**Residential Energy Consumption Forecasting Using Gradient Descent: A Neural Network Approach And Comparative Analysis With Regression**

Sharmila Devi S1, Penumaka Dedeepya2, Jayanthi Sharma3 and Vibha Rao4

1Assistant Professor, School of Computer Science and Engineering

Vellore Institute of Technology, Chennai, India

sharmiladevi.s@vit.ac.in

UG Scholar, School of Computer Science and Engineering

Vellore Institute of Technology, Chennai, India

2penumaka.dedeepya2022@vitstudent.ac.in

[3jayanthi.sharma2022@vitstudent.ac.in](mailto:3jayanthi.sharma2022@vitstudent.ac.in)

4vibha.rao2022@vitstudent.ac.in

**Abstract:** This paper explores predicting household energy consumption based solely on the outside temperature. By concentrating on this single variable, the study aims to develop a more accurate and efficient forecasting model. The core of the study involves implementing a gradient descent algorithm to enhance prediction performance. The research employs both ridge regression and gradient descent algorithms to compare predicted values, highlighting their respective impacts on prediction accuracy. The research includes a review of various gradient descent algorithms, including Stochastic Gradient Descent (SGD) and its variants, to understand their impact on similar prediction tasks. The findings offer practical insights for utility companies, enabling better anticipation of energy demand and more effective resource management, showcasing the benefits of using temperature-based predictions and advanced optimization techniques for smarter energy management.

**1 Introduction**

Accurately predicting building energy requirements during the design phase is an essential task for architects, engineers, and energy planners, especially in the context of modern sustainable development initiatives. As the world increasingly prioritizes energy efficiency and the responsible management of natural resources, the ability to forecast energy consumption has become a critical component in the design and operation of residential and commercial buildings. This project aims to address this need by developing a system capable of predicting household energy consumption based on a single yet highly influential factor: outside temperature.

The prediction model developed in this project is built using a simple neural network architecture, consisting of a single neuron. This approach can also be conceptualized as an implementation of logistic regression, a fundamental technique in machine learning that is well-suited for binary classification tasks. Although the model's structure is straightforward, it is designed to effectively capture the relationship between external temperature and energy usage within residential settings. By focusing on this relationship, the project provides a tool that can help utility companies and energy managers anticipate fluctuations in energy demand, allowing for more efficient allocation of resources and better preparation for peak usage periods. To ensure that the predictions generated by this model are both accurate and reliable, the project leverages the gradient descent optimization method. Gradient descent is a widely used technique in machine learning, renowned for its ability to iteratively adjust the parameters of a model to minimize the loss function. This method ensures that the neural network converges to a solution that best fits the data, leading to more precise predictions.

In this project, we take a hands-on approach by implementing linear regression, neural network, and gradient descent optimization from scratch in Python. This not only provides a deeper understanding of the underlying mechanics of machine learning algorithms but also allows for greater customization and control over the model's behavior. Additionally, we aim to implement a TensorFlow model alongside the custom neural network. By conducting a comparative analysis of both approaches, we intend to gain deeper insights into their respective performance, scalability, and practical applications in energy consumption forecasting.

Ultimately, this project seeks to contribute to the broader field of energy management by providing a practical, easy-to-implement tool that can assist utility companies in better understanding and anticipating energy demand. The simplicity of the model, combined with the effectiveness of the gradient descent optimization, makes it a valuable addition to the arsenal of tools available to energy professionals. As the world continues to move towards more sustainable energy practices, the ability to accurately forecast energy consumption will become increasingly important, making this project both timely and relevant.

**2 Literature review**

[1] addresses the challenge of tuning the learning rate in traditional gradient descent algorithms, particularly in deeper models for image classification, where convergence can be suboptimal. It introduces the Fast Gradient Descent (FGD) algorithm, which improves both the efficiency and effectiveness of training neural networks. The proposed algorithm combines the advantages of SGD, backpropagation, and introduces a modification to the classical gradient descent by incorporating a Nesterov step and dynamically updating the learning rate using either the Armijo rule or a control step. Nesterov's method helped in accelerating gradient descent by taking a "look-ahead" step, which reduces the oscillations and leads to faster convergence and the control learning rate adapts during the training process, using the Armijo condition, thereby ensuring that each step is effective in minimizing the loss function. The algorithm was tested on the MNIST dataset, comprising 60,000 training images and 10,000 testing images of handwritten digits and was compared against other optimization methods such as SGD, NAG, Momentum, and Adadelta. It achieved a training accuracy of 97.54% and a test accuracy of 94.59%. It also reduced the CPU time compared to other methods. For instance, FGD had a CPU time of 83.17 seconds, which was significantly lower than the others. The paper concludes by highlighting the significance of the FGD algorithm in achieving faster and more accurate results in image classification, making it a valuable contribution to the field of machine learning and computer vision.

