Smart House: A Digital Twin

Project III

Degree in Data Science (GCD) - Academic Year 2023/2024

Javier Luque, Daniel Garijo, Ángel López, Andrea Sánchez, Claudia Martínez and Pablo Parrilla

Presentation

This project has been carried out by students from the ETSINF (Escola Tècnica Superior d'Enginyeria Informàtica) of the Universitat Politècnica de València. Specially, it has been developed in the subject of Project III, of the second semester of the third year of the "Data Science" Degree. During the quarter, our team has been working on this study about a smart home. This work has resulted in this report with a series of elaborate analyzes that will be presented throughout this document.

The dataset used in the project has been the result of having observed and modified several datasets collected over time, as will be explained later.

2030 Agenda Objectives

Speaking about the *"Sustainable Development Goals of the 2030 Agenda"*, this project can be convenient for *SDG 9*, in order to build sustainable and innovative infrastructures, as well as to promote responsible energy consumption and reduce possible unnecessary expenses in accordance with *SDG 12*. Lastly and perhaps most notable, *SDG 11*, sustainable cities and communities, may be the most represented given the essentiality of the project.

Authors

This team is made up of six members: Ángel López, Javier Luque, Daniel Garijo, Pablo Parrilla, Claudia Martínez and Andrea Sánchez, who, as we have already mentioned before, will be in charge of this project about the Smart House, and of responding to the different questions that arise.

So that, working as a team can make all the difference when trying to get something done. That is why our team is made up of a group of talented and diverse people, each of whom brings something unique to the table. We have different skills and experiences, which means we can look at problems from all kinds of angles and find creative solutions. Everyone has their own strengths and together we can create an innovative and supportive environment. This section will talk about who we are, what we do best, and how we work together to achieve our goals.

Acknowledgment

This project has not had much collaboration or support from external entities. However, we must thank our tutors, Sonia Tarazona Campos, José Alberto Conejero Casares and José Hernández Orallo, the follow-up, guidance and help that they have given us on numerous occasions throughout the development of the work.

In addition, we must undoubtedly thank the Universitat Politècnica de València, which has offered us software (Microsoft Office 365), teaching and Internet connection, as well as the creators of the web pages who have offered us information and to the creators of the databases, as they have been the main driving force of this project and, without them, this work would have been doomed to failure.

Abstract

The "Smart House: Digital Twin" project aims to optimize energy consumption and reduce costs in smart homes using advanced statistical and machine learning techniques. Comprehensive data on appliance usage and weather conditions were used to build predictive models. The SARIMA and SARIMAX models capture seasonal patterns and incorporate climatic variables, enhancing prediction accuracy. Machine learning models such as DecisionTreeRegressor and advanced models like RandomForestRegressor and GradientBoostingRegressor were explored. The final models, including an AdaBoostRegressor and a StackingRegressor, demonstrate robust performance in predicting energy consumption. Additionally, a user-friendly Streamlit web application provides real-time predictions and cost estimations, enabling users to manage energy more effectively. This innovative solution not only aids in reducing energy bills but also promotes sustainable living practices.

The six most used words are: model, energy, consumption, prediction, data, and regression.

Index

List of Figures

1 Introduction

In this *"Smart House: Digital Twin"* project, the intention is to delve into the world of smart homes with the aim of optimizing energy consumption and reducing costs. To achieve this, it has been gathered comprehensive data on appliance usage and weather conditions.

Following a meticulous process of data cleaning and preparation, it has been employed advanced statistical and machine learning techniques, including regression, to build predictive models. Regression techniques, whether analyzing trends, time series, or predicting the values of a continuous variable from the evolution of another continuous variable (generally time in time series) or from other continuous or nominal variables (general regression), have been pivotal in this process. Now, we embark on the evaluation phase to ensure that these models fulfill our primary purpose: **enhancing energy efficiency in smart homes**.

2 Data Overview

The data used in this project comes from a database containing the kilowatts consumed by each appliance in a house, recorded every minute over a year. The database consists of a total of 503,911 rows and spans across 32 columns.

Before discussing the structure of the CSV file (from which the information for the analyses is extracted), it's important to mention how this specific CSV was chosen. After comparing various datasets and based on references about smart homes on *"medium.com"* it was concluded that valuable insights and information could be obtained from working with this data. [5]

However, it is important to note that it cannot be assured that this dataset is open data because the license on Kaggle states *"unknown"*, and therefore, we cannot confirm its open data status.

The CSV file used pertains to a smart home with appliance consumption (in kW) recorded by a smart meter and the weather conditions of that region. The columns in this dataset contain values about: a *"time"* column with the hour and minute along with the day, month, and year (where each row differs by a 1-minute interval); the *"House overall"* column explaining the total energy consumption; and the *"gen"* column, which details the energy generated through solar energy or other sources.

Focusing on the energy consumed by the appliances specifically, data is available in columns such as: *"Dishwasher"* for dishwasher consumption; *"Furnace 1"* and *"Furnace 2"* for the furnaces; and *"Fridge"* for the refrigerator. Other relevant columns in the dataset include *"Garage door"* and *"Microwave"*.

Additionally, data on specific locations in the house and their total consumption are available, such as: *"Kitchen 12"*, *"Kitchen 14"*, and *"Kitchen 38"* representing the kitchens; *"Barn"* for barn consumption; *"Well"* for well consumption; *"Living room"* for energy expenditure in the living room; and *"Wine cellar"* for consumption in the wine cellar.

It should be noted that all columns referring to appliances and specific locations in the house contain real values measured in kW.

