.3.1 Dealing with class imbalance

In our dataset, the majority of households (more than 90%) belong to the class with label 0, i.e. they are not energy-poor. It means that we could achieve more than 90% accuracy on the test dataset by using the majority class for prediction without the help of a supervised machine learning algorithm. Thus, training a model that can achieve approximately 90% test accuracy for class 0 would not indicate that the models learned a useful algorithm. Therefore, it is crucial in this study to emphasise the performance of class 1 (energy-poor households), which is the research's main goal.

Besides the evaluation of the models, the class imbalance can affect the learning algorithm during the training. Since machine learning algorithms often optimize a reward or loss function as a sum of the training instances seen during fitting, the decision rule is likely to be biased toward the majority class [3]. It means that the model tries to optimize the algorithm based on the majority class to maximize the performance.

There are mainly three solutions for this problem, which are [3]:

1. Generate new training data
2. Upsampling, downsampling for the minority and majority class, respectively.
3. Penalize approach: Assign a larger penalty to the wrong prediction on the class which has a minority.

Given the time and scope of this project, the first approach is not viable. The second option, despite its potential usefulness, is not used due the unavailability of the necessary software packages in the CBS Microdata environment. In the next steps of the whole pipeline, the third method, adding penalizing to the training model, will be taken.

In this study, we use the term "class weighting" to refer to the penalizing method. This approach involves considering the class proportion or weight. Class proportion assigns a substantial penalty to incorrect predictions on the minority class during the model fitting process [3]. The primary objective of class weighting is to give more significant weight to errors made on the minority class, thus preventing the model from achieving high accuracy solely by focusing on the majority class. By assigning higher weights to the minority class, class weighting helps to balance the impact of different classes and encourages the model to make accurate predictions for both the majority and minority classes.

The class weight implementation involves adding it as a hyperparameter to the model. For all the models, except the XGBoost model, adding class weights means that the weight assigned to each class is inversely proportional to its frequency in the dataset. [23]:

$$w_j = \frac{N_{samples}}{(N_{classes} * N_{samples,j})} \tag{4.1}$$

Here, w_j is the weight of class j, $N_{samples}$ is the total number of samples, $N_{classes}$ is the total number of unique classes, and $N_{samples,j}$ is the total number of the class j samples.

For the XGBoost model, we tackle the imbalanced data issue by adding the hyperparameter `scale_pos_weight`. This parameter is a common solution for handling unbalanced classes and is set to a value higher than the default (default = 1). By adjusting `scale_pos_weight`, we aim to assign greater importance to the positive class, thereby improving the model's performance on imbalanced datasets. The `scale_pos_weight` hyperparameter controls the balance between positive and negative weights in the model. A typical value to consider is $\frac{Sum(Negative\ instances)}{Sum(Positive\ instances)}$.

4.4 Training, validation and test data split

After analyzing the datasets, they were divided into training, validation, and test sets for further model training and evaluation.

In the same-year predictions, the data was split into three sets with the following ratios: 70% for the training set, 15% for the validation set, and 15% for the test set. This division ensures that the models can learn from a large portion of the data during training while also being able to assess their performance on unseen data using the validation and test sets.

For the next-year 2020 prediction, the entire dataset of 2019 was used as the training set. Then, the dataset of 2020 was split into two sets: a 50% validation set and a 50% test set. This division allows for model evaluation on data from the subsequent year while maintaining a balanced representation of the validation and test sets.

By dividing the data into various sets, the models may be trained on a subset of the data, validated on another subset, and finally evaluated on unseen data to determine their generalization performance.

The dataset sizes for each prediction are provided in Table 4.4.

Table 4.4: Training and test datasets

Prediction	**Dataset**	**Size**	**Not energy poor**	**Energy poor**
Same-year 2019	Training (2019)	4,874,681	0.916	0.084
	Test (2019)	1,044,574	0.916	0.083
Same-year 2020	Training (2020)	4,926,190	0.936	0.064
	Test (2020)	1,055,612	0.935	0.064
Next-year 2020	Training (2019)	6,963,830	0.916	0.084
	Test (2020)	2,111,224	0.936	0.064

4.5 Data preprocessing

4.5.1 Data Scaling

In order to ensure the effective use of the logistic regression model, it is essential to scale the numerical data appropriately. For this purpose, the normalization strategy is applied in this study. This strategy involves linearly transforming the data using the equation 4.2. The formula utilized for min-max normalization in this thesis is as follows [37]:

$$y = \frac{x - min(x)}{max(x) - min(x)} \tag{4.2}$$

Where x is an original value, y is the normalized value. This normalization process maps the data to a range between 0 and 1, which helps the machine learning model to identify clearer trends in the data and normalize the influence of different parameters [21]. By applying Min-Max normalization, the impact of various parameters can be more effectively taken into account by the logistic regression model.

For the single-layer network (perceptron) model, the data is standardized using a technique called Standardization or Z-score Normalization. This process rescales the distribution of values so that the mean of the data becomes 0 and the standard deviation becomes 1. Standardization is particularly useful when the input features have different scales or units. The data is standardized using the following formula [37]:

$$y = \frac{x - \bar{x}}{\sigma} \tag{4.3}$$

Where x is an original feature value, y is the standardized value, $\bar{x}$ is the mean of x, and σ is its standard deviation.

Researches suggest that normalizing representations of neural networks can significantly improve convergence rates in feed-forward neural networks [38]. These processes ensure that all input features have a similar scale and help the neural network model to learn effectively from the data.

4.6 Models feature selection

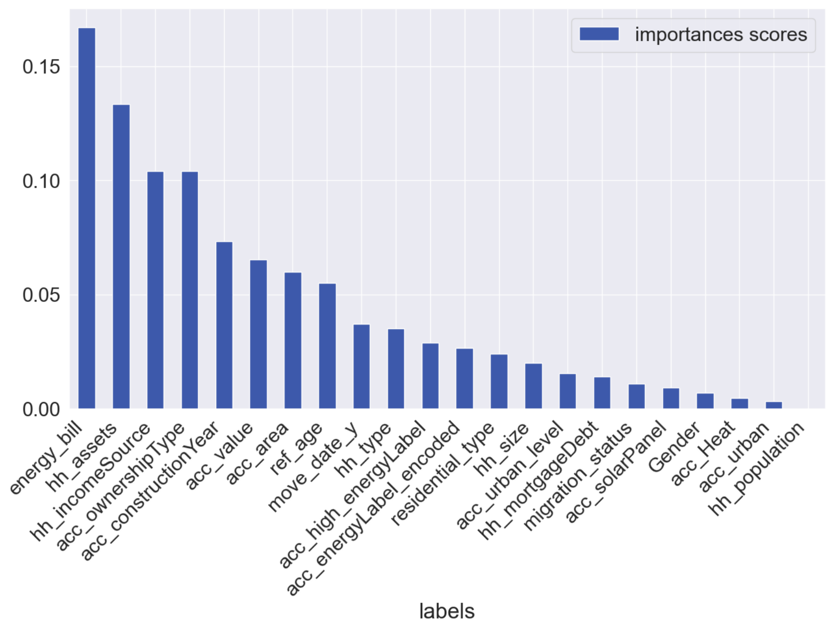
In Scikit-learn [23], feature importance scores are accessible via the `feature_importance_` attribute after fitting the model. After execution of the code, the rank of features based on their relative importance will be ready. The sum of feature importance scores is 1.0 since they are normalized [3].

Figure 4.2 shows the importance score of the features for the Decision Tree and XGBoost model. As it is evident, the importance score of each feature varies across different machine learning models due to their distinct learning algorithms.

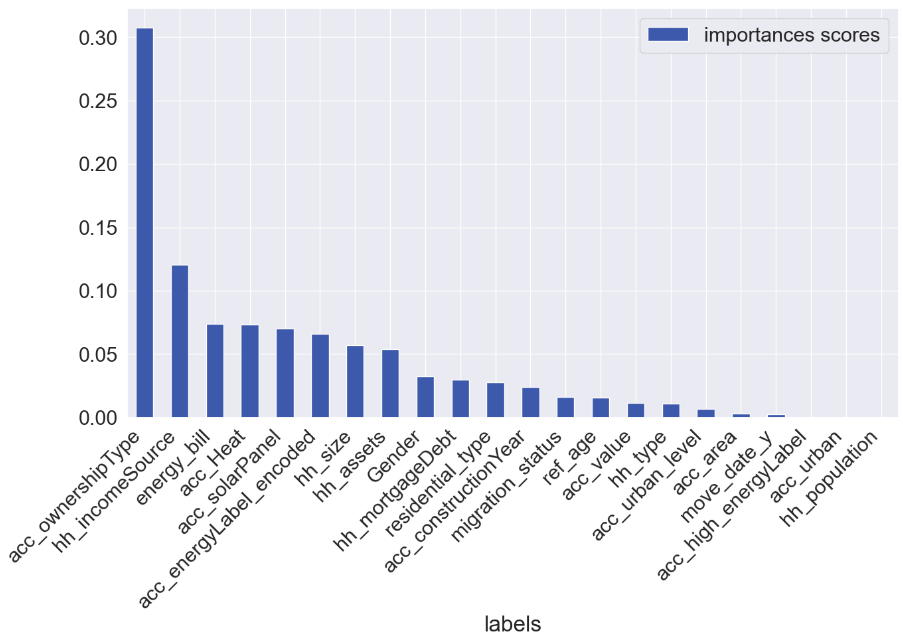
Figure 4.3 demonstrates the number of features and the performance of the model by having those number of features on the validation set. For this part, as mentioned before, regarding the class imbalance issue, the F1-score of the energy-poor household class (class 1) is considered.

To compare the performance of the selected features from the Recursive Feature Elimination (RFE) method and the SelectKBest method, both methods were tuned to identify the optimal number of features. Subsequently, each machine learning model was evaluated separately using the RFE and SelectKBest method. For both approaches, different sets of features were assessed and compared based on their f1-score for class 1.

To conduct the comparison, both methods were applied using the 2019 training set and evaluated on the corresponding validation set. The Decision Tree model was selected for



(a) Importance score DT

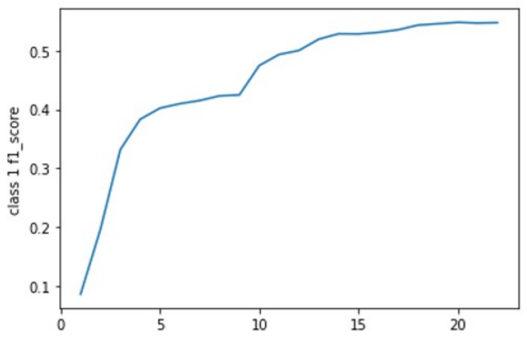


(b) Importance score XGBoost

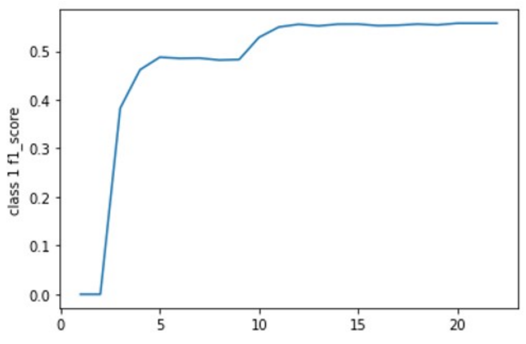
Figure 4.2: Feature importance score, DT and XGBoost

this analysis due to its reasonable running time, allowing for the entire dataset to be processed and the methods to be effectively tested.

In order to apply the Select K-Best method, the initial step involves testing various values



(a) Features performance DT



(b) Features performance XGBoost

Figure 4.3: Performance of number of features, DT and XGBoost

of k to determine the most suitable one. The selection is based on the performance of the f1-score for class 1. To assess the values of k, the k-fold cross-validation method is employed, utilizing the following setup: `RepeatedStratifiedKFold(n_splits=5, n_repeats=3)`.

Once the optimal value of k is determined for both numerical and categorical features, the top k features are extracted using the `SelectKBest()` method. These top features are then used to train a Decision tree model, which is subsequently evaluated on the validation set to assess its performance. The range of k features for numerical and categorical features is $(2, 6)$ and $(2, 10)$, respectively. Then, based on the best number of features for each category of features, the KBest features are determined accordingly.

Table 4.5 presents the results of the Select K-Best method for feature selection based on the 2019 training set with the decision tree model. The feature selection process is conducted separately for numerical and categorical features, and the table 4.5 displays the selected number of features and the corresponding feature names for each category.

Table 4.5: SelectKBest method features

	Numerical Features	Categorical Features
Running time	40min 39s	26min 42s
#features	3	4
Features	hh_mortgageDebt, hh_size, acc_value	hh_type, hh_incomeSource acc_energyLabel_encoded, acc_ownershipType

To determine the optimal number of features for the Recursive Feature Elimination (RFE) method, a comprehensive analysis was conducted using 5-fold cross-validation. The process was repeated three times to ensure reliability, resulting in a total of 15 experiments for each value in the number of features set.

Figure 4.4 displays boxplots illustrating the distribution of mean f1-scores for class 1 obtained from the RFE method with the Decision Tree model and the 2019 training set. As mentioned, each number of features is associated with 15 experiments. The x-axis represents the number of features, while the y-axis represents the mean f1-score. The entire process took approximately 24 hours and 10 minutes to complete.

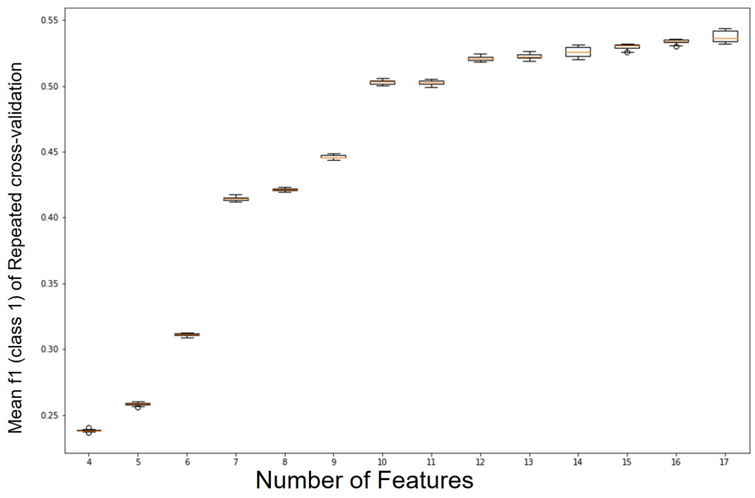


Figure 4.4: Mean of repeated cross-validation boxplots of features number, DT model

From the analysis of Figure 4.4, it can be observed that after incorporating 10 features, the model's performance does not exhibit significant changes. Therefore, in order to maintain a less noisy model, selecting $k = 10$ as the number of features from the RFE method is deemed appropriate. Table 4.6 shows all the features and their ranking from RFE for 2019 DT model.