[2] investigates the enhancement of conventional solar distillers (CDS) by incorporating an inclined absorber plate and applying machine learning techniques, specifically Stochastic Gradient Descent (SGD) in artificial neural networks (ANN), to predict productivity. The paper involved designing and fabricating two solar distillers—a conventional one and a modified version with an absorber plate and conducting experiments over a 10-hour period, recording hourly measurements for water temperature, glass cover temperature, ambient temperature, and productivity. The results found that the addition of the absorber plate increased productivity by 138.68%, from 1311.3 ml/m².h to 3129.8 ml/m².h. in the Modified Solar Distiller (MSD). A two-hidden-layer neural network model was developed to forecast productivity, which was further improved using SGD. The SGD-enhanced neural network (NN-SGD) achieved high accuracy, with a determination coefficient of 1.000 and low prediction errors. The NN-SGD model outperformed the traditional neural network model in predicting productivity for both the conventional and modified solar distillers, demonstrating lower error measurements (RMSE and CVRMSE). The paper demonstrated that this use of SGD in neural networks proved to be an effective way to enhance predictive performance, and this novel approach could be applied to other renewable energy systems. The study concludes that combining machine learning with physical enhancements can significantly improve solar distiller productivity and suggests further research to optimize these techniques for broader applications in renewable energy.

[3] explores the application of deep learning techniques in ophthalmology, focusing on the use of Stochastic Gradient Descent (SGD) as the primary optimization method. The goal of the research was to develop classification models that could predict various eye conditions based on medical imaging data, such as optical coherence tomography (OCT) and fundus photographs. A convolutional neural network (CNN) architecture was employed, which is well-suited for image classification tasks, and was trained with SGD to fine-tune the model's weights. The dataset included labeled images of different eye conditions, which allowed the model to learn from both positive and negative examples. SGD was chosen because it efficiently handles large datasets and quickly converges to a good solution by updating weights incrementally with mini-batches of data. The paper reported remarkable accuracy rates exceeding 99.9% on both training and validation datasets for various classification tasks. The use of SGD enabled the model to effectively minimize the loss function, demonstrating its robustness in managing complex datasets typical in medical applications. Overall, the study highlighted the potential of deep learning, particularly with SGD, to enhance diagnostic accuracy in ophthalmology, paving the way for automated disease detection systems.

[4] addresses the challenge of fusing low spatial resolution multi-spectral (MS) aerial images with high spatial resolution panchromatic images to produce images with enhanced spectral and spatial resolutions. The authors proposed a novel method that combines model-based and data-driven approaches by utilizing an unrolled version of the Projected Gradient Descent (PGD) algorithm. In this method, instead of a traditional projection step, they use a deep Convolutional Neural Network (CNN), making the process more interpretable and effective. The gradient descent step updates the solution iteratively to minimize the error between the current estimate and the actual measurements. The experiments were conducted using a dataset of high-resolution aerial MS images and corresponding low-resolution versions and the methodology was tested against baseline methods such as bicubic interpolation, shrinkage fields, and deep coupled analysis and synthesis dictionary methods. By unrolling the PGD and training the whole process with CNNs, the method becomes both interpretable and powerful. Furthermore, the paper explores two scenarios: using an identity operator and learning the forward operator alongside the CNN. The latter proves to be more successful, as it better captures the underlying data distribution. The proposed method significantly outperforms baseline techniques in terms of Peak Signal-to-Noise-Ratio (PSNR) and Structural Similarity Index (SSIM), showcasing its ability to achieve superior results. The paper concludes that this approach not only outperforms existing baselines but also generalizes purely data-driven methods by incorporating learnable components throughout the process.

[5] compares different gradient descent-based optimization algorithms used in training Convolutional Neural Networks (CNNs) for image classification. The study focuses on popular techniques like Adam and RMSprop and compares them to traditional Stochastic Gradient Descent (SGD). The authors tested these algorithms on a standard image classification dataset, such as CIFAR-10 or ImageNet, using various CNN architectures. They adjusted hyperparameters like learning rates and momentum to see how they affected the speed of convergence and the accuracy of the classification. The results showed that Adam and RMSprop outperformed traditional SGD, both in terms of how quickly they converged and the accuracy they achieved. The paper highlighted that these advanced optimization techniques effectively adjusted learning rates dynamically, allowing for better exploration of the loss landscape. The paper concludes by emphasizing the importance of choosing the right optimization algorithm, as it can greatly impact the performance of deep learning models.