The columns dealing with climatological and temporal variables include: *“Temperature”*, measuring temperatures in degrees Celsius; *“Icon”*, overall weather condition; ”Humidity”, percentage of humidity; *“Visibility”*, visibility percentage from 0 to 10; *“Summary”*, summary of sky conditions; *“ApparentTemperature”*, felt temperatures; *“Pressure”*, measured in Pascals; *“Windspeed”*; *“Cloudcover”*, presence of clouds; *“Windbearing”*, wind angle; *“Precipipointensity”*, precipitation intensity; *“Dewpoint”*, dew point temperature; and *“precipprobability”*, probability of precipitation.

3 Data Preprocessing

The first modification that had to be made to the database was the correction of the date. The column containing this information was in UNIX format, so it was necessary to make the modification to properly handle the dates. Thus, a Python script [1] was used to replace the values in the column with the actual dates. These values were referenced from several articles and the Kaggle repository from which the data was extracted.

This state of the database, named *“data1.csv”*, was used to develop different predictive models the SARIMAX model. From here, three additional transformations were implemented to prepare the data for machine learning training.

First, an inconvenient amount of possible values for the categorical variable “*summary*” was found through the exploratory data analysis. Therefore, a renaming of these values was made to reduce the possible values from 18 to 5.[2]

Next, for each appliance or room (except the fridge) in the expenses variables, a new binary variable is created to represent whether this appliance or room is active. An important note on this transformation is that, in the original data, there were two furnaces and three kitchens, which have been combined (addition of consumption) to be dealt as one furnace and one kitchen. To create the binary variables, a threshold is established for each appliance or room and if the expense exceeds this limit, the corresponding “*ON*” variable will be 1, and otherwise 0.[3]

Finally, time variables are created from the date variable, resulting in the addition of the variables “*Mes*”, “*Dia*”, “*Hora*” and “*Minuto*” (month, day, hour and minute, respectively). Furthermore, the variables “*Mes*” and “*Hora*” are one-hot encoded.[4]

This final version, named “*data4.csv*” is the one used for Machine Learning models training.

[1]Date transformation
[2]*Summary* variable transformation
[3]Creation of *ON* variables
[4]Creation of dummy time variables

4 Predictive Models and Machine Learning Implementation

To predict and simulate household energy consumption, two approaches were used: daily data aggregation for time series analysis with SARIMA and SARIMAX models, and hourly data aggregation to train machine learning models such as DecisionTreeRegressor and RandomForestRegressor. These methods captured seasonal patterns, integrated climatic variables, and optimized both accuracy and computational efficiency. Additionally, simulations of different weather scenarios were performed using the SARIMAX model to evaluate their impact on energy consumption.

4.1 Models for Time Series

Different predictive models have been trained with the intention of forecasting daily electricity consumption. We have explored multiple models, optimized their parameters and chosen appropriate evaluation metrics. This process uses various Python packages and tools for hyperparameter optimization and automation of iterative improvements.

4.1.1 Premodel Analysis

Before proceeding to the modeling prototype, it is necessary to have a deeper understanding of the data. Thus, an analysis using correlations has been conducted to identify which variables may cause fluctuations in energy consumption.

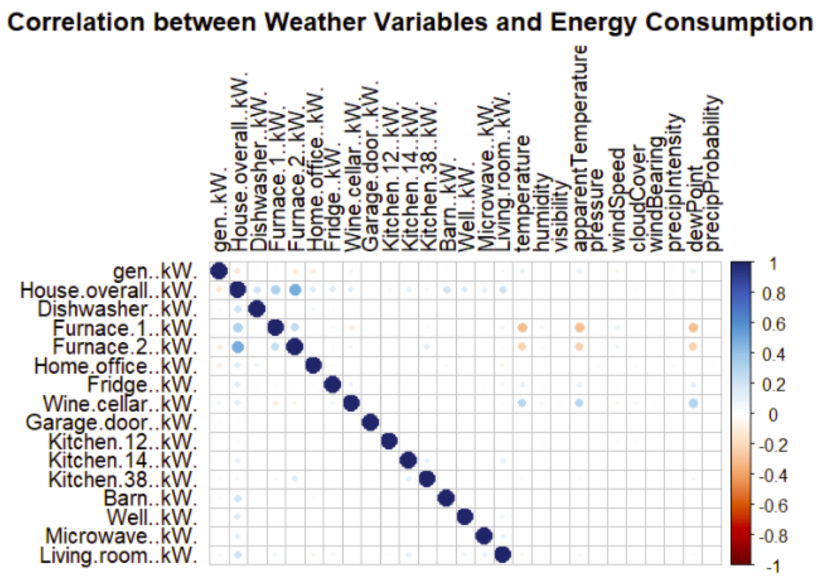

Figure 1: Correlation Weather Variables and Energy Consumption

At first glance, strong correlations between weather variables and device energy consumption have not been observed, beyond slight correlations between temperature and the consumption of the cellar or oven (in this case inverse). Although it is true that it seems that higher energy expenditure occurs during warm periods than in cold seasons. In addition, the graph evidence that the more the furnaces are used the higher the overall energy expenditure is. This indicates that the oven accounts for a significant percentage of the household's electricity consumption, and for future savings on bills, focusing on the use of this appliance could be particularly important.

4.1.2 SARIMA model

Firstly, an estimated prediction of energy expenditure is sought without considering weather conditions to the model. Therefore, the model will attempt to predict the total energy consumption of the house based on historical data (variable 'House.overall..kW.').

In this case, to serve as a basis for further iterations and deepening of the model, an attempt will be made to predict 15 days of the already studied period to check the operation and corroborate the potential of the model. The study is chosen to be conducted on a daily basis in order to reduce noise and capture daily consumption patterns effectively. Moreover, conducting it on a weekly basis would greatly reduce the sample and complicate the development of the model, while conducting it hourly would increase costs and probably a prediction on an hourly basis would not be more useful to consumers. Therefore, this specific number of days has been chosen to balance both the effectiveness and usefulness of the model.