Table 4.6: Features ranking, DT model, the full training dataset

Features	Ranking
hh_assets	1
ref_age	1
hh_size	1
acc_constructionYear	1
acc_area	1
acc_value	1
energy_bill	1
hh_incomeSource	1

Table 4.6: Features ranking, DT model, the full training dataset

Features	**Ranking**
acc_energyLabel_encoded	1
acc_ownershipType	1
move_date_y	2
residential_type	3
acc_urban_level	4
hh_mortgageDebt	5
hh_type	6
acc_high_energyLabel	7
migration_status	8
acc_solarPanel	9
Gender	10
acc_Heat	11
acc_urban	12
hh_population	13

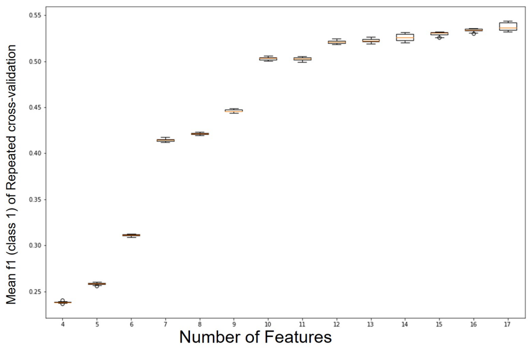
In order to conclude the feature selection (FS) process, the features selected by both the Select K-Best and RFE methods are used separately, and used as input for a Decision tree model. The performance of these feature sets is evaluated on the validation set, and the results are presented in Table 4.7. Although the Select K-Best method offers significantly faster feature selection compared to RFE, the difference in performance for class 1 is not negligible. Therefore, RFE is chosen as the preferred feature selection method for all machine learning models.

Table 4.7: Compare feature selection methods

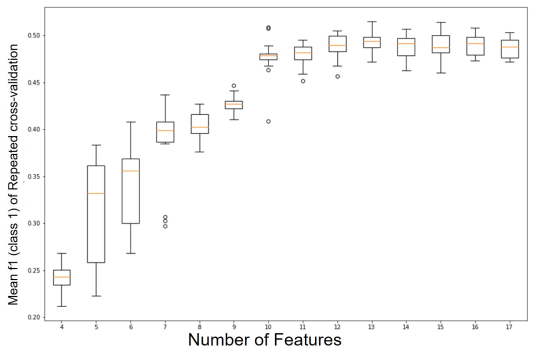
		Precision	**Recall**	**F1-score**
RFE	class 1	0.49	0.52	0.50
	macro avg	0.72	0.73	0.73
SelectKBest	class 1	0.49	0.23	0.32
	macro avg	0.71	0.23	0.32

Figure 4.5 illustrates the distribution of means of experiments for each number of features using the Decision Tree (DT) model, similar to the previous analysis conducted on the full dataset. However, this time the analysis is performed on subsets of the training data, specifically 100% and 1% of the original dataset.

By comparing panels (a) and (b) in Figure 4.5, it can be observed that the number of suitable features remains consistent across different dataset sizes. The size of the dataset mainly affects the variance in the means of experiments, while the overall trend remains



(a) 100% training dataset



(b) 1% training dataset

Figure 4.5: Mean of repeated cross-validation boxplots of features number for the 2019, DT model, using the full training set (a) and 1% of it (b)

similar. As indicated in both figures, $k = 10$ features appear to be appropriate for the DT model.

In addition, the running time for the Decision Tree (DT) model on the full training set was 24 hours, 10 minutes, and 42 seconds. According to the complexity of the RFE method, which is $O(\max(n, m)n^2)$ [39], determining the suitable number of features and ranking them is performed on a reduced dataset containing 1% of the training data. This exploration to identify the appropriate number of features is repeated for each machine learning algorithm individually.

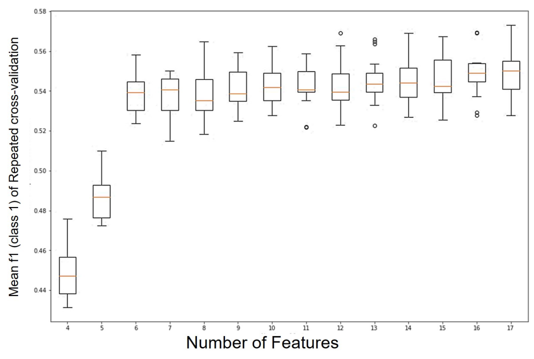
Figure 4.6 presents the boxplots displaying the means of each feature number, aiming to find the appropriate k for the RFE method. Based on the obtained k, the features are ranked for each model and selected for subsequent steps in the pipeline.

The ranking of the features is displayed completely for each ML algorithm in the appendix A, and the number and the name of features are shown in the table. 4.8.

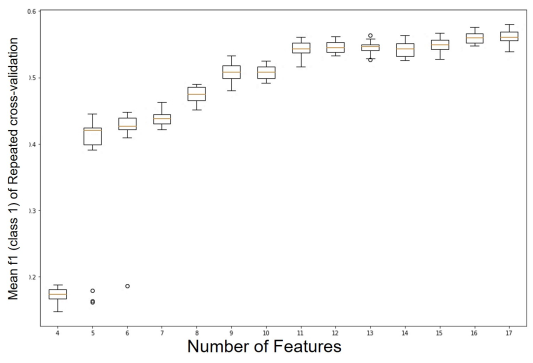
4.7 Hyperparameter Tuning

One of the common and simplest methods for hyperparameter tuning is Grid search. Grid search is based on evaluating all possible combinations of given parameter space [22].

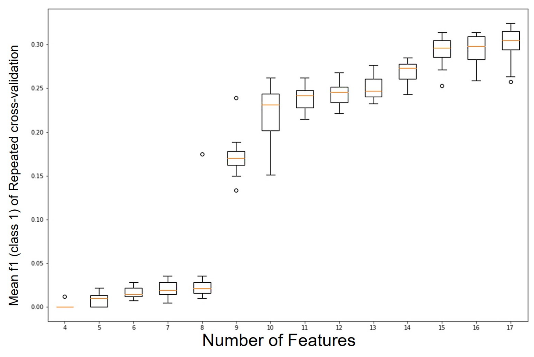
A common and efficient strategy to evaluate the performance of an algorithm with different values of the hyperparameters in the tuning is k-fold cross-validation on the training set [22]. Then averaging the results of repetitions of the whole cross-validation procedure provides a reliable result.



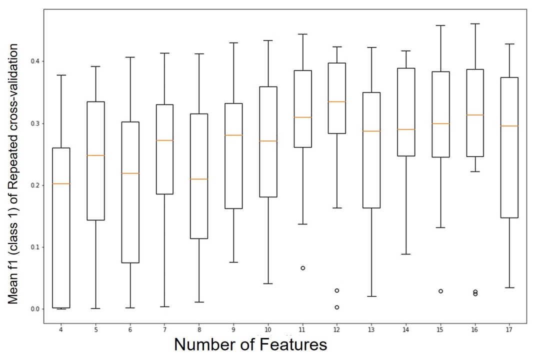
(a) XGBoost



(b) Random Forest



(c) Logistic Regression



(d) Single-layer network (Perceptron)

Figure 4.6: Boxplots of performance means of features, RFE method

In this study, hyperparameter tuning is performed using the grid search method with 5-fold cross-validation. Due to the large size of the training datasets and the available computational capacity, only a percentage of the data is used for tuning. The tuning is applied to the training and validation sets of the same-year 2019 prediction scenario, and the resulting optimal hyperparameters are then used for the other scenarios.

To validate the effectiveness of tuning on a subset of the dataset, the decision tree model is tuned using both the full dataset and the percentage subset. The results from both approaches are found to be the same. Therefore, for the other machine learning models, tuning is performed only on the subset of the data.

4.8 Experimental Setup

Table 4.8 presents a comprehensive summary of all the experimental setups, including the number of features, selected features, and hyperparameters for each machine learning model. The implemented models include logistic regression, random forest, decision tree, and a single-layer network (perceptron), all built using the scikit-learn Python library [23]. Additionally, the XGBoost model was constructed using the XGBoost Python library [40].

Table 4.8: Experimental setup

Model	**# features**	**features**	**Hyperparameters**
Decision Tree	10	hh_assets, ref_age, hh_size, energy_bill, acc_constructionYear, acc_value, acc_area, hh_incomeSource, acc_energyLabel_encoded, acc_ownershipType	criterion: entropy, max_depth: 15, min_samples_leaf: 50
Random Forest	11	hh_assets, ref_age, hh_size, energy_bill, acc_constructionYear, acc_value, acc_area, hh_incomeSource, acc_energyLabel_encoded, acc_ownershipType, move_date_y	criterion: entropy, max_depth: 30, max_features: sqrt, n_estimators: 400
XGBoost	6	hh_assets, hh_size, energy_bill, hh_incomeSource, acc_energyLabel_encoded, acc_ownershipType	colsample_bytree: 0.8, learning_rate: 0.1, max_depth: 6, min_chold_weight: 1, subsample: 1
Logistic Regression	10	hh_mortgageDebt, ref_age, hh_size, energy_bill, acc_constructionYear, acc_value, acc_area, hh_incomeSource, acc_energyLabel_encoded, acc_ownershipType	C: 100, penalty: l2, solver: newton-cg
Single-layer network (Perceptron)	11	hh_mortgageDebt, hh_assets, hh_size, energy_bill, acc_area acc_constructionYear, move_date_y, hh_type gender, acc_Heat acc_energyLabel_encoded, acc_ownershipType	eta0: 1, max_iter: 10

4.9 Performance Evaluation

When dealing with highly imbalanced data, it is crucial to select evaluation metrics and comparison methods that are robust and appropriate for class imbalance. Relying solely on accuracy can be misleading in such scenarios. Instead, it is recommended to use evaluation metrics such as precision, recall, F1-score, or area under the precision-recall curve (AUC-PR) to assess model performance accurately.

In this study, the model's performance will be evaluated using the following metrics will be employed:

1. Precision, recall, and F1-score: These metrics provide insights into the model's ability to correctly identify positive instances (energy poor households) while avoiding false positives.

2. Precision-Recall Curve (PRC) and Area Under the Curve (AUPRC): The PRC graphically depicts the trade-off between precision and recall for various classification thresholds. The AUPRC provides a single scalar value representing the overall performance of the model.

Using these evaluation metrics will enable a robust comparison of the models' performance, particularly in the context of imbalanced data.

4.9.1 Precision, Recall and F1-score

Firstly, a confusion matrix is a square matrix that displays the counts or percentages of true positive (TP), true negative (TN), false positive (FP), and false negative (FN) predictions made by a classifier. It provides a comprehensive summary of the performance of a classification model. Figure 4.7 visually represents the confusion matrix.

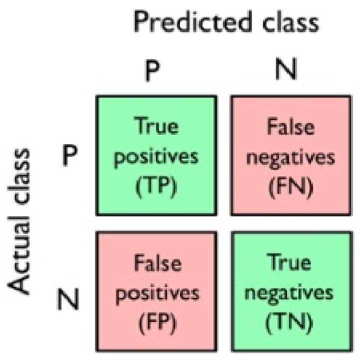


Figure 4.7: The confusion matrix [3]

Precision is an important evaluation metric that measures the proportion of predicted relevant records that are actually relevant to the task at hand [27]. It is formulated as follows:

$$Precision = \frac{TP}{TP + FP} \tag{4.4}$$

Recall, on the other hand, measures the proportion of relevant records that are correctly captured [27]. It is formulated in the following:

$$Recall = \frac{TP}{TP + FN} \tag{4.5}$$

The F1-score is the harmonic mean of precision and recall, providing a balanced measure of the model's overall performance [27]. F1-score is often a good choice as it balances precision and recall, providing a single metric that considers both true positives and false positives. It is formulated as follows:

$$F1 = 2\frac{Precision * Recall}{Precision + Recall} \tag{4.6}$$

In the context of this study, the emphasis is placed on detecting more poor households, which is reflected in the emphasis on precision. Additionally, the aim is to detect fewer non-poor households as poor, which is reflected in the emphasis on recall.

4.9.2 Precision-Recall Curve

To evaluate and compare the performance of a classifier, accuracy, which measures the proportion of correctly predicted labels, is considered inappropriate, especially for imbalanced datasets.

Instead of accuracy, the area under the curve (AUC) is another common metric. The AUC can be calculated using different curves, such as the receiver operating characteristic (ROC) curve or the precision-recall curve. When dealing with highly imbalanced datasets, the AUPRC is often preferred over the AUROC. The AUPRC provides a more appropriate metric for evaluating the performance of models in such scenarios, as it takes into account the precision and recall trade-off in imbalanced datasets [41]. The higher AUPRc shows the better performance of the model.

The Precision-Recall curve illustrates the trade-off between Precision and Recall in a classification model [27]. This curve visually illustrates the performance of the classifier, with the ideal model achieving 100% precision and recall, represented at the upper right corner of the plot [27]. The Precision-Recall curve is constructed by plotting the positive predictive value (precision) against sensitivity (recall) [42].

The area under the Precision-Recall curve (AUPRC) quantifies the overall performance of the classifier and represents the area under the plotted curve. For AUPRC, the corresponding baseline value is the proportion of true positive cases in the distribution [42]. In other words, it is the ratio of positive cases in the dataset expressed as [27]:

$$Baseline\ Value = \frac{positive\ cases}{positive\ cases + negative\ cases} \tag{4.7}$$

An ideal classifier predicts every positive instance (perfect recall) without incorrectly identifying any negative instances (perfect precision) [42]. As a result, the area under the Precision-Recall curve (AUPRC) for an ideal classifier is 1. The AUPRC serves as a metric to measure the overall performance of the classifier and ranging from 0 to 1.

5

Models results and analysis

5.1 Predictive Modeling: Performance Evaluation and Results

In this chapter, the results obtained from machine learning models for three different predictions of energy poverty will be presented and analyzed. Furthermore, all models are re-run while considering the issue of class imbalance. To address this problem, class weights are incorporated into the models, as discussed in chapter 4. The performance of different models is compared, and the results are presented in confusion matrices. Additionally, an ensemble modeling approach using majority voting is applied in each situation. The situations considered in this chapter are as follows:

- Same-year prediction for the year 2019
- Same-year prediction for the year 2020
- Next-year prediction for the year 2020

5.1.1 Same-year prediction for the year 2019

Table 5.1 presents the performance metrics of the machine learning models for predicting energy poverty in the same-year scenario, specifically focusing on class 1, which represents energy poor households. The precision, recall, and f1-score metrics are reported for class 1, as they are the key evaluation measures for accurately identifying energy poverty. Given that energy poverty prediction is the main goal of this study, the focus is placed on the performance of class 1 first. In addition, the performances of the models by adding the class weight are shown for comparison.

Table 5.1: Class 1 performance of Same-year 2019

Model		**Class 1, Energy poor**		
	Class weight	**Precision**	**Recall**	**F1-score**
Decision Tree	without CW	0.76	0.48	0.59
	with CW	0.36	**0.87**	0.51
Random Forest	without CW	**0.77**	0.48	0.60
	with CW	0.67	0.57	**0.62**
XGBoost	without CW	0.75	0.45	0.56
	with CW	0.34	**0.87**	0.49
Logistic Regression	without CW	0.64	0.17	0.27
	with CW	0.24	0.83	0.38
Single-layer Network	without CW	0.24	0.40	0.30
	with CW	0.18	0.73	0.29

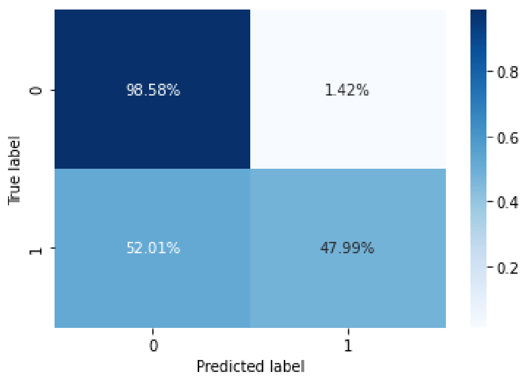
Table 5.1 demonstrates that the random forest model achieves the highest f1-score for predicting energy poverty, even after adding class weight to the models. The f1-scores for logistic regression and single-layer network increase with class weight, while for the decision tree model, it slightly decreases.