[6] introduces a new method called Sample Gradient Descent (SGD). This approach aims to tackle the challenges of computational complexity and slow convergence that often come with traditional gradient descent, especially when working with large datasets. Instead of using the entire dataset, the Sample Gradient Descent method selects a representative sample of the data using Principal Component Analysis (PCA). By retaining the rows and columns that explain 90% of the overall variance, this approach creates a smaller yet still informative dataset. The authors implemented their approach through a GDRegressor algorithm that adjusts parameters like learning rates and epochs to minimize the loss function. In their experiments, the Sample Gradient Descent method showed faster convergence, taking only 8 epochs compared to the 20 epochs required by traditional methods. Each epoch also took less time, around 3.41 seconds, making the method more efficient overall. Although there were slight trade-offs, such as a small increase in Mean Absolute Error (MAE) and a minor decrease in R² score, the reduced dataset size offered significant computational savings. The paper concludes that Sample Gradient Descent is a more efficient option for large datasets, offering strong performance with reduced computational demands and suggests that future research could explore optimizing hyperparameters and applying this method to deep learning, highlighting its potential in machine learning and optimization.

[7] provides a detailed analysis of Stochastic Gradient Descent (SGD) and its variations in natural language processing (NLP). The paper highlights how crucial SGD is for training models in tasks like sentiment analysis and language translation. The authors tested several NLP models, including recurrent neural networks (RNNs) and transformers, using SGD to optimize the model parameters by minimizing cross-entropy loss associated with classification tasks. They also discussed the benefits of mini-batch SGD, which speeds up convergence and improves generalization by updating weights based on smaller data batches. The results showed that SGD greatly enhances the performance of NLP models, particularly with large datasets. The efficiency of SGD in handling complex, high-dimensional data made it an essential tool for modern AI applications. It concludes that models trained with SGD achieved higher accuracy and faster training times compared to traditional optimization methods.

[8] explores the significance of Convolutional Neural Networks (CNNs) in the realm of artificial intelligence, particularly for image recognition tasks. CNNs are a specialized type of artificial neural network that utilize the mathematical method of convolution, enabling efficient feature extraction, weight sharing, and dimensionality reduction. Their hierarchical structure contributes to reliable computational speed and a reasonable error rate. The integration of Back Propagation (BP) and Gradient Descent (GD) mechanisms allows CNNs to learn from data through iterative feedback and optimization, enhancing their self-learning capabilities. The paper discusses the fundamental structure and functions of CNNs, details each layer's role, and examines the principles of BP and GD, concluding with practical examples that illustrate their applications in real-world scenarios.

[9] this study aims to predict the energy requirements of residential buildings in Agadir, Morocco, by utilizing artificial neural networks (ANN) as a learning algorithm. The research generated a training dataset of 5,625 samples through parametric analysis, considering various factors such as building orientation, relative compactness, glazing rate, wall surface area, height, and total surface area. Three types of residential buildings—Economic Villa, Economical Construction, and Medium Class—were selected for testing the ANN model. Energy demand calculations were performed using the Design Builder tool, and predictions were made with a Python program. The results demonstrated that the ANN model achieved high accuracy, with 98.7% for prediction data and 97.6% for test data, indicating its effectiveness in forecasting energy needs based on the specified parameters.

[10] this research focuses on accurately predicting the hourly cooling energy requirements for educational buildings at the University of Technology in Iraq during the design phase. Utilizing a feedforward artificial neural network (ANN), the study identifies key building parameters to serve as inputs for the model. Due to the lengthy process of collecting actual energy consumption data, the researchers employed the Hourly Analysis Program (HAP) for building simulation, generating a comprehensive database for the summer months. The study explored various training algorithms, ultimately finding that the Bayesian regularization backpropagation algorithm with a learning rate of 0.05 yielded the best results. The performance of the optimized ANN model was evaluated using mean square error (MSE) and correlation coefficient (R), demonstrating high prediction accuracy with values of 5.99×10⁻⁶ and 0.9994, respectively.

**3 Proposed Methodology**

**3.1 Data Collection**

The proposed methodology for this paper begins with data collection. Historical data on household energy consumption and corresponding outside temperature readings were gathered to form the basis of the study. Specifically, we utilized a rich dataset from the iFlex dynamic pricing experiment.