After data preparation, the dataset is divided into a training set and a test set, which will be used to verify the accuracy of the model. Since it is a time series, cross validation is not possible. However, a SARIMA (Seasonal Auto Regressive Integrated Moving Average) model has been chosen to perform the task, especially effective in contexts where data exhibit trends and seasonal patterns, as is common in energy consumption records.

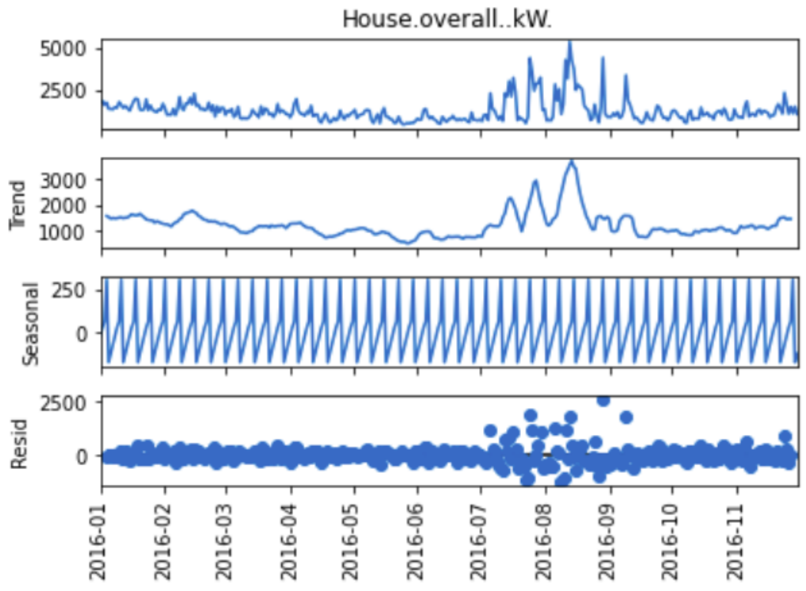

Figure 2: Seasonal decomposition for House Overall (kW)

Seasonal decomposition was performed to understand the underlying patterns in the data. It revealed a weekly seasonality, which informed our choice of seasonal components in the SARIMA model. In addition, it can be observed that the trend chart illustrates a noticeable upward trend, particularly during the summer months, thus confirming the positive correlation of the energy consumption for some appliances with temperatures.

The auto_arima function from the pmdarima library have been employed to automate the process of selecting the best SARIMA model parameters. This function iteratively tests different combinations of ARIMA, and seasonal parameters and selects the model with the best performance based on the Akaike Information Criterion (AIC).

The selected model has been ARIMA(1,0,0)(0,0,2)[7], indicating one autoregressive term, without moving average term, and a seasonal autoregressive component of order 3 with a seasonal moving average component of order 1, considering weekly seasonality. Additionally, it includes an intercept term. The use of auto_arima automates hyperparameter optimization in the modeling workflow, ensuring that the chosen model parameters are optimal.

After fitting the model to the training data, the following predictions for the last 15 days are obtained:

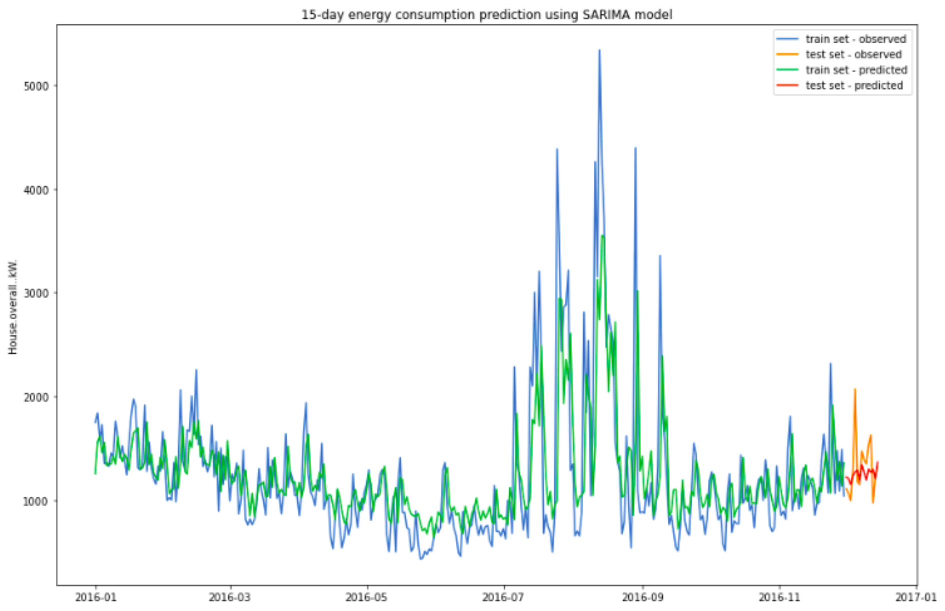

Figure 3: SARIMA predictions: Last 15 days

This line graph compares the actual and predicted energy consumption by SARIMA model for both training and test data sets. The x-axis represents the time period, and the y-axis represents energy consumption in kilowatts (kW). We can see that the model effectively captures the patterns in the training data, while there is a higher error for the test data.

There are some more discrepancies between the real and predicted lines for peak consumption periods, maybe caused by factors like extreme weather events or unexpected changes in consumption patterns. The prediction generally follows the trend of the test real line. However, we deduce the model is overfitted due to the small amount of samples.

4.1.3 SARIMAX model

After analyzing the ARIMA model, the next step is to iterate on the baseline model to improve the effectiveness of the results. The hyperparameters have already been optimized. However, further improvements can be made by adding new variables that influence the prediction, such as climatic variables. Therefore, a SARIMAX model will be used to predict household energy consumption, considering both the time series nature of the data and the influence of external climatic variables.