The most notable observation is the increase in recall values for all models after adding class weight. Before incorporating class weight, the highest recall value is 0.48 for decision tree and random forest models. However, after that, the highest recall value overall is 0.87 for the decision tree and XGBoost models. Additionally, it is worth mentioning that logistic regression and single-layer network models show a significant rise in recall from 0.17 and 0.40 to 0.83 and 0.73, respectively.

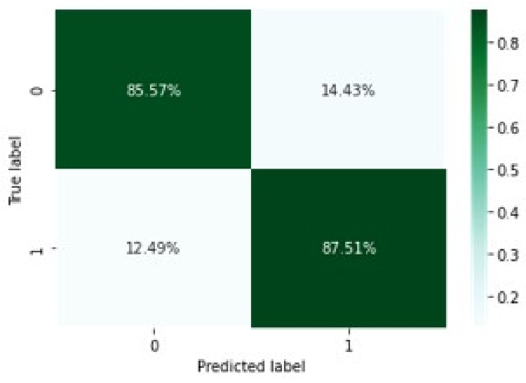
Regarding precision, the random forest model achieves the highest value of 0.77. However, after adding class weight, the precision increases for logistic regression and single-layer network models, while it decreases for the Tree-based models. Still, the highest precision with class weight is 0.67 from the random forest model.

Figures 5.1 through 5.5 depict the confusion matrices illustrating the performance of the machine learning models on test data, representing the true values. Each row in the confusion matrices sums up to 100%.

Figure 5.1 illustrates for the decision tree model that the inclusion of class weights leads to an increase in false positives (FP) and, consequently, an increase in type I errors. This change also results in a decrease in true negatives (accurate predictions of class 0), while significantly improving the true positives (accurate predictions of class 1 - energy poverty). It appears that the decision tree model focuses more on improving the prediction of class 1 but at the expense of reduced performance in predicting class 0.



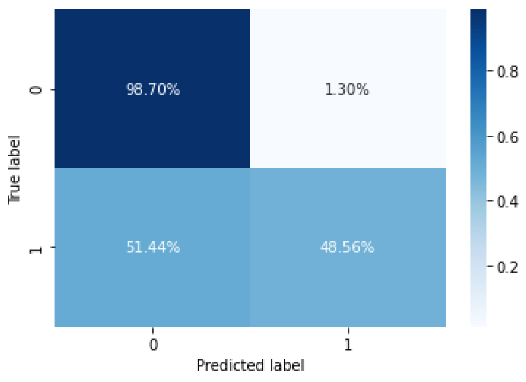
(a) Without class_weight



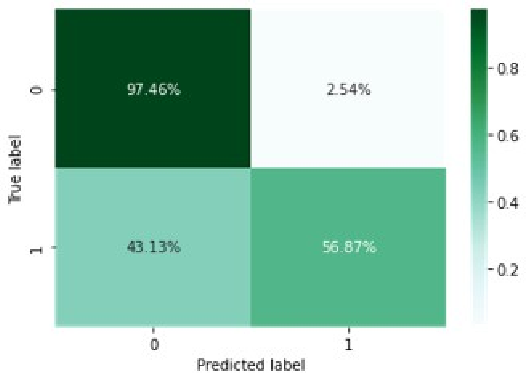
(b) With Class_Weight

Figure 5.1: Decision Tree confusion matrix for Same-year 2019 prediction

Figure 5.2 demonstrates that for the random forest model, the increase in true positives (TP) is not substantial when class weights are added, leading to a relatively constant type I error. As mentioned previously, the addition of class weights improves the type II error, indicating a better performance in correctly predicting class 1 (energy poverty).



(a) Without class_weight

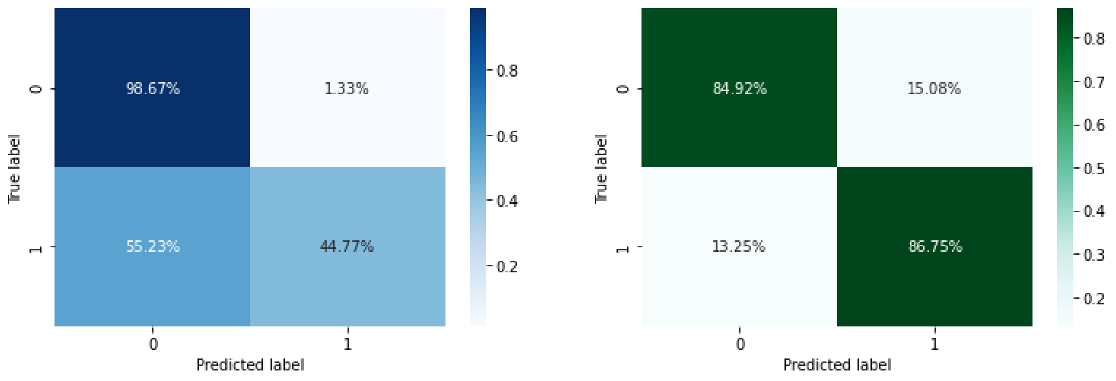


(b) With Class_Weight

Figure 5.2: Random Forest confusion matrix for Same-year 2019 prediction

Figure 5.3 demonstrates that for XGBoost model, the performance is very similar to that of DT, with a slightly higher incidence of type II (but also type I) errors.

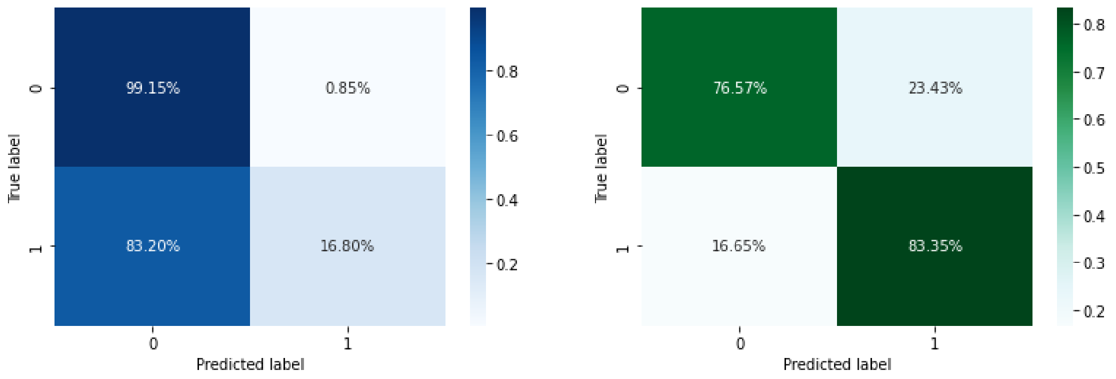
Figure 5.4 illustrates a notable increase in true positives (TP) in the logistic regression model after the inclusion of class weights. However, there is a significant decrease in true negatives (TN) compared to XGBoost. This improvement indicates that the logistic regression model shows a considerable improvement in its prediction ability, particularly for class 1 (energy poverty). Although the logistic regression model's performance on TP



(a) Without class_weight **(b)** With Class_Weight

Figure 5.3: XGBoost confusion matrix for Same-year 2019 prediction

is not as strong as the decision tree model, the inclusion of class weights significantly enhances its overall performance.



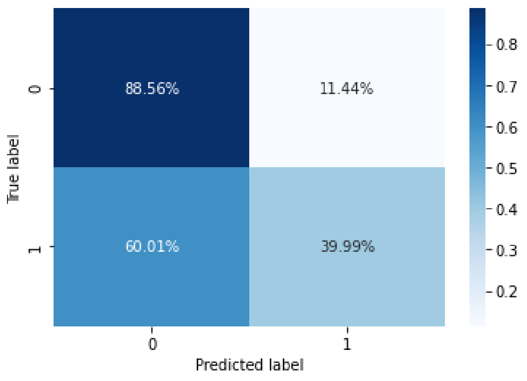
(a) Without class_weight **(b)** With Class_Weight

Figure 5.4: Logistic Regression confusion matrix for Same-year 2019 prediction

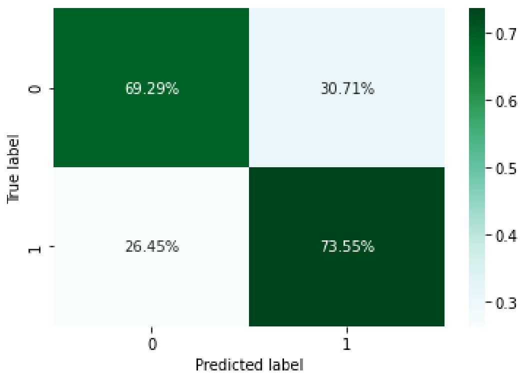
Similar to the logistic regression algorithm, the neural network model also exhibits a significant improvement in performance when class weights are incorporated, as shown in Fig. 5.5. However, in terms of overall performance and improvement, the logistic regression model outperforms the single-layer network model.

In general, the inclusion of class weight can significantly enhance recall, which is one of the primary objectives of this study. However, achieving high precision and recall simultaneously for class 1 is not feasible.

Table 5.2 demonstrates the performance metrics of the entire model for both classes, considering precision, recall, and f1-score. The reported metrics represent the macro average



(a) Without class_weight



(b) With Class_Weight

Figure 5.5: Single-layer Network confusion matrix for Same-year 2019 prediction

from the corresponding confusion matrix. In addition, the running time of each training is mentioned.

Table 5.2: Performance of Same-year 2019 prediction

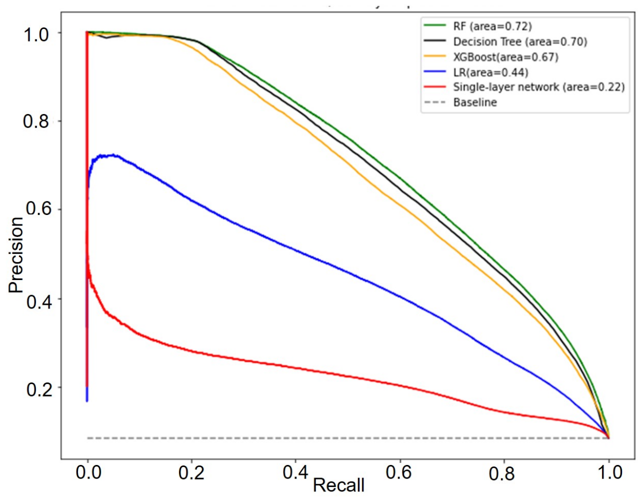
Model			**Macro Average**		
	Class_weight	**Running Time**	**Precision**	**Recall**	**F1-score**
Decision Tree	without CW	129s	0.85	0.73	0.78
	with CW	101s	0.67	0.86	0.71
Random Forest	without CW	11919s	0.86	0.74	0.78
	with CW	12600s	0.82	0.77	0.79
XGBoost	without CW	507s	0.85	0.72	0.76
	with CW	123s	0.66	0.86	0.70
Logistic Regression	without CW	277s	0.78	0.58	0.61
	with CW	103s	0.61	0.80	0.62
Single-layer Network	without CW	12.6 s	0.59	0.64	0.61
	with CW	13.1 s	0.57	0.71	0.55

First of all, there is not a significant difference in the running time of the models with or without class weight.

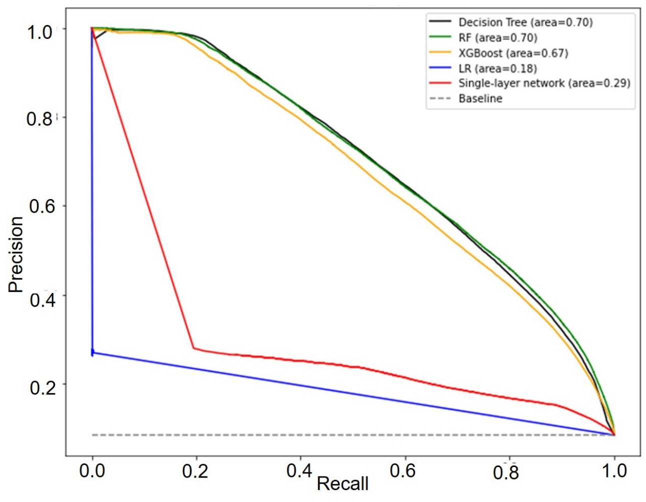
Before adding class weight, it can be observed that the Tree-based models (DT, RF and XGBoost) have similar performance and outperform the linear models (LR and single-layer network). After adding class weight, there is an improvement in the f1-score for all models, especially for LR and single-layer network. The random forest model with class weight achieves the highest f1-score of 0.79.

The decision tree and XGBoost models with class weight have the highest recall value. The precision values are quite similar for the Tree-based models before adding class weight, with a precision of 0.85.

Figure 5.6 illustrates the precision-recall curves of all the machine learning models, both with and without class weights. By plotting these curves, we can compare the models in terms of both precision and recall and also evaluate their performance relative to the baseline. The precision-recall curve is a suitable visualization as it represents the trade-off between precision (the ability to make accurate positive predictions) and recall (the ability to correctly identify positive instances).



(a) Without class_weight



(b) With Class_Weight

Figure 5.6: Precision-Recall Curve for Same-year 2019 prediction

In Figure 5.6(a), the random forest model achieves the highest area under the curve, indicating a good balance between recall and precision. This means that the random forest model performs well in terms of both identifying positive instances accurately (high recall) and making accurate positive predictions (high precision). On the other hand, the single-layer network model shows the worst performance among the models, suggesting lower precision and recall compared to the other models. In Figure 5.6(b), the logistic regression model exhibits the lowest AUPRC. However, after adding class weight to the models, the AUPRC of the decision tree and random forest models became equal, which is mainly due to the reduction in AUPRC for LR.

It is important to note that there is a distinction between confusion matrices, which primarily show recall, and precision-recall curves, which display the trade-off between precision and recall. While adding class weight may improve recall and create more balanced confusion matrices, it does not necessarily lead to better AUPRC. In fact, the trade-off between precision and recall may worsen, resulting in a lower AUPRC value. Therefore, it is crucial to consider both recall and precision simultaneously when evaluating model performance, especially in the context of imbalanced datasets, to ensure a well-balanced approach to prediction accuracy.

Ensemble modeling

In the last step, an ensemble modeling approach is applied to combine the five models mentioned in this section. As mentioned in 2.2.6, the technique in this study is majority voting or hard voting.

Table 5.3 presents the results of ensemble modeling for the same-year prediction scenario in 2019, considering both with and without class weight (CW). The table includes performance metrics such as precision, recall, and F1-score for class 1 prediction, as well as the macro-average for both classes. Consequently, their related confusion matrices are depicted in Figure 5.7.