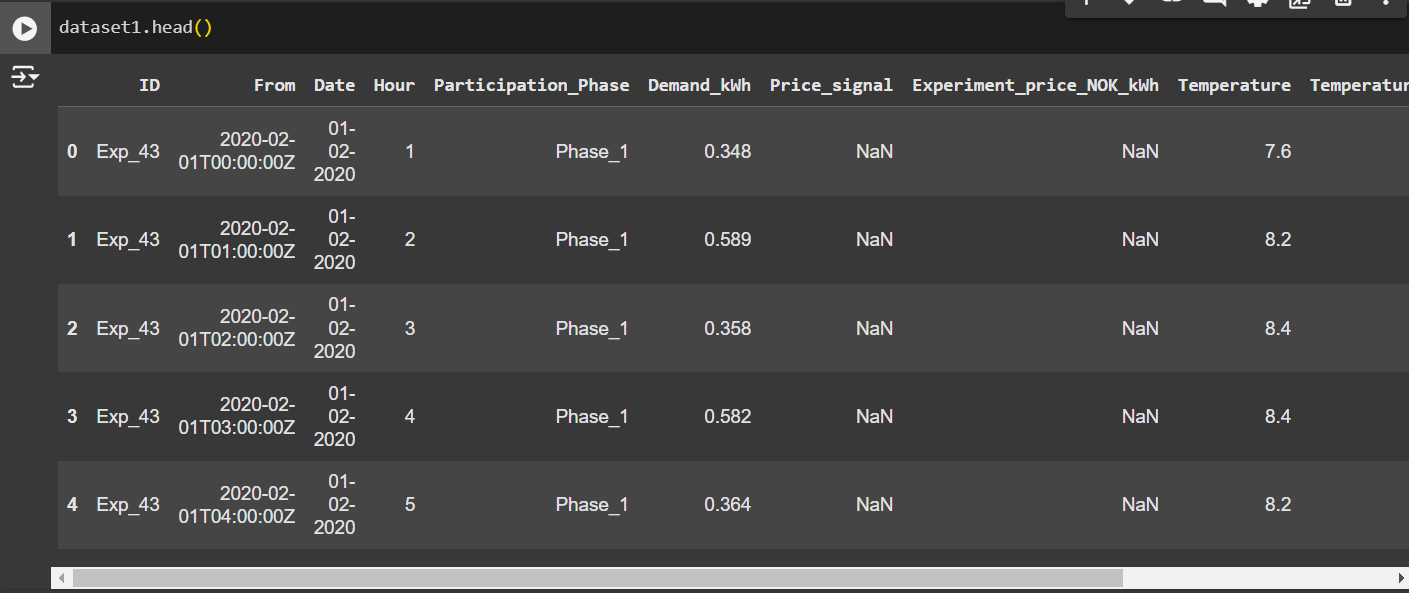
This dataset, created by Matthias Hofmann and Turid Siebenbrunner, contains hourly residential electricity consumption data along with survey responses. The experiment, conducted in several Norwegian regions over two winter periods (early 2020 to spring 2021), aimed to understand how households adjust their power consumption in response to variable electricity prices.

In addition to electricity consumption data, temperature readings were recorded, making this dataset particularly relevant to our study. The inclusion of temperature data enables an in-depth analysis of its impact on household energy consumption. This makes the iFlex dataset highly suitable for our research, as it provides comprehensive and relevant information to explore the relationship between outside temperature and energy usage.