One of the primary strengths of the SARIMAX model is its ability to incorporate exogenous variables, which are external to the target variable but can influence it significantly.

In our context, these exogenous variables include various climatic factors such as temperature, humidity, and other weather conditions.

The energy consumption in a household is not only dependent on past consumption patterns (captured by SARIMA) but is also heavily influenced by external climatic conditions. As higher temperatures may lead to increased use of air conditioning, thereby increasing energy consumption. By including these climatic variables in the model, SARIMAX leverages additional information that directly impacts energy usage, leading to more accurate and reliable predictions.

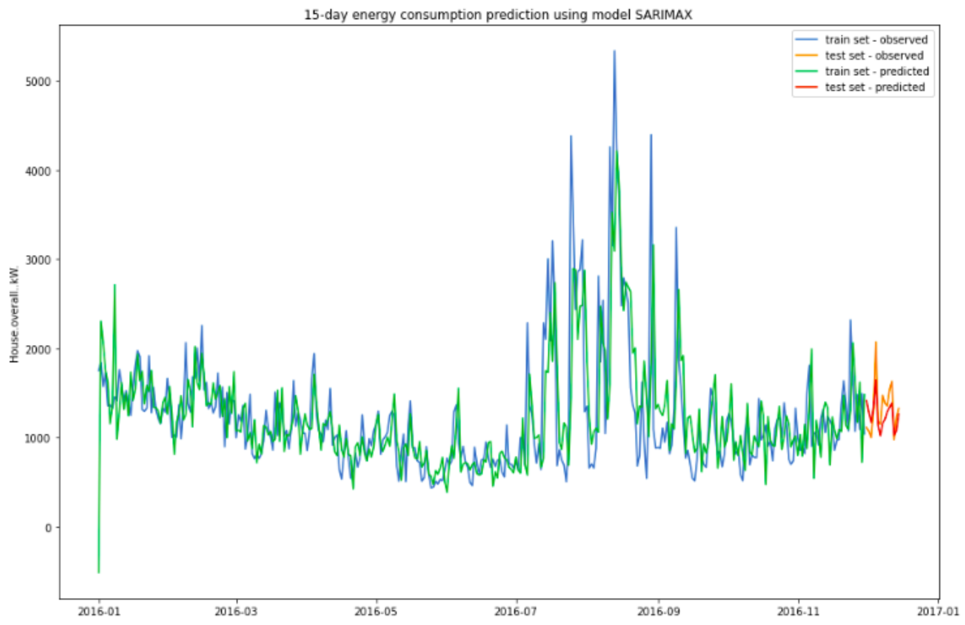

Figure 4: SARIMAX predictions: Last 15 days

The superior performance of SARIMAX can be attributed to its comprehensive approach, where it not only learns from past data but also adjusts its predictions based on current external conditions. Making it a robust forecasting tool for our purpose.

By predicting energy consumption more accurately, smart home systems can optimize energy usage, reduce waste, and lower costs. Particularly important in the context of increasing energy prices and the need for sustainable living practices.

It can also be useful to aid in predictive maintenance of household appliances. By understanding how external factors influence energy consumption, the system can anticipate when appliances are likely to be overused or stressed, prompting timely maintenance and preventing breakdowns.

4.2 Machine Learning Models

For training machine learning models, the last preprocessed dataset have been used ("*data4.csv*"). In addition, slight tranformations have been made to deal with the two categorical features. On one hand, "*summary*" has been one-hot encoded; whilst "*icon*" has been directly remove, given that it provides similar information to the first variable.

Then, the data has been split into three different arrays: one for the dates, one for the independent variables, and a last one for the variables that are going to be predicted. Lastly, the features are standardize [5].Everything is now set and ready to start looking for the best model.

4.2.1 Hyperparameters Search for Weak Models

To identify the best model for the regression problem, a GridSearchCV was conducted using the scikit-learn library. This process involved testing various models, including different linear models such as linear regression, ridge regression, and lasso regression, as well as SVR (Support Vector Regressor), DecisionTreeRegressor, and MLPRegressor (Multi-Layer Perceptron Regressor). By performing GridSearchCV, different hyperparameters for each model were systematically explored to identify the optimal combinations that yield the best performance. This comprehensive approach ensures that each model is fine-tuned to its best possible configuration, thereby improving the accuracy and robustness of the predictive capabilities.

The results of this hyperparameter search indicated that the best performers were the DecisionTreeRegressor, SVR, and MLPRegressor. However, the MLPRegressor often failed to converge and had a high computational cost without significantly outperforming the other two models. Subsequently, a new hyperparameter search was conducted for the SVR and DecisionTreeRegressor, this time incorporating a custom loss function. This function penalized low predictions more heavily than high predictions because the objective is to predict the energy cost of a house. In this context, it is more important to overestimate rather than underestimate the cost to ensure sufficient budget allocation and avoid potential shortfalls in energy expenditure.

The latest results obtained with the customized loss function show comparable performance to previous findings. Notably, the DecisionTreeRegressor demonstrates superior performance over SVR, although the difference is not significant enough to definitively exclude SVR from consideration among the advanced models and ensembles under evaluation. While the DecisionTreeRegressor's outperformance is notable, the relatively minor discrepancy suggests that SVR remains a viable option for further exploration in subsequent training iterations involving advanced models and ensembles.

4.2.2 Hyperparameter Tuning for Advanced Models

Following the initial exploration and tuning of several models, the focus shifted to more sophisticated models to further improve the accuracy of predicting the energy consumption. In this new phase a GridSearchCV is carried out on advanced models, including RandomForestRegressor, GradientBoostingRegressor, XGBRegressor, and LGBMRegressor. These models are known for their high performance and ability to handle complex datasets effectively. Once again, the custom loss function was used.