Table 5.3: Ensemble modeling for same-year 2019 prediction

Model		Precision	Recall	F1-score
Ensemble, without CW	Class 1	**0.79**	0.45	0.57
	Macro avg	0.87	0.72	0.77
Ensemble, with CW	Class 1	0.37	**0.84**	0.52
	Macro avg	0.68	0.86	0.72

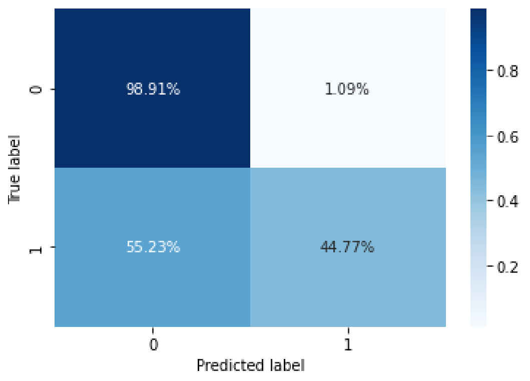
As the tree-based models demonstrated significantly good results, we decided to implement ensemble modeling using only tree-based models, including decision trees, random forest, and XGBoost. The results are presented in Table 5.4. The outcomes indicate that the performance of ensemble modeling, which combines all five models, is better than using only tree-based models. Therefore, we will proceed with running ensemble modeling for all the models and analyze the results in-depth at a later stage.

Table 5.4: Ensemble modeling for same-year 2019 prediction **with tree-based models**

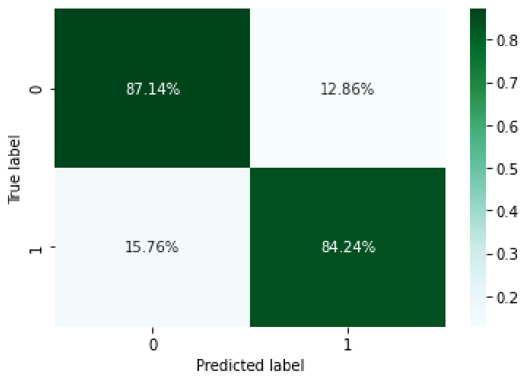
Model		Precision	Recall	F1-score
Ensemble, without CW	Class 1	**0.78**	0.47	0.59
	Macro avg	0.86	0.73	0.78
Ensemble, with CW	Class 1	0.66	**0.58**	0.62
	Macro avg	0.81	0.78	0.79

From the data presented in Table 5.3, we can observe several key findings. Firstly, in terms of the f1-score, the ensemble modeling without class weight achieves the highest value compared to all individual models. However, it is worth noting that the highest f1-score obtained by the ensemble model (0.57) is still lower than the f1-scores of the DT, RF, and XGBoost models.

When considering precision, the ensemble modeling without CW also demonstrated the highest values. Specifically, the precision for class 1 and the macro-average are 0.79 and 0.87, respectively, surpassing the precision of the individual models. Additionally, as observed previously, the inclusion of class weight led to higher recall values for all models. Consequently, the ensemble modeling with CW achieved the highest recall among the models; however, it was still lower than the recall of the decision tree and XGBoost models when evaluated individually.



(a) Without class_weight



(b) With Class_Weight

Figure 5.7: Ensemble modeling confusion matrices for Same-year 2019 prediction

5.1.2 Same-year prediction for the year 2020

The performance indicators of the machine learning models for the prediction of energy poverty in the same-year scenario for 2020 are presented in Table 5.5, with a focus on class 1, which represents energy poor households.

Similar to the last section 5.1.1, only for class 1, the precision, recall, and f1-score metrics are presented since they are critical evaluation indicators for effectively detecting energy poverty. Furthermore, the performance of the models after adding the class weight is displayed for comparison.

Table 5.5: Class 1 performance of Same-year 2020

Model		**Class 1, Energy poor**		
	Class_weight	**Precision**	**Recall**	**F1-score**
Decision Tree	without CW	0.74	0.43	0.55
	with CW	0.29	**0.87**	0.43
Random Forest	without CW	**0.77**	0.43	0.55
	with CW	0.68	0.50	**0.57**
XGBoost	without CW	0.74	0.39	0.51
	with CW	0.28	0.85	0.42
Logistic Regression	without CW	0.62	0.12	0.20
	with CW	0.19	0.82	0.31
Single-layer Network	without CW	0.21	0.0005	0.001
	with CW	0.19	0.75	0.30

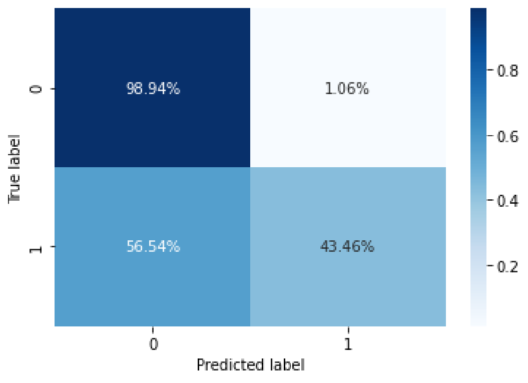
The random forest and decision tree models get the greatest f1-score, 0.55, for predicting energy poverty, as shown in Table 5.5. It is less than 0.59 for the same-year 2019 scenario. random forest model has the highest value, 0.57, after adding class weight to the models, indicating an increase in total. The f1-scores for the single-layer network model increase significantly with class weight, whereas they fall marginally for the decision tree and logistic regression models.

The most noticeable difference in this scenario, as in the prior one, is an increase in recall values for all models after adding class weight. The highest recall value after including class weight is 0.87 for decision tree models and, consequently, 0.85 for the XGBoost model, similar to the same-year 2019 scenario. Prior to that, the maximum recall value for the decision tree and random forest models was 0.43. Similarly, the recall of logistic regression and single-layer network models increases significantly from 0.12 and 0.0005 to 0.82 and 0.75, respectively.

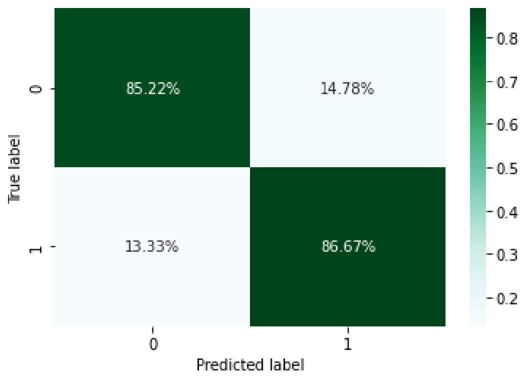
The random forest model achieves the greatest precision value of 0.77. However, when class weight is added, the precision falls for all other models. As a result, the random forest model again has the maximum precision with class weight of 0.68.

Figures 5.8 through 5.12 depict the confusion plots illustrating the performance of the machine learning models on test data, representing the true values. Each row in the confusion matrices sums up to 100%.

Figure 5.8 demonstrates that for DT by enhancing the recall of class 1, there is a decrease in the recall of class 0. However, it is noteworthy that despite a significant increase of 43% in true positives (TP), there is only a 13% decrease in true negatives (TN).



(a) Without class_weight



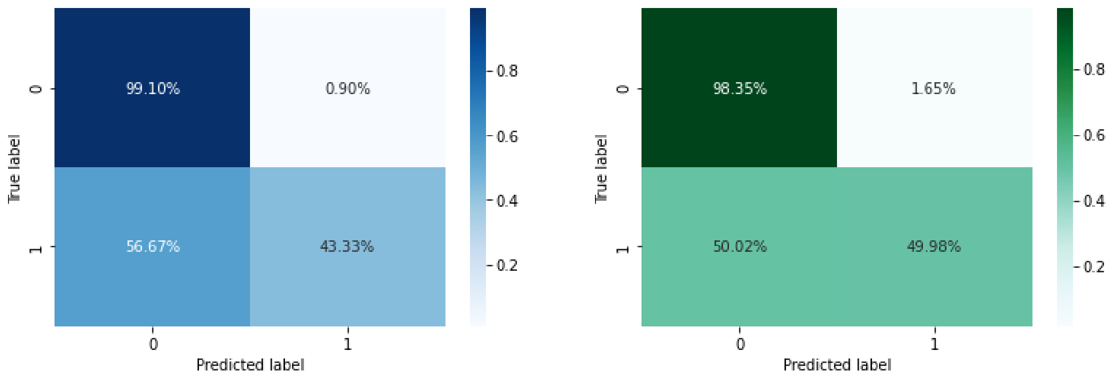
(b) With Class_Weight

Figure 5.8: Decision Tree confusion matrix for Same-year 2020 prediction

Figure 5.9 depicts that the random forest model does not show significant changes in performance after adding class weight, similar to the decision tree model. The prediction of class 1 is roughly evenly distributed between true and false predictions.

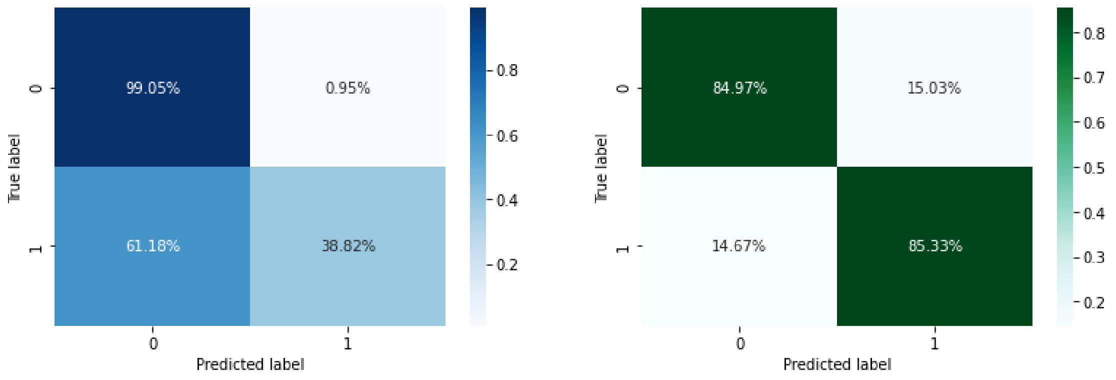
Figure 5.10 demonstrates for XGBoost a significant improvement in TP prediction by adding class weight. The final TP is similar to the decision tree model, but the extent of improvement in TP is better than that of the decision tree model.

Figure 5.11 and Figure 5.12 show that for both the logistic regression (LR) and single-layer network models, there is a considerable increase in true positives (TP). However, the increase in TP is more significant for the single-layer network model. The final recall values for both models are considerable, with RF achieving 81.89% and single-layer network achieving 75.46%. However, when comparing the false positives (FP) and false negatives (FN), the single-layer network model performs better. This indicates that boosting recall using the single-layer network model reduces the two forms of error more than logistic regression.



(a) Without class_weight **(b)** With Class_Weight

Figure 5.9: Random Forest confusion matrix for Same-year 2020 prediction



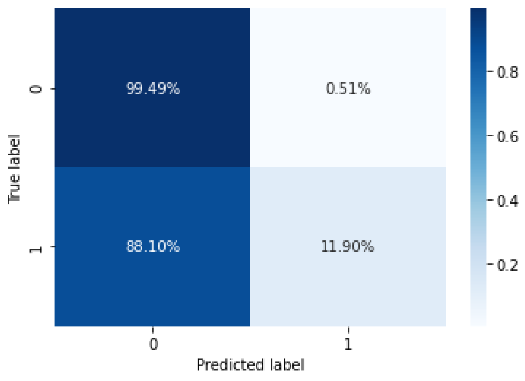
(a) Without class_weight **(b)** With Class_Weight

Figure 5.10: XGBoost confusion matrix for Same-year 2020 prediction

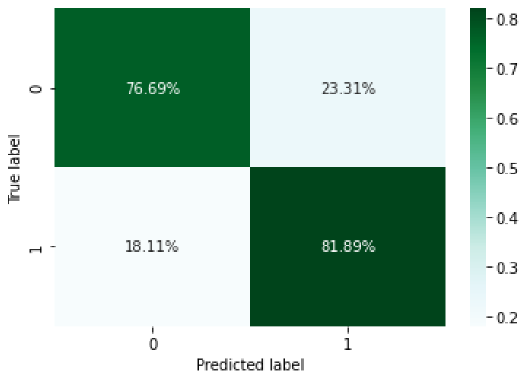
In general, the decision tree and XGBoost models show better recall performance compared to the other models, and their recall performance becomes almost equal after adding class weight. It is worth noting the significant improvement of the linear models after addressing the class imbalance issue.

Table 5.6 displays the overall model's performance for both classes. The metrics presented are the macro average of the corresponding confusion matrix. In addition, the duration of each training is specified.

Similar to the same-year 2019 scenario, there is no noticeable difference in the running times of the models with and without CW. Besides, before adding class_weight, it is clear that the Tree-based models (DT, RF, and XGBoost) outperform the linear models, logistic regression and single-layer network.

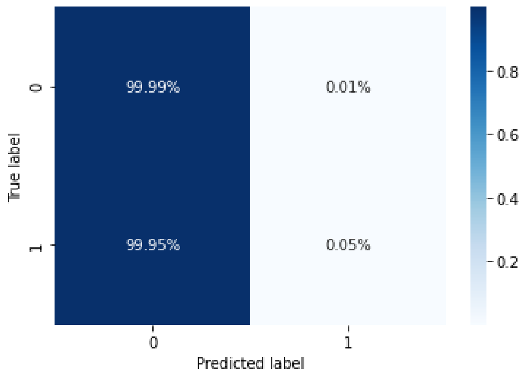


(a) Without class_weight

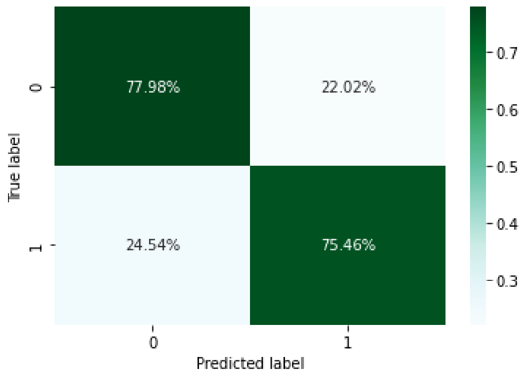


(b) With Class_Weight

Figure 5.11: Logistic Regression confusion matrix for Same-year 2020 prediction



(a) Without class_weight



(b) With Class_Weight

Figure 5.12: Single-layer Network confusion matrix for Same-year 2020 prediction

Table 5.6: Performance of Same-year 2020 prediction

Model			**Macro Average**		
	Class_weight	**Running Time**	**Precision**	**Recall**	**F1-score**
Decision Tree	without CW	149s	0.71	0.71	0.76
	with CW	108s	0.64	0.86	0.67
Random Forest	without CW	12,174s	0.86	0.71	0.76
	with CW	12,257s	0.82	0.74	0.77
XGBoost	without CW	123s	0.85	0.69	0.74
	with CW	142s	0.63	0.85	0.67
Logistic Regression	without CW	162s	0.78	0.56	0.58
	with CW	90s	0.59	0.79	0.58
Single-layer Network	without CW	14.5s	0.57	0.50	0.48
	with CW	13.6s	0.58	0.77	0.59

The f1-score improves after adding class weight for random forest and single-layer network. Similarly to the previous year, the random forest model with class weight earns the greatest f1-score of 0.77.