Link for the database: https://zenodo.org/records/8248802

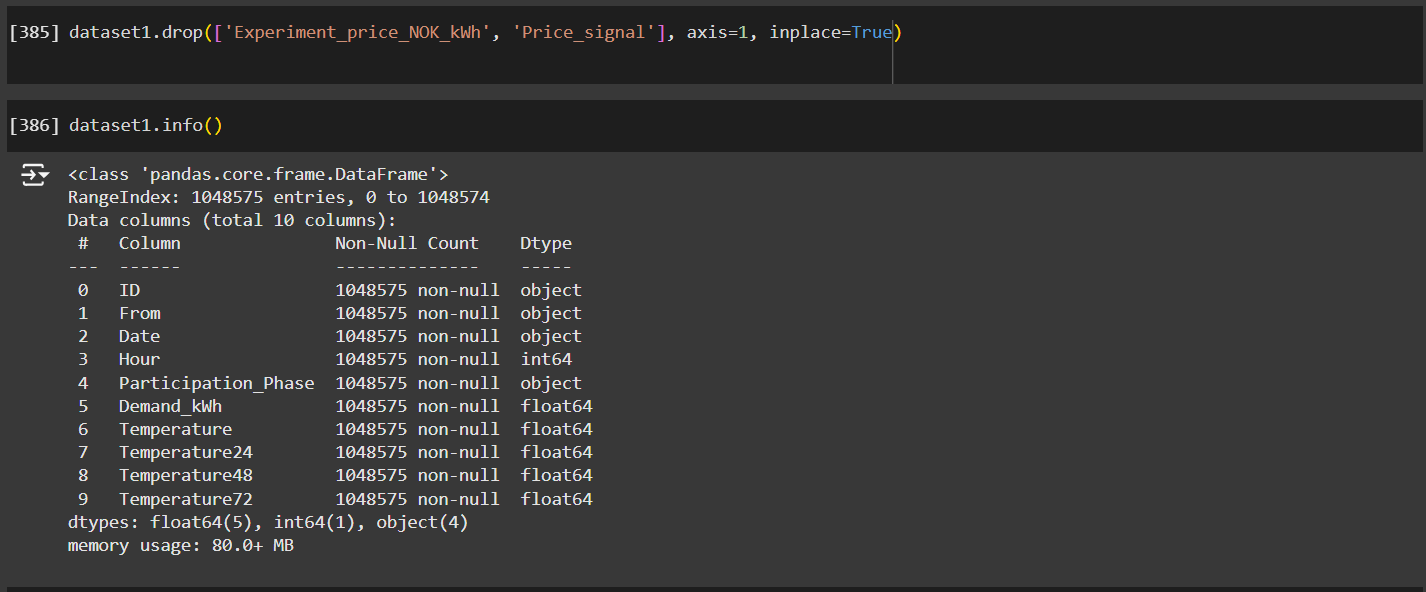
**3.2 Data Exploration And Preprocessing**

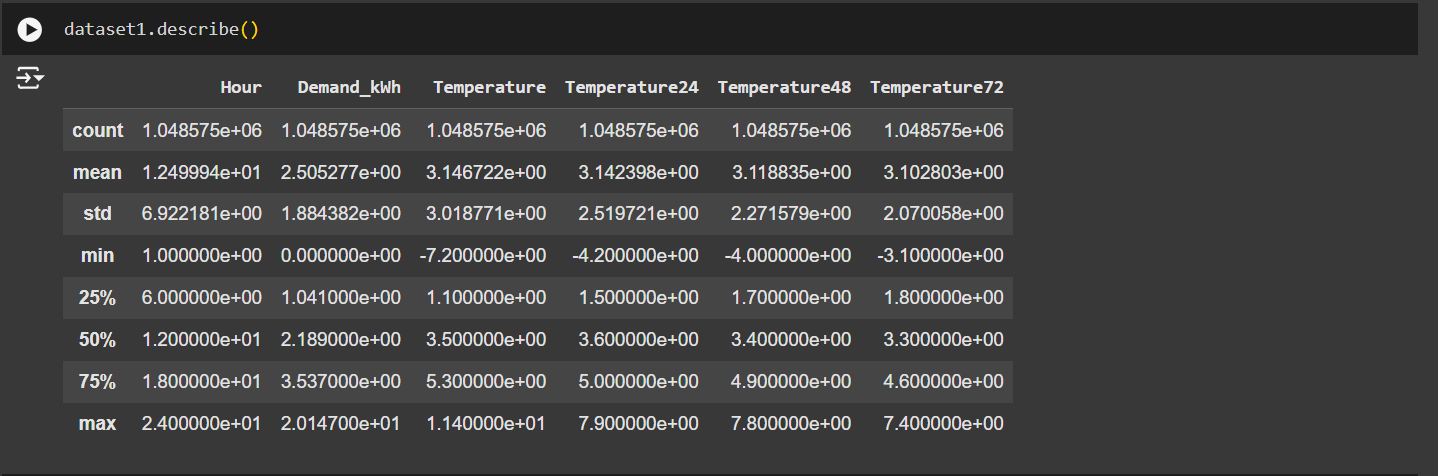
In the data exploration phase, we began with loading the dataset, followed by an initial exploration to understand its structure and content. This included examining the first few rows and obtaining the dataset's dimensions to determine the number of records and features.