[5]Preprocess for Machine Learning

The results indicated that all these models provided good outcomes, with RandomForestRegressor standing out in particular. Within the boosting family, performance improved with the model's complexity. XGBRegressor and GradientBoostingRegressor demonstrated strong performance, while LGBMRegressor lagged slightly behind. Notably, GradientBoostingRegressor performed competitively with XGBRegressor, offering a comparable performance while being less computationally expensive. This balance between performance and computational cost makes GradientBoostingRegressor an attractive option for efficient and accurate energy cost predictions.

4.2.3 Final Selection of ML Models

Various combinations of models and ensembles were tested, leading to the selection of two final models: an AdaBoostRegressor with a DecisionTreeRegressor; and a StackingRegressor incorporating XGBRegressor, SVR, and GradientBoostingRegressor as base models with a RandomForestRegressor as the final estimator. The second selection slightly outperforms the first one, but it is nothing significant enough, especially keeping in mind that the computational cost of the first model is much better than the second.

The first model, an AdaBoostRegressor, utilizes a DecisionTreeRegressor with a 'friedman_mse' criterion, a maximum depth of 20, a minimum of 10 samples per split, and a random splitter. This model includes 40 estimators, making it a computationally lighter option while still providing robust performance. The choice of a DecisionTreeRegressor as the base estimator in AdaBoost allows for capturing non-linear relationships in the data, which can be really important for accurate energy consumption predictions.

The second model is a more robust StackingRegressor. It combines the strengths of XGBRegressor, SVR, and GradientBoostingRegressor as base models, with a RandomForestRegressor serving as the final estimator. The selected hyperparameters for the XGBRegressor include a colsample_bytree of 0.8, a learning rate of 0.1, a maximum depth of 5, a min_child_weight of 5, 200 estimators, and a subsample of 0.6. The SVR uses a polynomial kernel with a C value of 100, and the GradientBoostingRegressor has a maximum depth of 4 and 200 estimators. The final RandomForestRegressor has a maximum depth of 30, a minimum of 2 samples per leaf, a minimum of 5 samples per split, and 300 estimators. This ensemble approach ensures superior performance by leveraging the diverse strengths of each base model, making it particularly well-suited for capturing complex patterns. The catch: it's computational cost. It cannot be trained quickly unless access to high-performance computing resources is available, which may limit its practicality in real-time applications or environments with limited computational power.

Both models offer distinct advantages: the AdaBoostRegressor is computationally efficient, making it broadly applicable, while the StackingRegressor, although more computationally intensive, delivers slightly superior performance, making it ideal for scenarios where the highest accuracy is required.

5 Models Evaluation

5.1 SARIMA

To assess the adequacy of the ARIMA model, we will measure it using RMSE, a standard metric that provides insight into the average magnitude of the prediction errors. This allows us to understand the error in the same units as the time series and to give more weight to larger errors, which is ideal for our case where energy consumption peaks are critical. We have achieved an RMSE of 266.56 kW for the SARIMA model on the test set and a mean Absolute percentage error of the 13.37

Due to the limited amount of data, it is highly probable to find overfitting in the model. We have applied a statistical measure, specifically the cumulative distribution function (CDF) of the F-distribution. The CDF value indicates the probability that the ratio of the mean absolute errors (MAE) between the training and test sets is less than or equal to the observed ratio, under the null hypothesis that the variances are equal. Having a result of 0.85 (greater than 0.5), it can be inferred that the model generalizes well.

5.2 SARIMAX

This SARIMAX model demonstrate a lower RMSE and MAPE compared to a standard SARIMA model. The RMSE of the model decreases from 266 kW to 179 kW, while the MAPE drops to below 10% (9.9%). This metrics' reduction reflects a lower average deviation between the predicted and actual values, validating the efficacy of incorporating exogenous variables.

5.3 ML Models

To ensure the robustness and reliability of the selected models, various validation techniques were employed. Initial validations involved simple "by-hand" tests to get a quick sense of model performance. However, the primary and more rigorous validation was performed using 10-fold cross-validation. This method divides the dataset into ten parts, training the model on nine parts and testing it on the remaining one, and repeating this process ten times. The results are then averaged to provide a comprehensive evaluation of the model's performance.

5.3.1 Boosting Model Validation

For the AdaBoostRegressor with DecisionTree, the validation results were:

- **RMSE (Root Mean Squared Error)**: 16.3795
- **MAPE (Mean Absolute Percentage Error)**: 20.5333%
- **Score**: 0.8842
- **R2-Score**: 0.8638

These results indicate that the AdaBoostRegressor performed well, with a relatively low RMSE showing that the average prediction error was small. The MAPE of 20.5333% suggests that the model's predictions were reasonably accurate. The high Score of 0.8842 and R2-Score of 0.8638 demonstrate that the model explained a large proportion of the variance in the data. Overall, the AdaBoostRegressor provided reliable and accurate predictions, making it a strong candidate for predicting energy domestic consumption, particularly when computational efficiency is a priority.

5.3.2 Stacking Model Validation

For the StackingRegressor, the validation results were as follows:

- **RMSE (Root Mean Squared Error)**: 16.608
- **MAPE (Mean Absolute Percentage Error)**: 21.3889%
- **Score**: 0.878
- **R2-Score**: 0.8677

The StackingRegressor also performed well, with an RMSE of 16.608 indicating slightly higher average prediction errors compared to the AdaBoostRegressor. The MAPE of 21.3889% was a bit higher, suggesting the model's predictions were slightly less accurate on average. However, the Score of 0.878 and R2-Score of 0.8677 were both high, showing that the model effectively explained the variance in the data. The StackingRegressor's robustness and use of multiple algorithms provide a balanced approach to capturing complex patterns in the data, making it a valuable option when the highest accuracy is desired, despite its higher computational cost.