The decision tree model with class weight achieves the highest recall value. Before adding class weight, the precision levels for the RF and XGBoost models are relatively comparable, with precision values of 0.86 and 0.85, respectively.

Figure 5.13 displays the precision-recall curves for all the machine learning models in the same-year 2020 prediction, both before and after adding class weight.

In Figure 5.13(a), similar to the 2019 case (Figure 5.6), the random forest model exhibits the highest precision and recall values. However, the area under the curve (AUC) for the single-layer network model is larger compared to the 2019 prediction. It is worth noting that as the recall increases, the precision tends to decrease, and vice versa, as observed for all the models except the single-layer network model.

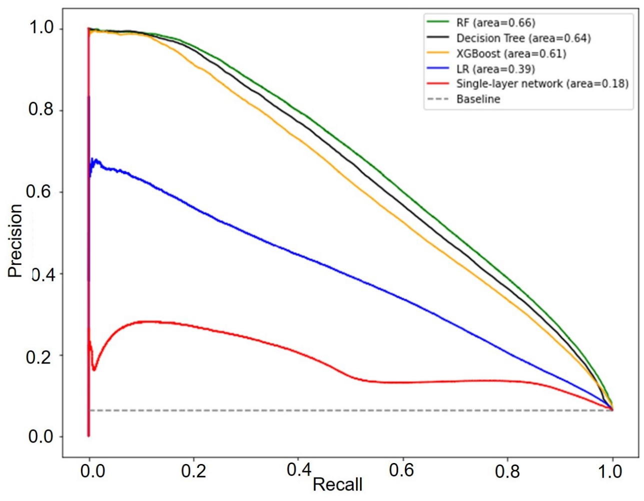
Ensemble modeling

In the same-year prediction scenario for 2020, an ensemble modeling approach using hard voting was employed on the combined individual machine learning models. The results of this ensemble modeling, with and without class weight (CW), are presented in Table 5.7. The table provides performance metrics for class 1 prediction, as well as the macro-average for both classes. Additionally, Figure 5.14 showcases the corresponding confusion matrices for the ensemble modeling results.

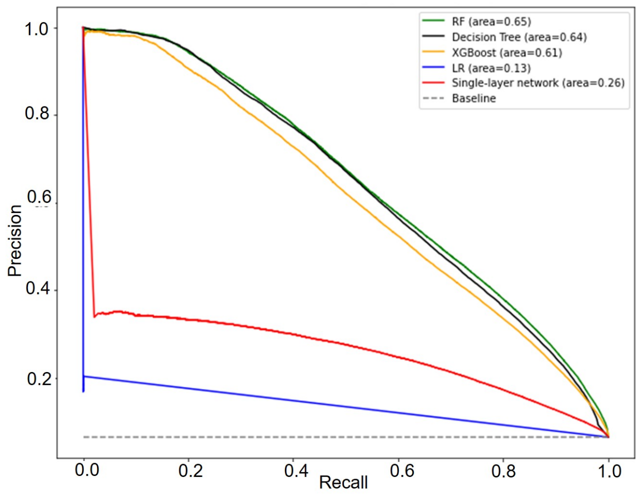
Table 5.7: Ensemble modeling for same-year 2020 prediction

Model		Precision	Recall	F1-score
Ensemble, without CW	Class 1	**0.81**	0.36	0.50
	Macro avg	0.88	0.68	0.74
Ensemble, with CW	Class 1	0.30	**0.82**	0.44
	Macro avg	0.64	0.84	0.68

The results of the ensemble modeling for the same-year prediction scenario in 2020 align with the findings discussed in Section 5.1.1 for the 2019 scenario. The ensemble model without class weight achieves the highest f1-score and precision, although the f1-score may be lower than that of the individual models (DT, RF, LR, XGBoost) in Table 5.7. However, the highest precision (0.81) in the ensemble model surpasses that of all the individual models.



(a) Same-year 2020 PRC



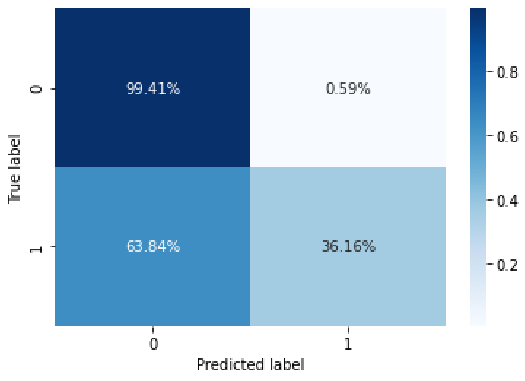
(b) Same-year 2020 PRC with CW

Figure 5.13: Precision-Recall Curve for Same-year 2020 prediction

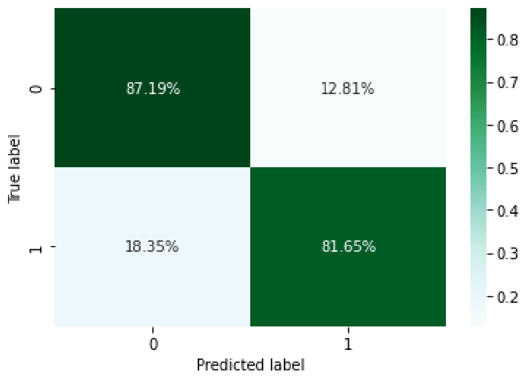
In terms of recall, the ensemble model with class weight performs the best, but it is worth noting that the decision tree and XGBoost individually achieve better results with class weight.

5.1.3 Next-year prediction for the year 2020

This section, like 5.1.1 and 5.1.2, begins with class 1 prediction performance. The performance indicators of the machine learning models for the prediction of energy poverty in the next-year prediction for 2020 are presented in Table 5.8, with a focus on class 1.



(a) Without class_weight



(b) With Class_Weight

Figure 5.14: Ensemble modeling confusion matrices for Same-year 2020 prediction

Given that predicting energy poverty is the primary purpose of this study, the emphasis is placed primarily on the performance of class 1. The performance of the models after adding the class weight is displayed as well for comparison.

Table 5.8 shows similar results to the prior situations, with the most obvious difference being an increase in recall values for all models after adding class weight.

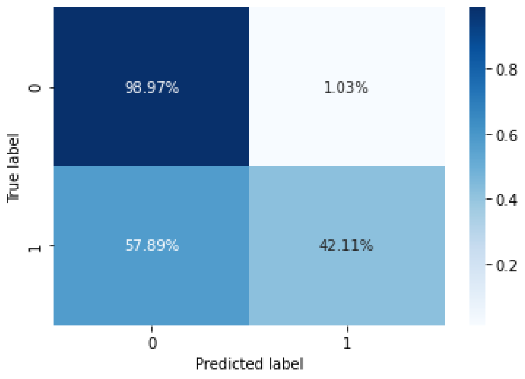
Table 5.8: Class 1 Performance of Next-year 2020

Model		**Class 1, Energy poor**		
	Class_weight	**Precision**	**Recall**	**F1-score**
Decision Tree	without CW	**0.74**	0.42	0.54
	with CW	0.33	**0.83**	0.47
Random Forest	without CW	0.73	0.44	0.56
	with CW	0.68	0.51	0.58
XGBoost	without CW	**0.74**	0.37	0.49
	with CW	0.32	0.81	0.46
Logistic Regression	without CW	0.64	0.12	0.20
	with CW	0.21	0.77	0.33
Single-layer Network	without CW	0.27	0.39	0.32
	with CW	0.22	0.68	0.33

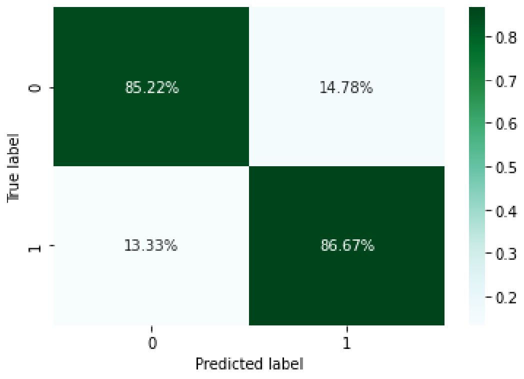
Prior to introducing class weight, the highest recall value for the decision tree model is 0.44, which is lower than in earlier instances. However, the maximum recall value is 0.83 for the decision tree model, like in previous scenarios. It is also worth noting that the logistic regression model's recall of class 1 increased from 0.12 to 0.77. In terms of precision, there is a consistent pattern of falling values for all models after adding class weight. Tree-based models have a comparable precision of 0.74 without class weight, while the random forest model displays the highest precision with class weight.

Finally, Table 5.8 shows that the random forest model achieves the greatest f1-score for predicting energy poverty, 0.56 and 0.58, respectively, before and after adding class weight to the models. The f1-scores for all models differ slightly.

Figures 5.15 through 5.19 depict the confusion plots illustrating the performance of the machine learning models on test data, representing the true values. They show the performance of models in terms of true positives (TP), true negatives (TN), and the two types of errors before and after adding class weight. Each row in confusion matrices sums up to 100%.



(a) Without class_weight



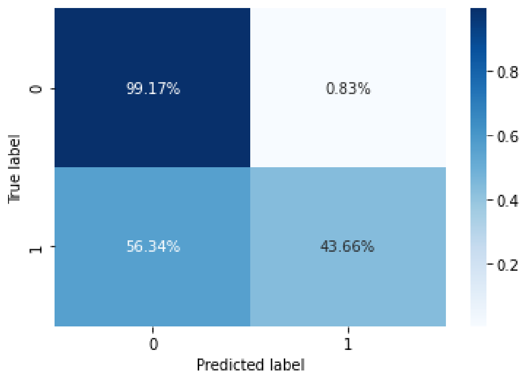
(b) With Class_Weight

Figure 5.15: Decision Tree confusion matrix for Next-year 2020 prediction

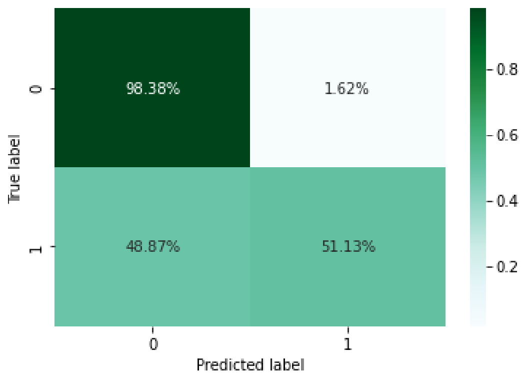
Figure 5.15 depicts that prior to adding class weight, the model achieves an exceptionally high recall of 98.97% for class 0. This high recall is mainly due to the model correctly predicting nearly all instances of class 0. After adding class weight the model's performance improves with a TP rate of 86.67%, indicating a significant enhancement in correctly identifying instances of class 1. This improvement comes at the cost of a slight decrease in TN, resulting in an almost equal representation of the two types of errors.

Figure 5.16 showcases the results for the random forest (RF) model similarly as in previous scenarios (Fig. 5.2, 5.9). When class weight is added to the model, there is no significant improvement observed in the prediction performance for class 1.

Figure 5.17 for XGBoost illustrates that by increasing the number of true positives, there is a decrease in the number of true negatives. Comparing it to the decision tree model, both models achieve similar results in terms of true positives and true negatives.

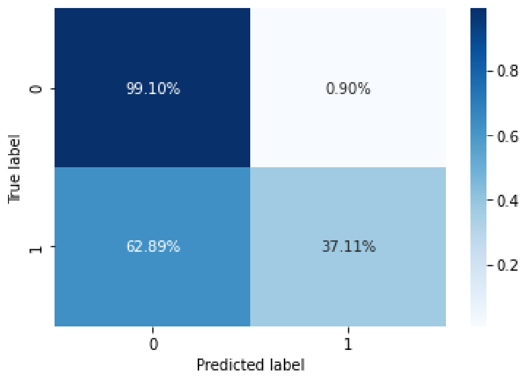


(a) Without class_weight

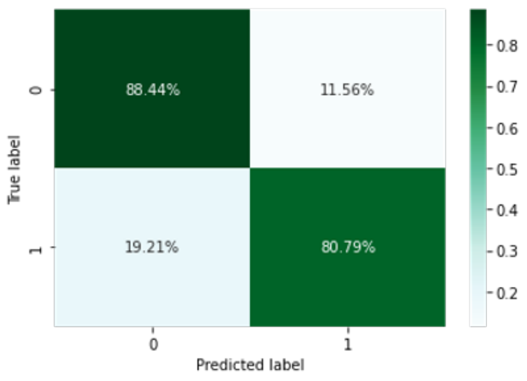


(b) With Class_Weight

Figure 5.16: Random Forest confusion matrix for Next-year 2020 prediction



(a) Without class_weight

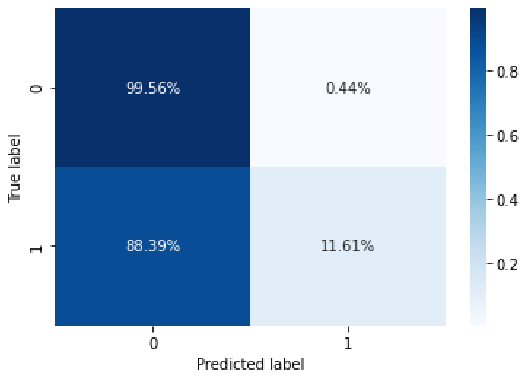


(b) With Class_Weight

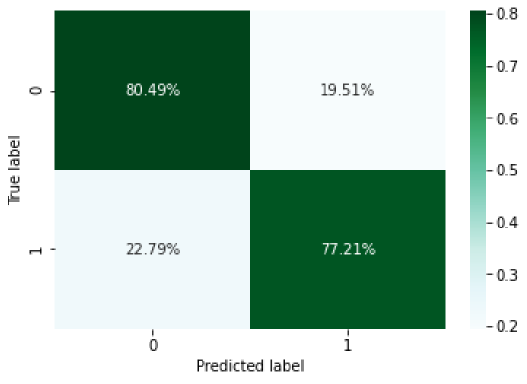
Figure 5.17: XGBoost confusion matrix for Next-year 2020 prediction

Figure 5.18 demonstrates that the recall for class 1 significantly increases in the logistic regression model when adding class weights. However, when comparing it to the decision tree model, the LR model achieves a higher number of true positives but also experiences a larger decrease in true negatives. This suggests that the LR model has a higher rate of false positives (FP) compared to the decision tree model. In terms of error rates, the decision tree model performs better, as it has lower values for both types of error (FP and FN) compared to the logistic regression model.