**Figure 1**  Data Examining

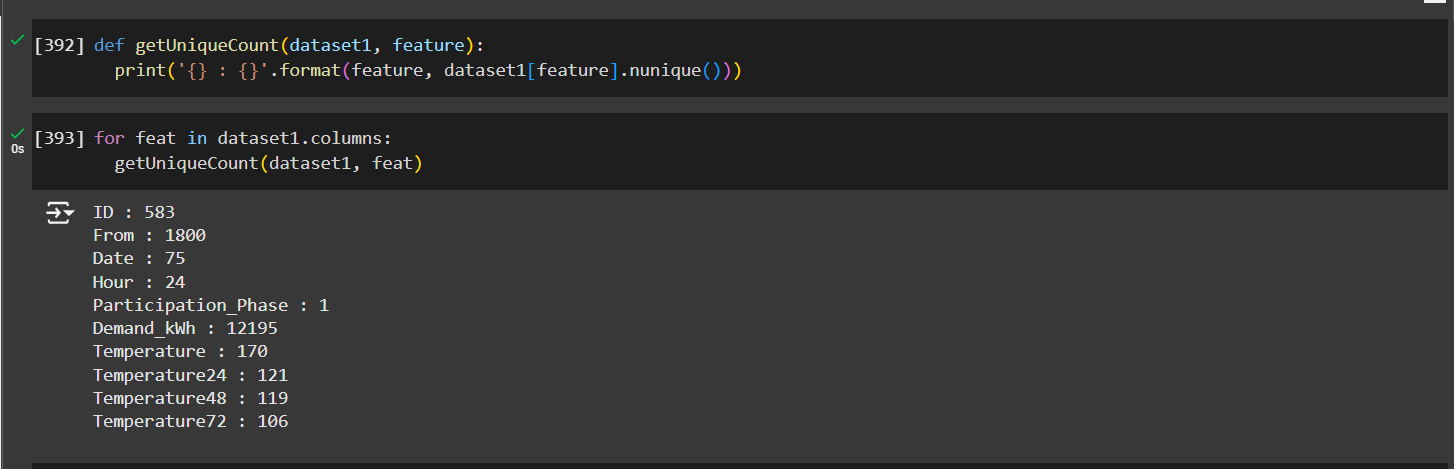
Next, irrelevant columns, such as Experiment\_price\_NOK\_kWh and Price\_signal, were removed to focus on variables relevant to the study. A detailed inspection of the dataset was conducted to check for missing values, ensuring data completeness. Statistical summaries provided insights into the data distribution and highlighted key characteristics of each feature.