While boosting may outperform in certain metrics, stacking offers different advantages that could lead to superior performance in practical scenarios. Stacking's ensemble approach combines predictions from multiple base models, potentially erasing individual model weaknesses and reducing the risk of overfitting. This adaptability allows stacking to capture a wider range of features and relationships within the data, enhancing its ability to generalize to unseen data. Additionally, stacking's robustness and versatility make it well-suited for handling complex datasets with nonlinear relationships and outliers. Therefore, despite boosting's strengths, stacking's capacity to blend diverse model predictions and its resilience to overfitting positions it as a compelling choice for predictive modeling tasks where stability and generalization are critical.

The validation results demonstrate that both models perform well, with the AdaBoostRegressor having a slight edge in terms of lower error rates and higher scores. The choice between the two models may depend on the specific application's computational constraints and the need for slightly better performance versus computational efficiency. The AdaBoostRegressor is ideal for scenarios requiring quicker training and prediction times, while the StackingRegressor offers superior robustness and accuracy for applications where computational resources are less of a concern.

6 Application Deployment

Already mentioned the Machine Learning models and their evaluation, it's time to talk about the application deployment of them, where we can find a *"Simulation"* and a *"StreamLit Application"*:

6.1 Simulation

One of the objectives of the project was to perform simulations that would allow the study of "What If" scenarios. To this end, the same Python library (statsmodels) with which the SARIMAX model was developed was used, as it allows for quick and easy simulations.

For these simulations, data aggregated by days were used. In this case, only the numerical climatic variables and the target variable, *"House..Overall.kW,"* were used. For the aggregation, the target variable was summed, while the climatic variables were averaged, and with these, the SARIMAX model was calculated.

Once the different scenarios were chosen, the las fifteen days were used as the basis to start the simulation to maintain some continuity, using the mean and standard deviation of the predictor variable in that set.

Finally, several functions were defined to generate data according to the scenarios defined, adding some randomness using the *"Numpy"* library.

6.1.1 Results

For the simulations, four different scenarios were simulated: mild winter, a colder winter than usual, high wind days and stormy days with low visibility.

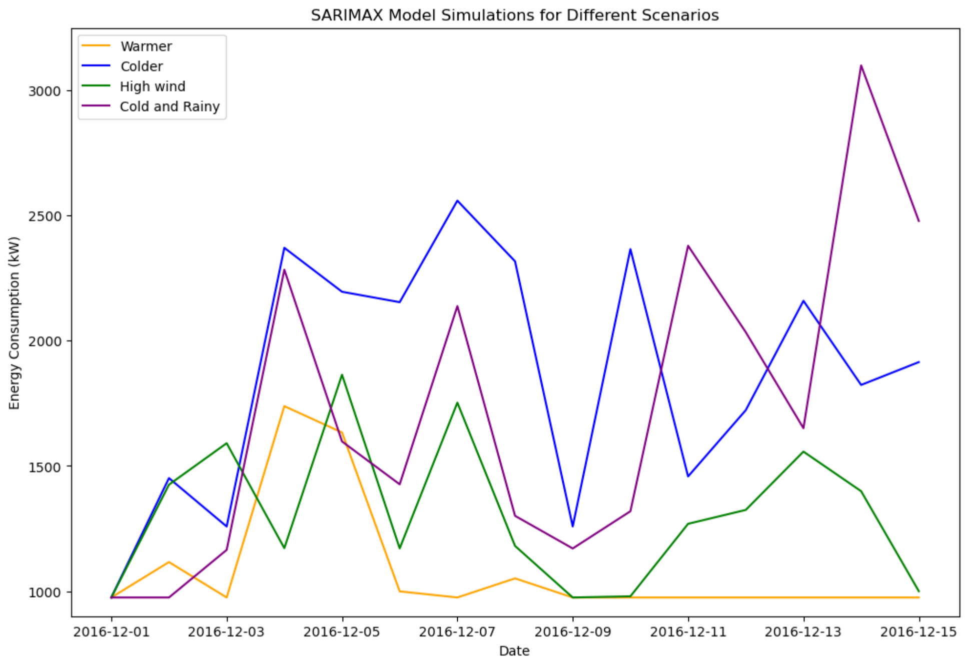

Figure 5: Simulation Results

In this figure, the simulation results of the different scenarios can be seen. When interpreting this data, it is important to consider the randomness introduced to make the data more realistic.

Firstly, a warmer winter is highlighted in orange. The most notable aspect of this simulation is that there is lower House Overall most of the observed days, with the exception of the fourth and fifth days. Additionally, there is generally very little variability, coinciding with the low House Overall.

On the one hand, the simulation of a colder winter, where temperatures have dropped suddenly, can be observed. During the first few days, House Overall does not increase much, but from the fourth day onwards, it increases considerably.

On the other hand, in the scenario of higher winds, House Overall increases on some days, but it is not as considerable an increase as in the second or fourth scenario.

Finally, in the scenario where temperatures and visibility drop and the probability of rain increases, House Overall generally increases considerably, with a peak in the last few days.

6.2 Streamlit Application

From the two best performer machine learning models that have been reached, the Boosting with a DecissionTreeRegressor is selected to be deployed in this web application, available in `https://smarthouse-proyiii.streamlit.app`. This is because it is a much quicker model, which is an essential characteristic to make the user experience better. In this web application, the user can choose a period of time from 01/01/2016 (00:00) to 15/12/2016 (22:45). What this application does is use the period selected by the user as the test data, and all the other observations will be used to train the model.