Figure 5.19 displays the performance improvement of the model after applying class weight. However, this improvement is less prominent in other models. Notably, there is a significant decrease in true negatives (TN) compared to the increase in true positives (TP). This indicates that the single-layer model faces challenges in achieving a high number of

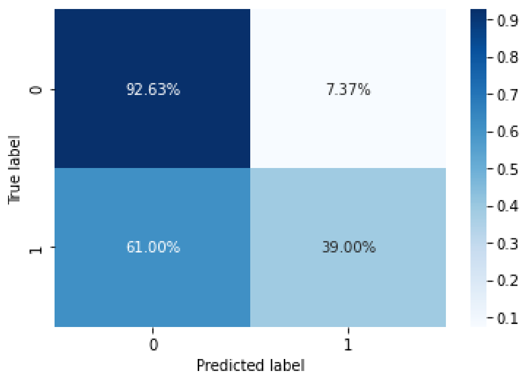


(a) Without class_weight

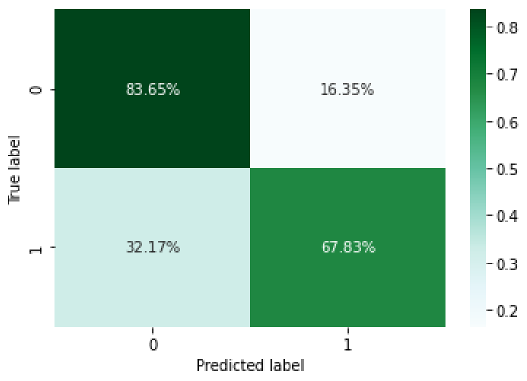


(b) With Class_Weight

Figure 5.18: Logistic Regression confusion matrix for Next-year 2020 prediction



(a) Without class_weight



(b) With Class_Weight

Figure 5.19: Single-layer Network confusion matrix for Same-year 2019 prediction

true positives without simultaneously increasing the rate of false positives. In general, the single-layer network model's ability to correctly predict instances of class 1 comes at the cost of a decrease in its ability to accurately predict instances of class 0.

Table 5.9 shows the performance metrics of the entire model for both classes for the macro average from the corresponding confusion matrix. In addition, the running time of each training is mentioned.

The running times of the models with and without class weight are comparable, with the exception of random forest, which takes less than an hour.

Table 5.9: Performance of Next-year 2020 prediction

Model			**Macro Average**		
	Class weight	**Running Time**	**Precision**	**Recall**	**F1-score**
Decision Tree	without CW	136s	0.85	0.70	0.76
	with CW	154s	0.66	0.86	0.70
Random Forest	without CW	22,133s	0.87	0.71	0.77
	with CW	18,750s	0.82	0.75	0.78
XGBoost	without CW	194s	0.85	0.68	0.73
	with CW	150s	0.65	0.85	0.61
Logistic Regression	without CW	273s	0.78	0.57	0.60
	with CW	120s	0.60	0.79	0.61
Single-layer Network	without CW	16.3 s	0.59	0.59	0.59
	with CW	22.4 s	0.57	0.76	0.53

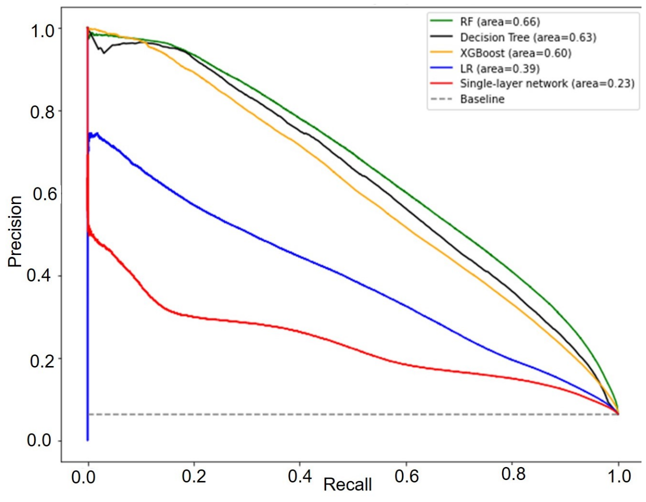
Before adding class weight, it is clear that the Tree-based models (DT, RF, and XGBoost) outperform the linear models. The random forest model with class weight has the best recall of 0.77. Recall values are increased after adding class weight, as in Table 5.8. The decision tree model with class weight achieves the highest recall value. The greatest precision value for RF with CW is 0.87.

The final comparison of the models is presented in Figure 5.20, where the precision-recall curves with and without class weight are plotted together. Consistent with previous observations, the random forest model exhibits the highest area under the curve (AUC), indicating a strong balance between precision and recall. The precision-recall curve for the logistic regression model appears almost linear, suggesting a less trade-off between precision and recall. Prior to the addition of class weight, the single-layer network model had the lowest AUC among the models. After adding CW, In Figure 5.20(b), the random forest model continues to have the highest Area Under the Precision-Recall Curve (AUPRC), with a value of 0.66. On the other hand, after adding class weight, the logistic regression model exhibits the lowest AUPRC, indicating a decrease in the trade-off between precision and recall for this model.

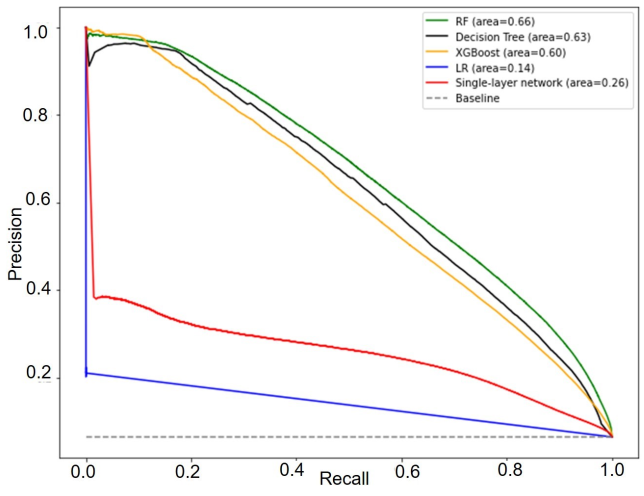
Ensemble modeling

Ultimately, the five models discussed in this section are combined using an ensemble modeling approach with majority voting or hard voting technique.

Table 5.10 presents the results of ensemble modeling for the next-year prediction scenario, considering both with and without class weight (CW). The table includes performance metrics, and Figure 5.21 is their related confusion matrices.



(a) Next-year 2020 PRC



(b) Next-year 2020 PRC with CW

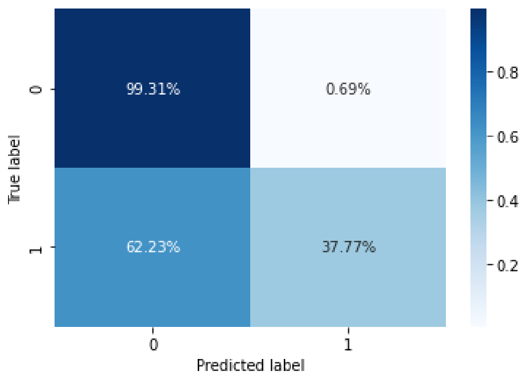
Figure 5.20: Precision-Recall Curve for Next-year 2020 prediction

Table 5.10: Ensemble modeling for next-year 2020 prediction

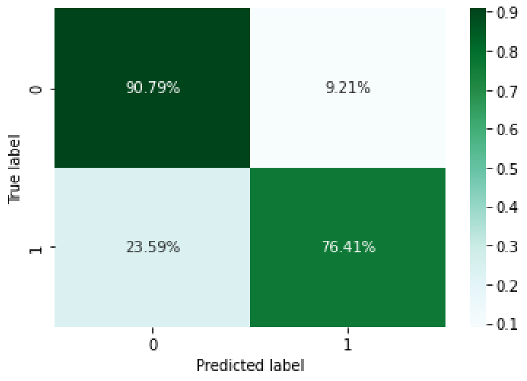
Model		Precision	Recall	F1-score
Ensemble, without CW	Class 1	**0.79**	0.38	0.51
	Macro avg	0.87	0.68	0.74
Ensemble, with CW	Class 1	0.36	**0.76**	0.49
	Macro avg	0.67	0.84	0.72

From Table 5.10, it is evident that the ensemble modeling without class weight achieves the highest precision of 0.79, which surpasses the individual models. However, when considering recall, the ensemble modeling with class weight attains a value of 0.76, which is slightly lower than that of the decision tree model or XGBoost individually.

Figure 5.21 presents the confusion matrices for ensemble modeling with and without class weight. It can be observed that the number of true positives is doubled with the addition of class weight, but there is a 9% decrease in the true negatives.



(a) Without class_weight



(b) With Class_Weight

Figure 5.21: Ensemble modeling confusion matrices for Next-year 2020 prediction

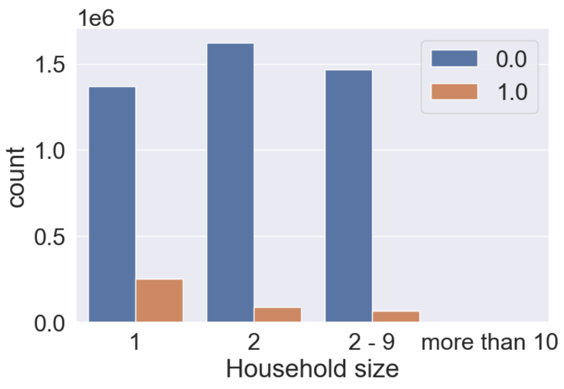
5.2 Relative features analysis

In chapter 4, Table 4.8 presents the common features shared by all the machine learning models. These features are essential for achieving the highest predictive power. The four common features are as follows:

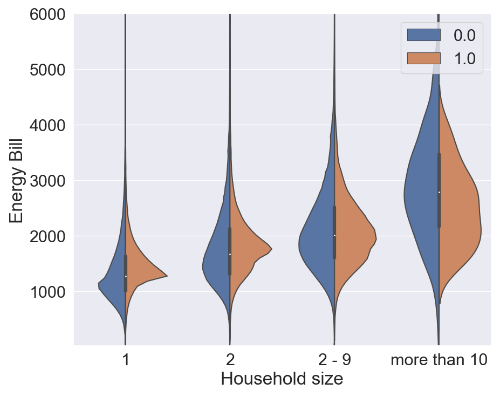
- Household size (hh_size)
- Energy bill (energy_bill)
- Energy label of accommodation (acc_energyLabel_encoded)
- Ownership type of the accommodation (acc_ownershipType)

Additionally, household assets (hh_assets) and household income source type (hh_incomeSource) are also common in 4 out of 5 models. In the following section, we will visualize the relationship of energy poor households for each of these common features using bar plots based on their measurements. Furthermore, we will utilize violin plots to compare the distribution of energy bills between energy poor and non-energy poor households. It is worth mentioning that violin plots are normalized on peak height, not on the area.

In Figure 5.22, we present the distribution of energy poor households categorized by their household size. The chart provides insights into the prevalence of energy poverty among different household sizes, aiding in our understanding of its impact on varying family compositions.



(a) Numbers



(b) Distribution regarding energy bill (euro)

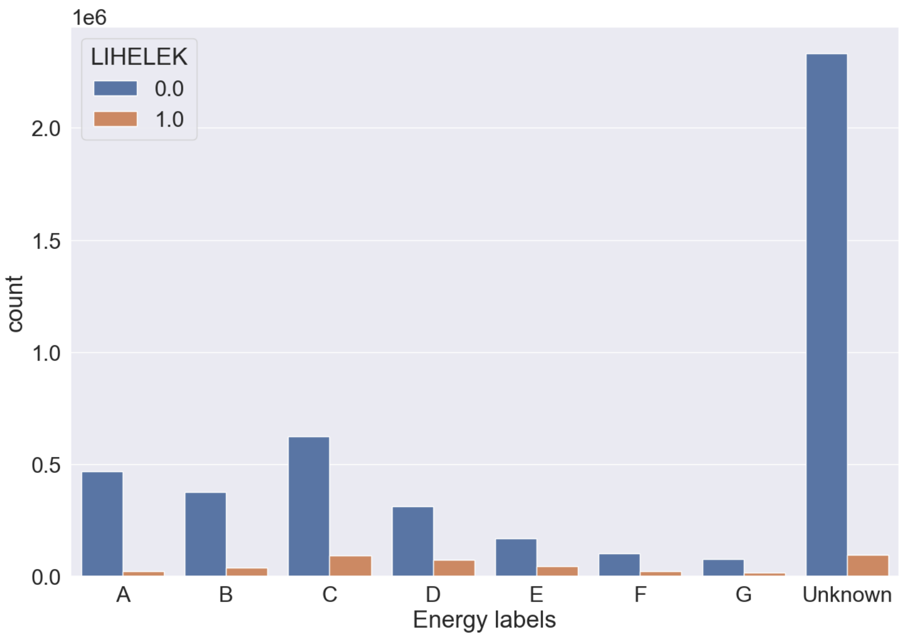
Figure 5.22: Number and Distribution of Energy Poor Households based on Size

As shown in Fig. 5.22(a), single households have the highest proportion of energy poor households, indicating that they are more vulnerable to energy poverty. Additionally, the energy bills of single households are generally lower than other types, which is expected since less number people live together, but they still struggle with affordability issues compared to other household types.

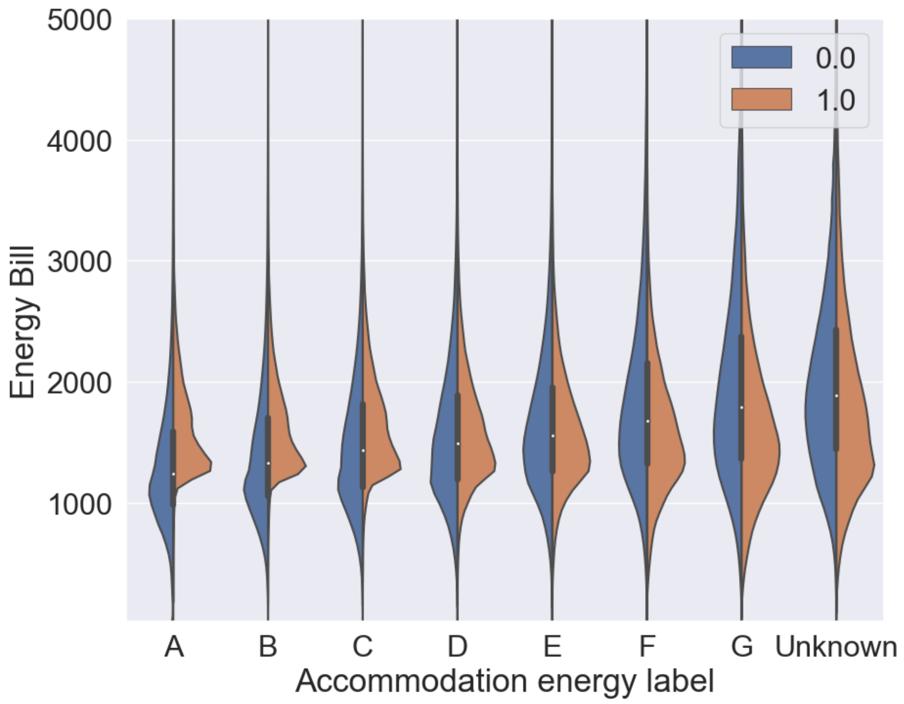
Regarding Fig. 5.22(b), the peak of energy bills for energy poor households is higher than non-energy poor households, except for families larger than 9 members, where the peak has a broader range. This suggests that larger families may face diverse energy consumption patterns and varying affordability challenges.

Figure 5.23 showcases the number and distribution of energy poor households classified according to their accommodation energy labels.