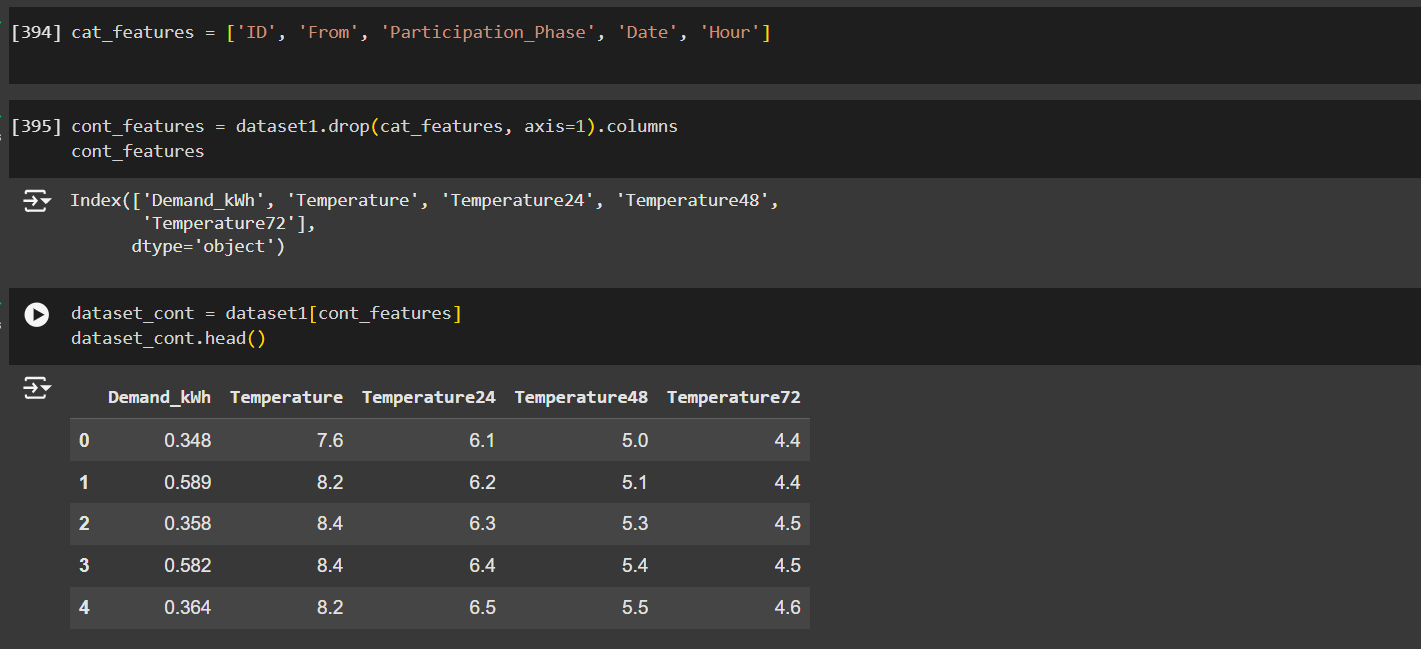
**Figure 2** Data Removal

The dataset was then analyzed to understand the distribution of values for each feature. The frequency and count of unique values were evaluated to identify categorical and continuous variables.



**Figure 3** Unique Features Retrieval

This was followed by segmenting features into categorical (e.g., ID, From, Participation\_Phase, Date, and Hour) and continuous categories. Continuous features were further explored to gain a deeper understanding of their statistical properties, laying the groundwork for model training.

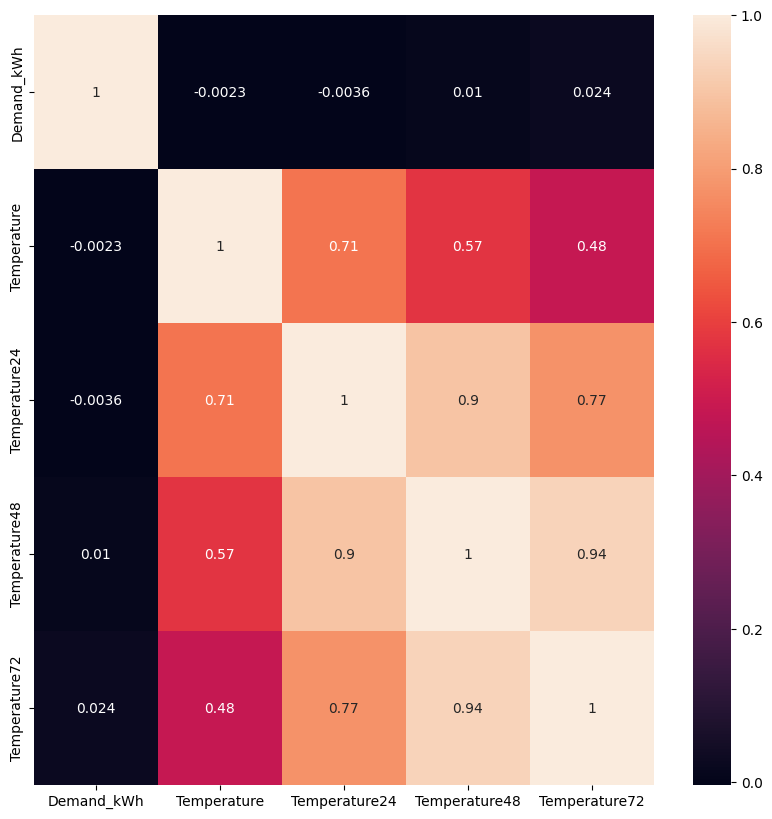


**Figure 4** Data Segmentation and Removal

Through this preprocessing ensured that the data was clean, well-structured, and ready for analysis, supporting the study’s goal of predicting household energy consumption based on temperature.

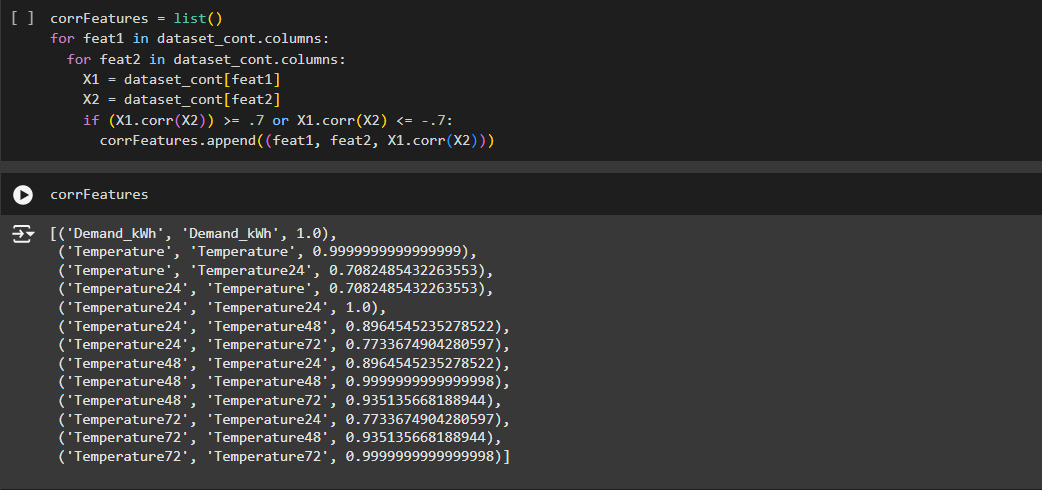
**3.3 Correlation Analysis and Feature Selection**