Once the model is trained and the predictions are made, the web application displays a graphic that compares the real values with the predicted ones like shown in Figure Streamlit Application. In order to improve visualization, if the period selected contains more than 125 hours, 75 folds are created and the values inside each fold are added, showing therefore only 75 point in the graffic. The "*Aggregation*" label avobe the graphic indicates how many hour are added in each point.

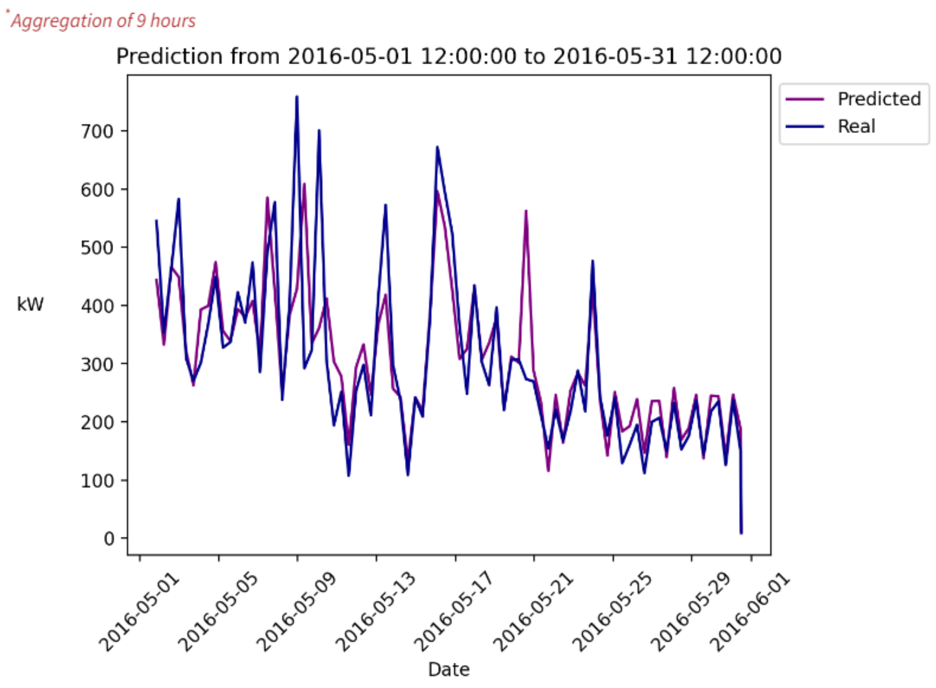

Figure 6: Streamlit Application

Below the graphic, the user is given the choice of select multiple evaluation metrics for regression, which will be shown in a card. By default, the RMSE and the MAPE are shown.

Once the user has selected the dates, the web application will make a call to the API of the area where the house is located. In this case, and as an example, **the API of the Spanish Electricity Network** has been used for the execution.

The first step will be to format the dates as a string. With that done, the Python Requests library is imported to make the API call. After that, the determined parameters are given to the program to receive a response in .json format. Finally, the program will unpack the prices and save them for later use.

With these prices for each hour, the application multiplies by the values predicted and displays two graphics:

- Figure Predicted Cost. Shows the spending in euros for each hour (or the aggregation explained before).
- Figure Accumulated Predicted Cost. Shows the increase in total spending throughout the period.

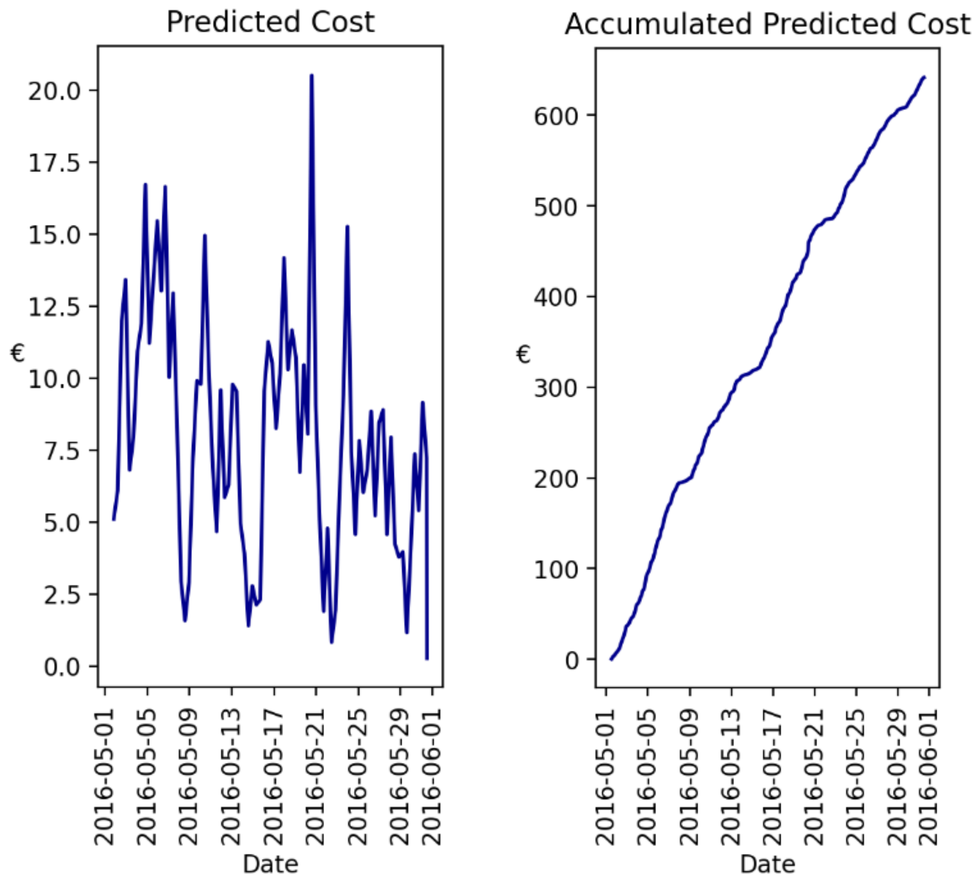

Figure 7: Predicted Costs

Finally, other card is shown with the total expected spending in euros.