According to the findings in Fig. 5.23(a), the accommodations with energy labels "C" and "unknown" have the highest number of energy poor households, which is not surprising given that they are the two largest groups. However, in terms of relative incidences, classes D and E also show relatively high energy poverty rates. Notably, a significant portion of houses has an unknown energy label, indicating a need for more comprehensive and precise data on energy labels.



(a) Numbers



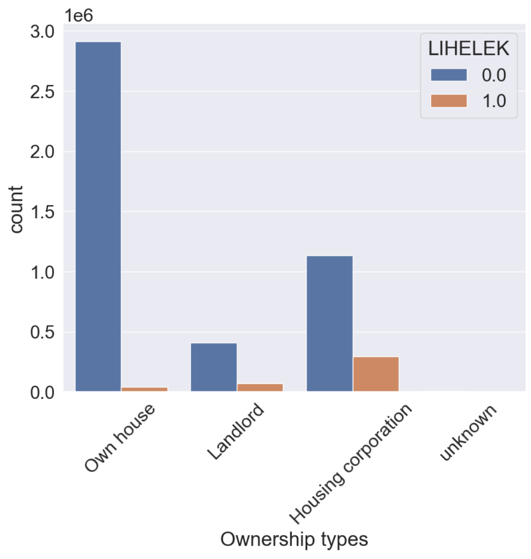
(b) Distribution regarding energy bill (euro)

Figure 5.23: Number and Distribution of Energy Poor Households based on Energy Label

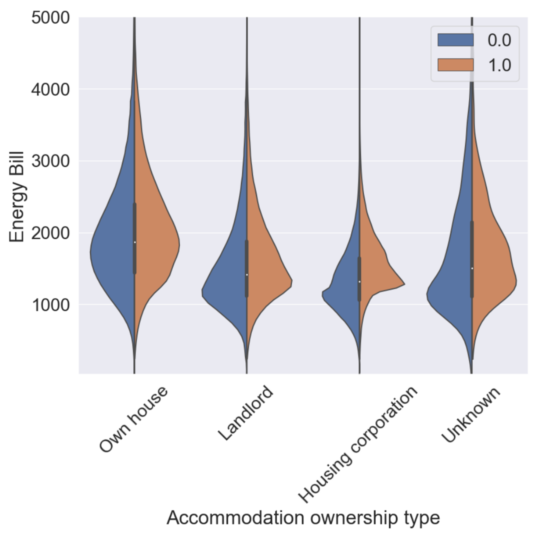
As shown in Fig. 5.23(b), energy poor households exhibit higher energy expenses than non-energy poor households for categories A, B, C, and D, but the difference is not as significant for the other categories. Notably, for the Unknown and G energy labels, the distribution of energy poor households peaks at a lower value compared to non-energy poor households. This observation aligns with the expectation that energy poor households, which are typically low-income households, would not have high energy bills in absolute terms (although their energy quote may be high).

For the better energy labels (A, B, C), it is expected that energy poverty is mainly influenced by the HE (high energy quote) part of the indicator, while for the lower energy labels, it is driven mainly by the LEK (low energy quality of the house) part of the indicator. The violin plots roughly support this interpretation.

Figure 5.24 illustrates the number and distribution of energy poor households categorized according to their accommodation ownership type.



(a) Numbers



(b) Distribution regarding energy bill (euro)

Figure 5.24: Number and Distribution of Energy Poor Households based on Accommodation Ownership Type

According to the data shown in Fig. 5.24(a), accommodations in housing cooperatives have the highest number of energy poor households, making them more vulnerable to energy poverty compared to other ownership types. Approximately 20% of households living in cooperative housing experience energy poverty, while it is 15% for houses with landlords and about 1% for homeowners. This indicates that households in cooperative housing are more likely to face challenges in affording and maintaining their energy needs compared to those in other types of accommodation.

However, in Fig. 5.24(b), it is revealed that accommodations in own houses have the highest peak value for energy bills among all ownership types. This implies that while housing cooperatives may have a higher number of energy poor households, accommodations in own houses face the highest energy bill burden.

Figure 5.25 presents the number and distribution of energy poor households categorized according to their sources of income. The graph displays the amount of energy poverty across different income sources, which is useful for understanding the relationship between the main source of income and energy poverty levels among families.

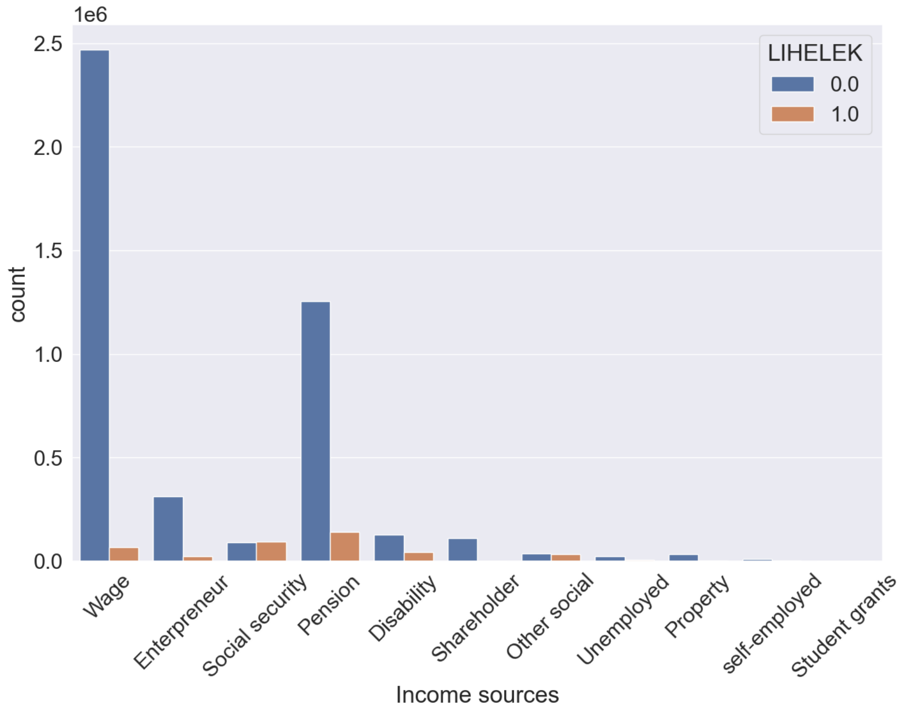
Based on the data shown in Fig. 5.25(a), the highest number of energy poor households is observed among households receiving the first pension, followed by families receiving social benefits from the government (which we call it here social in brief)[1]. This suggests that these two income source categories are more susceptible to energy poverty compared to others.

Furthermore, Fig. 5.25(b) reveals that energy poor households in each income source category tend to have higher energy bills compared to not energy poor households. Specifically, for the pension category, the energy bill peak for energy poor households is higher than that of not energy poor households, indicating a higher energy consumption and potentially greater financial burden on energy costs for these households.

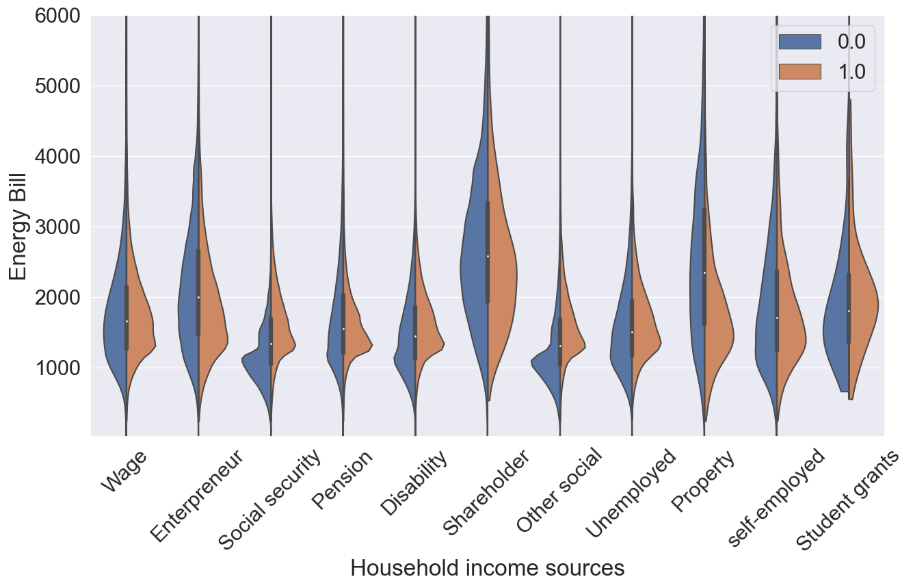
The most common features with the highest predictive power for the machine learning models in this study have been identified. In this section, we have analyzed each feature separately and determined the vulnerability of each category within these features. It is important to note that while individual features provide valuable insights, the true strength of machine learning models lies in their ability to combine and learn from multiple features to make accurate predictions. Therefore, no single feature alone is sufficient to fully cover the predictive power of the models. However, based on the feature selection method, we have identified which features are important and common among the models.

By exploring and understanding the impact of each feature and its categories, we gain valuable insights into the factors contributing to energy poverty. This information can be used to create design strategies and policies to tackle the unique vulnerabilities of various family types. The combination of these features in the machine learning models enhances their predictive capabilities, allowing us to effectively identify energy poor households and develop strategies to mitigate energy poverty.

[1]"Social" refers to Social Assistance Benefit (in Dutch: Bijstandsuitkering), while "Other Social" indicates other types of welfare payments (in Dutch: Uitkering sociale voorziening overig)[10]. More explanation: [43]



(a) Numbers



(b) Distribution regarding energy bill (euro)

Figure 5.25: Number and Distribution of Energy Poor Households based on Income Source

6

Conclusions

This chapter is the conclusion of the thesis, providing a comprehensive summary of the findings and conclusions drawn from the previous chapters. It reflects on the overall work conducted, discusses the limitations encountered during the research, and presents potential areas for future research in the domain of energy poverty prediction.

In this thesis, an original approach is taken to address the lack of machine learning prediction of energy poverty using big data of households in the Netherlands. The study focuses on a multidimensional energy poverty concept, and the research questions aim to assess the accuracy of machine learning models in predicting energy poverty and identify the most influential features for prediction.

To achieve these objectives, two datasets from Statistics Netherlands (CBS) for the years 2019 and 2020 are utilized. Extensive data preprocessing is carried out, followed by feature selection algorithms to identify the features which are most powerful for prediction. The experimental setup for machine learning algorithms is carefully designed, and both models with and without class weights are implemented to handle the imbalanced data. The study then analyzes the results and performs statistical analysis on the common features found among the models.

Throughout the thesis, several key aspects were explored and considered, including:

1. *Socio-economic variables*: The models utilize various socio-economic variables as input features, such as income source, household size, ownership of the accommodation, household reference person age and migration background, and residential type. These variables provide insights into the economic and social characteristics of households and their accommodations which are important factors in predicting energy poverty, and for designing adequate policy measures to mitigate it.

2. *Energy-related variables*: The models also consider energy-related variables, such as energy bills, energy sources, and accommodation energy labels. These variables capture the energy usage patterns and efficiency of households, which are crucial in identifying energy poverty.

3. *Imbalanced data handling*: Since energy poverty is often characterized by an imbalanced distribution of classes, the models employ techniques to handle imbalanced data. This includes considering the class weighting and penalizing wrong predictions on minority class, and the use of evaluation metrics that are suitable for imbalanced datasets, such as the area under the precision-recall curve.

4. *Feature selection*: To enhance the robustness of the prediction models and handle the large datasets effectively, the Recursive Feature Elimination (RFE) method is employed in this study. This feature selection process helps to reduce the dimensionality of the data, improve computational efficiency, and focus on the features that have the most significant impact on predicting energy poverty.

5. *Machine learning algorithms*: Various machine learning algorithms are employed in this study, including decision tree, random forest, logistic regression, XGBoost, and single-layer network (Perceptron). These algorithms provide different modeling approaches, allowing for a comprehensive analysis of energy poverty prediction.

6. *Model evaluation metrics*: The models are evaluated using performance metrics such as precision, recall, and F1-score. These metrics assess the accuracy and effectiveness of the models in predicting energy poverty. Additionally, the area under the precision-recall curve is used to evaluate the models' performance on imbalanced datasets.

7. *Ensemble modeling*: An ensemble modeling approach, specifically hard voting, is utilized to combine the predictions of multiple individual models. This ensemble approach aims to leverage the strengths of different models and improve overall prediction accuracy.

By incorporating these common steps, the models in this study aim to provide insights into the prediction of energy poverty and inform decision-making processes related to energy affordability and efficiency.

The findings revealed that the addition of class weight generally improved recall for all models, indicating enhanced identification of positive instances. However, the random forest model's ability to identify class 1 instances remained relatively unchanged with the addition of class weight. Moreover, ensemble models showed better performance in achieving high precision for detecting energy poor households compared to individual models.

When comparing the models based on their F1-scores, the random forest emerges as the top-performing model. Moreover, the XGBoost and decision tree models demonstrate similar performance, with XGBoost achieving comparable results while utilizing fewer features and shallower trees. This suggests that XGBoost's efficiency makes it an attractive option when considering predictive accuracy and model complexity. Linear models exhibited improved performance with class weight application, suggesting their benefit from the adjustment and potential for further enhancement through hyperparameter tuning.

The investigation of pre-trained models in both same-year and next-year predictions has yielded consistent performance, indicating their potential usefulness for future predictions. However, further research and validation are essential to ensure their reliability and effectiveness. The existence of a pre-trained model opens up possibilities for developing tools that can predict energy poverty using only a few features, offering valuable insights for policymakers and facilitating more accurate scenario analysis. For research purposes, additional exploration is needed to identify proper models and experimental setups to achieve optimal results.

Based on the findings, several recommendations were proposed for future works to enhance studies in energy poverty prediction. First, exploring alternative approaches to address the imbalanced data issue, such as oversampling techniques or cost-sensitive learning methods. This can help improve the performance of the models, especially for minority class prediction. Due to the limitations of the CBS virtual machine, only one method is considered in this study to deal with imbalanced data.

To address the challenge of working with highly sensitive data, data scientists can explore a field that employs synthetic datasets to expand the available data. This approach allows more researchers to work with the data without privacy concerns.

Another potential area of study is to examine households whose energy poverty status changed over two years. By identifying variables that influence these changes, we can better understand the factors impacting energy poverty. Additionally, it would be interesting to compare these variables with the features found in this study as having the highest predictive power.

While this thesis classified households into binary classes as energy poor or non-energy poor, it raises the question of how far each household is from being energy poor. This notion of an "energy poverty gap" can be explored using the definition developed by Croon et al. [44]. Investigating regression machine learning models for predicting the energy poverty gap can provide valuable insights for policymakers to make more accurate investment decisions.

References

[1] Huang Zixi. **Poverty Prediction Through Machine Learning**. In *2021 2nd International Conference on E-Commerce and Internet Technology (ECIT)*, pages 314–324. IEEE, 2021. iii, 8, 9, 11

[2] SAGAR SHARMA. **What the Hell Is Perceptron?** *Medium*, Oct 2019. iii, 11

[3] Sebastian Raschka, Yuxi Hayden Liu, Vahid Mirjalili, and Dmytro Dzhulgakov. *Machine Learning with PyTorch and Scikit-Learn: Develop machine learning and deep learning models with Python.* Packt Publishing Ltd, 2022. iii, 6, 7, 13, 26, 27, 29, 37

[4] Paris Agreement. **Paris agreement**. In *report of the conference of the parties to the United Nations framework convention on climate change (21st session, 2015: Paris). Retrived December*, **4**, page 2017. HeinOnline, 2015. 1

[5] Peter Mulder, Francesco Dalla Longa, and Koen Straver. **Energy poverty in the Netherlands at the national and local level: A multi-dimensional spatial analysis**. *Energy Research & Social Science*, **96**:102892, 2023. 1, 2, 4, 16, 20

[6] Osvaldo Simeone. **A very brief introduction to machine learning with applications to communication systems**. *IEEE Transactions on Cognitive Communications and Networking*, **4**(4):648–664, 2018. 1

[7] Willem van Hove, Francesco Dalla Longa, and Bob van der Zwaan. **Identifying predictors for energy poverty in Europe using machine learning**. *Energy and Buildings*, **264**:112064, 2022. 1, 4, 18

[8] Francesco Dalla Longa, Bart Sweerts, and Bob van der Zwaan. **Exploring the complex origins of energy poverty in The Netherlands with machine learning**. *Energy Policy*, **156**:112373, 2021. 2, 4, 15

[9] Peter Mulder, Francesco Dalla Longa, and Koen Straver. **The facts about energy poverty in the Netherlands; Insights at the national and local level (in Dutch: De feiten over energiearmoede in Nederland; Inzicht op nationaal en lokaal niveau)**. *TNO Report P11678*, 2021. 2, 4, 20

[10] Centraal Bureau voor de Statistiek. **Monitor energiearmoede 2020**. *Centraal Bureau voor de Statistiek*, Jan 2023. 2, 4, 20, 64

[11] Peter Mulder, Anika Batenburg, and Francesco Dalla Longa. **Energiearmoede in Nederland 2022**. *Een actuele inschatting op nationaal en lokaal niveau*, 2023. 4, 16, 20

[12] Authoring Team and Claire Baffert. **Energy poverty and vulnerable consumers in the energy sector across the EU: analysis of policies and measures**. *Policy*, **2**:64–89, 2015. 4

[13] D.H. Wolpert and W.G. Macready. **No free lunch theorems for optimization**. *IEEE Transactions on Evolutionary Computation*, **1**(1):67–82, 1997. 6

[14] Rafael Gomes Mantovani, Tomáš Horváth, Ricardo Cerri, Sylvio Barbon Junior, Joaquin Vanschoren, and André Carlos Ponce de Leon Ferreira de Carvalho. **An empirical study on hyperparameter tuning of decision trees**. *arXiv preprint arXiv:1812.02207*, 2018. 6, 7

[15] Bahzad Charbuty and Adnan Abdulazeez. **Classification based on decision tree algorithm for machine learning**. *Journal of Applied Science and Technology Trends*, **2**(01):20–28, 2021. 6, 7

[16] Sebastian Nowozin, Carsten Rother, Shai Bagon, Toby Sharp, Bangpeng Yao, and Pushmeet Kohli. **Decision tree fields**. In *2011 International Conference on Computer Vision*, pages 1668–1675. IEEE, 2011. 7

[17] Rafael García Leiva, Antonio Fernández Anta, Vincenzo Mancuso, and Paolo Casari. **A novel hyperparameter-free approach to decision tree construction that avoids overfitting by design**. *Ieee Access*, **7**:99978–99987, 2019. 7

[18] Rui Shi, Xinyue Xu, Jianmin Li, and Yanqiu Li. **Prediction and analysis of train arrival delay based on XGBoost and Bayesian optimization**. *Applied Soft Computing*, **109**:107538, 2021. 8, 9

[19] XGBoost. **Notes on Parameter Tuning - xgboost 1.7.5 documentation**. 9

[20] Mirka Saarela and Susanne Jauhiainen. **Comparison of feature importance measures as explanations for classification models**. *SN Applied Sciences*, **3**:1–12, 2021. 9, 12, 13

[21] Rahma Atallah and Amjed Al-Mousa. **Heart disease detection using machine learning majority voting ensemble method**. In *2019 2nd international conference on new trends in computing sciences (ictcs)*, pages 1–6. IEEE, 2019. 9, 10, 12, 28

[22] Philipp Probst, Marvin N Wright, and Anne-Laure Boulesteix. **Hyperparameters and tuning strategies for random forest**. *Wiley Interdisciplinary Reviews: data mining and knowledge discovery*, **9**(3):e1301, 2019. 10, 34

[23] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. **Scikit-learn: Machine Learning in Python**. *Journal of Machine Learning Research*, **12**:2825–2830, 2011. 10, 11, 12, 14, 27, 29, 36

[24] Daud Muhajir, Muhammad Akbar, Affindi Bagaskara, and Retno Vinarti. **Improving classification algorithm on education dataset using hyperparameter tuning**. *Procedia Computer Science*, **197**:538–544, 2022. 10

[25] Jason Brownlee. **Perceptron Algorithm for Classification in Python**. *Machine Learning Mastery*, Dec 2020. 11

[26] Vijay Kotu and Bala Deshpande. *Predictive analytics and data mining: concepts and practice with rapidminer*. Morgan Kaufmann, 2014. 12

[27] Jiaju Miao and Wei Zhu. **Precision–recall curve (PRC) classification trees**. *Evolutionary intelligence*, **15**(3):1545–1569, 2022. 12, 38, 39

[28] Jack Dunn, Luca Mingardi, and Ying Daisy Zhuo. **Comparing interpretability and explainability for feature selection**. *arXiv preprint arXiv:2105.05328*, 2021. 13

[29] Scikit-Learn. **Scikit-learn/forest.py at 0abd95f742efea826df82458458fcbc0f9dafcb2 · scikit-learn/scikit-learn**. 13

[30] Amina Benkessirat and Nadjia Benblidia. **Fundamentals of feature selection: An overview and comparison**. In *2019 IEEE/ACS 16th International Conference on Computer Systems and Applications (AICCSA)*, pages 1–6. IEEE, 2019. 14

[31] Centraal Bureau voor de Statistiek. **Kerncijfers wijken en buurten 2022**. *Centraal Bureau voor de Statistiek*, April 2023. 15, 21

[32] Milena N Rajić, Miroslav B Milovanović, Dragan S Antić, Rado M Maksimović, Pedja M Milosavljević, and Dragan Lj Pavlović. **Analyzing energy poverty using intelligent approach**. *Energy & Environment*, **31**(8):1448–1472, 2020. 17

[33] Zhe Hong and In Kwon Park. **Comparative Analysis of Energy Poverty Prediction Models Using Machine Learning Algorithms'**. *Journal of Korea Planning Association Vol*, **56**(5), 2021. 17

[34] Khizar Abbas, Khalid Manzoor Butt, Deyi Xu, Muhammad Ali, Khan Baz, Sanwal Hussain Kharl, and Mansoor Ahmed. **Measurements and determinants of extreme multidimensional energy poverty using machine learning**. *Energy*, **251**:123977, 2022. 18

[35] Ascensión López-Vargas, Agapito Ledezma-Espino, and Araceli Sanchis de Miguel. **Methods, data sources and applications of the Artificial Intelligence in the Energy Poverty context: A review**. *Energy and Buildings*, **268**:112233, 2022. 18

[36] Centraal Bureau voor de Statistiek. **Gbaperson tab: Personal characteristics of persons in the BRP**. *Centraal Bureau voor de Statistiek*, June 2023. 20

[37] SGOPAL Patro and Kishore Kumar Sahu. **Normalization: A preprocessing stage**. *arXiv preprint arXiv:1503.06462*, 2015. 28, 29

[38] César Laurent, Gabriel Pereyra, Philémon Brakel, Ying Zhang, and Yoshua Bengio. **Batch normalized recurrent neural networks**. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2657–2661. IEEE, 2016. 29

[39] Jie Cai, Jiawei Luo, Shulin Wang, and Sheng Yang. **Feature selection in machine learning: A new perspective**. *Neurocomputing*, **300**:70–79, 2018. 34

[40] Tianqi Chen and Carlos Guestrin. **Xgboost: A scalable tree boosting system**. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016. 36

[41] Qi Qi, Youzhi Luo, Zhao Xu, Shuiwang Ji, and Tianbao Yang. **Stochastic optimization of areas under precision-recall curves with provable convergence**. *Advances in neural information processing systems*, **34**:1752–1765, 2021. 38

[42] Akash A Shah, Sai K Devana, Changhee Lee, Amador Bugarin, Elizabeth L Lord, Arya N Shamie, Don Y Park, Mihaela van der Schaar, and Nelson F SooHoo. **Machine learning-driven identification of novel patient factors for prediction of major complications after posterior cervical spinal fusion**. *European Spine Journal*, pages 1–8, 2021. 38, 39

[43] Centraal Bureau voor de Statistiek. **Documentatie Personen met een uitkering uit overige sociale voorzieningen in de verslagmaand (SECM-SOCVOORZOVMNDBEDRAGBUS)**. *Centraal Bureau voor de Statistiek*, Dec 2021. 64

[44] TM Croon, JSCM Hoekstra, MG Elsinga, F Dalla Longa, and Peter Mulder. **Beyond headcount statistics: Exploring the utility of energy poverty gap indices in policy design**. *Energy Policy*, **177**:113579, 2023. 68

Appendix A

Appendix

A.1 Dataset complete features

Table A.1: All the features, with full description

Category	Feature	Measure (unit)	CBS Name
EP Indicator	LIHELEK	Households with a low income and a high energy bill and/or low energy quality home	LIHELEK
Household Features	Year	Staring year of household composition	DATUMAAN VANGHH1Jan
	Type	Typification of household 1= single, 2= unmarried couples without children, 3= Married couples without children, 4= Unmarried couple without children, 5= Married couple with children, 6= single parent household, 7= other household, 8= institutional, -=unknown	TYPHH1Jan
	Financial assets	Total value of a household's financial assets	VEHW1110FINH
	Mortgage Debt	Mortgage debt related to a household's owner-occupied home	VEHW1210SHYH
	Size	Number of persons that are a household	AantalPersonen PerHuishouden1Jan

Table A.1: All the features, with full description

Category	Feature	Measure (unit)	CBS Name
	Population	Population delimitation of households with perceived income 10= 1 private household; income known, 11= more households, income known, 20= 1 private student household, income known, 21= only student households, income known 22= student and non-student households, income known, 70= more households, income partly known, 99= unknown	WB_POPIIV WONING1Jan
	Income source	Main source of household income 11= Wage, 12, pay director, 13= profit independent entrepreneur 14= other self-employed, 21= unemployment benefits, 22= social benefits, 23= other social benefits, 24= sickness benefits, 25= pension benefits, 26= student grants, 30 = property income, 32= without income, 88= not apply	WB_BBIHJ1Jan
	Disposable income	Household disposable income	WB_BESTINKH 1Jan
	Income percentage	Income relative to the low-income threshold	INHARMLAG
	Standardized income	the disposable income of a household corrected for the size and composition of a household	WB_GESTINKH 1Jan
	Payment budget	The disposable income of a household, excluding the included expenditure components	WB_BETAAL BUDGET1Jan

Table A.1: All the features, with full description

Category	Feature	Measure (unit)	CBS Name
	Payment reserve	Value of the total assets of a household, excluding the value of the owner-occupied home and negatively valued components in the survey year	WB_BETAAL RESERVE1Jan
	Accommodation ownership	The type of an accommodation C= housing corporation, E= own house, O= unknown, V= landlord other than housing corporation	TypeEigenaar1Jan
	Gas Usage	Gas consumption in m3 in the year under review	GAS
	Electricity usage	Electricity consumption in kWh in the year under review	ELEK
	Solar panel usage	The volume of the feed-in of electricity by solar panels	ELEK_ Teruglevering
	City heat usage	Estimated heat consumption in Kj in the year under review	WARMTE
	Allowance	Energy allowance	Recht_op_toeslag
	Energy bill	Energy bill on average energy p rices for customers m2	Energiebedrag
	Energy quote	Energy bill divided by the payment budget	Energiequote
	High energy bill	High energy bill	Hoge_energiequote
	Energy amount	Energy amount usage	LEKbedrag
	Relative energy bill	Energy bill divided by payment budget	LEK_p20
Household Ref person features	Age	Age of the reference person (at the beginning of the year)	LeeftijdRef Persoon1Jan
	Date of birth	Date of birth of the reference person of the household	
	Education	Education of the reference person of the household 1= low education, 2= Mid education, 3= High education	OPNIVSOI2021 AGG4HBmetNIRWO
	Gender	Gender of the reference person 1= Male, 2= Female, - = unknown	GBAGESLACHT

Table A.1: All the features, with full description

Category	Feature	Measure (unit)	CBS Name
	Migration background	Migration background of the reference person of the household 0= Both parents were born in NL, 1= One of parents was born in NL, 2= Both parents born in a foreign country	GBAGENERATIE
Accommodation Features	House value	Value of a house owned by a household	VEHW1121WONH
	Type	Accommodation type 1= residential function, 2= non-residential function, 3= inhabited place, 4= inhabited berth	VSLVoorraad Type1Jan
	Residential type	Typification of the accommodation with residential function 01= detached house, 02= semidetached house, 03= corner house, 04= terraced house, 05= multi-family house, 99= unknown	VBOWoning Type1Jan
	Construction year	Year of the building construction	VBOBouwjaar1Jan
	Area	Usable area of the accommodation in square metres	VBOOppervlakte 1Jan
	Residential/not	Permitted functional use of the accommodation for living 0= no, 1= yes, - = no accommodation object	VBOWoonFunctie 1Jan
	Value	Value of immovable property by a municipality in term of real states	WOZWaardeObject BAG1Jan
	Energy label	Registered energy label 1= A, 2= B, 3= C, 4= D, 5= E, 6=F, 7= G, 9= unknown	Energielabel
Geographical Features	Municipality code	Municipality code of the accommodation	gm_naam
	District code	District code of the accommodation	ste_mvs
	Neighbourhood code	Neighbourhood code of the accommodation	GWBCODE2022

A.2 Features ranking

Table A.2: Features Ranking for all models

Labels	**DT**	**XGBoost**	**RF**	**LR**	**Single-layer network**
hh_assets	1	1	1	2	1
hh_mortgageDebt	9	3	2	1	1
ref_age	1	6	1	1	7
hh_size	1	1	1	1	1
acc_constructionYear	1	4	1	1	11
acc_area	1	13	1	1	1
acc_value	1	2	1	1	2
energy_bill	1	1	1	1	1
move_date_y	2	14	1	9	1
hh_type	8	11	4	8	1
hh_population	13	17	12	13	12
hh_incomeSource	1	1	1	1	5
residential_type	3	7	3	10	3
Gender	5	5	6	6	1
migration_status	6	9	8	7	8
acc_energyLabel_encoded	1	1	1	1	1
acc_ownershipType	1	1	1	1	1
acc_solarPanel	10	8	10	3	4
acc_Heat	12	10	11	4	1
acc_urban_level	4	12	5	11	6
acc_urban	11	15	9	12	10
acc_high_energyLabel	7	16	7	5	9