The goal of this application is to provide an easy-to-use interface for the user, that will allow them to find out its expected spending in a quick way.

7 Legacy

7.1 Data, code and reports available after the project

To begin with the legacy of the project, both the data and the code are available. The original data can be found in the same Kaggle repository from which they were extracted [2], while all the code and the intermediate datasets used are available in the GitHub repository [1]. In addition, the reports and the papers will be uploaded to the Github so they will be available for everyone.

7.2 New tools and knowledge adquired during the process

Regarding the tools and knowledge acquired, there has been notable learning in data processing. The available data were recorded by the minute, necessitating an aggregation process complicated by the different natures of the features of the data. From this, skills in handling data of this nature to adapt them to the required models were developed.

Additionally, new tools such as the library that enables the use of the SARIMAX model were learned. Furthermore, the process of conducting a simulation with this library and generating new "What If" scenarios using it is now understood.

7.3 Final cost and energy impact of the project

The project aims to optimize energy consumption in smart homes through the use of advanced statistical analysis and machine learning techniques. The energy impact of the project is significant, as the implementation of predictive models can lead to a considerable reduction in household energy consumption. By improving energy efficiency, the project contributes to reducing the carbon footprint and greenhouse gas emissions, thereby supporting environmental sustainability and the fight against climate change.

Regarding the final cost of the project, most of the resources used came from the Universitat Politècnica de València, which provided software (such as Microsoft Office 365), teaching, and internet access. The support from the tutors was also crucial and did not involve any additional costs. However, the main cost lies in the time and effort dedicated by the students and tutors in data collection, cleaning, and analysis, as well as in the development of the models and the implementation of the proposed solutions.

7.4 Who really benefitted from the project

The main beneficiaries of the project include several key groups. Firstly, we as students benefit from participating in this project, where we acquire valuable knowledge and skills in optimizing energy consumption in smart homes. Through this process, we have learned advanced techniques in statistical analysis and machine learning, as well as skills in handling and processing large volumes of data.

Additionally, electric companies are significant advantaged of this project. By implementing predictive models and advanced analysis techniques, companies can manage energy demand more efficiently and improve the efficiency of the electrical grid. This can lead not only to a reduction in operating costs but also to greater stability and reliability of the electrical supply. Furthermore, by optimizing energy consumption in homes, companies can contribute to sustainability and responsible management of energy resources.

Finally, the environment significantly benefits from this project. Optimizing energy consumption in smart homes can reduce the carbon footprint and greenhouse gas emissions associated with electricity generation. By reducing energy demand, there is less need to use less clean energy sources, directly contributing to mitigating climate change and conserving natural resources.

7.5 Ethical or legal issues

Regarding ethical or legal issues, the data used in the project has been extracted from Kaggle. It is important to note that the legal status of the dataset is marked as "unknown" on Kaggle, which could potentially raise concerns regarding its usage in academic or commercial projects. However, as long as the dataset complies with Kaggle's terms of use and any applicable copyright or data sharing regulations, it should generally be considered legal for use in research and analysis.

8 Discussion

On the one hand, the SARIMAX model's comprehensive approach and integration of exogenous variables significantly enhance its performance, making it an ideal tool for forecasting energy consumption in smart homes. By optimizing energy usage, reducing waste, and enabling predictive maintenance, it supports sustainable living practices and helps manage increasing energy costs. The improvements in predictive accuracy and robustness underscore the model's value in creating efficient and resilient smart home systems.

It has been verified that the simulation has produced satisfactory and relevant results. In scenarios where the temperature drops, the probability of rain increases, and visibility decreases, it is observed that energy usage increases considerably. This may be due to unfavorable conditions for going outside, leading residents to spend more time at home. Consequently, they will use more energy and more household appliances. Although data for this is not available, it is possible that it is related to the use of heating.

On the other hand, when temperatures rise, the opposite effect occurs. The energy consumed is much lower, likely because residents do not spend as much time at home. Additionally, as mentioned earlier, this may be related to the non-use of heating. Finally, when there is a lot of wind, energy usage increases on several occasions. This could be related to poor weather conditions, similar to those previously mentioned. Furthermore, in this scenario as well as in the colder scenarios, the use of cars for transportation is encouraged, which increases the usage of the garage door.

9 Bibliography

[1] GitHub. Project data repository.
https://github.com/JavierLuqueSaiz/web-smarthouse.git

[2] Kaggle. Smart Home Dataset with weather Information.
https://www.kaggle.com/datasets/taranvee/smart-home-dataset-with-weather-information

[3] Streamlit. Smarthouse PROYIII.
https://smarthouse-proyiii.streamlit.app

[4] SCIE. GII-GRIN-SCIE (GGS) Conference Rating.
https://scie.lcc.uma.es/ratingSearch.jsf

[5] Medium. Time Series Analysis on Smart Home IOT with Weather data.
https://medium.com/@mrkarthik71/time-series-analysis-on-smart-home-iot-with-weather-data-346037010ac2

[6] MDPI. Ensemble-Based Spam Detection in Smart Home IoT Devices.
https://www.mdpi.com/2078-2489/11/7/344

[7] Science Direct. Machine learning approach of detecting anomalies and time-series of IoT devices.
https://www.sciencedirect.com/science/article/pii/S1110016822001260

[8] Preprints. Energy Consumption Prediction Using Machine Learning.
https://www.preprints.org/manuscript/201903.0131/v1