To better understand the relationships between continuous variables, we generated a correlation heatmap using Seaborn’s heatmap function. This allowed us to visually inspect the pairwise correlations among the features. The heatmap was annotated to show the correlation coefficients, which helped identify both strong positive and negative correlations.

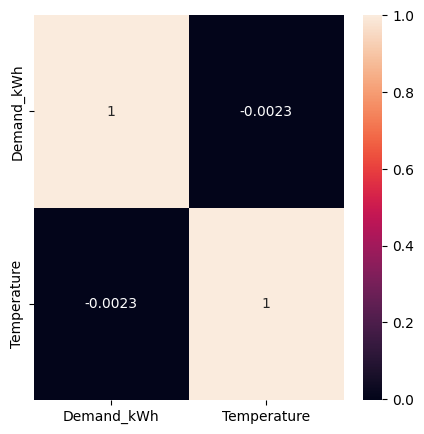


**Figure 5** Correlation Graph

Following the heatmap analysis, a more detailed correlation check was conducted for pairs of features. For any pair with a correlation coefficient greater than 0.7 or less than -0.7, it was flagged as a strongly correlated pair. These correlations were stored in a list for further review, helping to understand which features might have overlapping information. From this analysis, it was observed that Temperature and Temperature24 were strongly correlated. Given this high correlation, we concluded that using Temperature alone would be sufficient to predict energy consumption, as it encapsulates the same information as Temperature24 without adding redundancy.

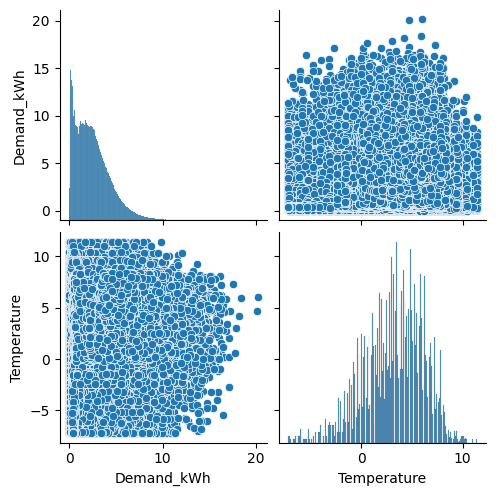
 **Figure 6** Correlation Values

Given the focus of the study on temperature and energy consumption, only the most relevant features—Temperature and Demand\_kWh (household energy consumption)—were selected for further analysis. The dataset was filtered to retain only these features, allowing for a more targeted investigation into their relationship.



**Figure 7** Temperature and Demand\_kWh Correlation

To visualize this, a correlation heatmap and pairwise scatter plots (using Seaborn’s pairplot) were created for the selected features. The pairplot provided a more granular look at the relationships between Temperature and Demand\_kWh, supporting the analysis of how temperature impacts energy consumption.



**Figure 8** PairwiseScatter Plot of Temperature and Demand\_kWh

**3.4 Data Preparation for Predictive Modeling**

**3.4.1. Anderson-Darling Test for Normality**

To prepare the data for predictive modeling, we first assessed the normality of the selected features using the Anderson-Darling test. This test helped determine whether the features Demand\_kWh (energy consumption) and Temperature followed a normal distribution, which is an important assumption for many linear models. The results showed that both features did not meet the normality assumption, as the test statistic for both Demand\_kWh and Temperature exceeded the critical values at all significance levels. Since normality was not confirmed, further steps were necessary to prepare the data for linear regression.

**3.4.2 Outlier Detection and Removal**

Outliers were identified based on both normality and the Interquartile Range (IQR) method. For normally distributed data, Z-scores were used to detect outliers. For non-normally distributed data, the IQR method was applied, identifying values that lie outside the range of 1.5 times the IQR below the first quartile (Q1) or above the third quartile (Q3). For both Demand\_kWh and Temperature, a substantial number of outliers were identified (22,000 for Demand\_kWh and 14,256 for Temperature). These extreme values could distort the model’s performance, so they were removed to ensure the data was more representative of typical values. After removing the outliers, the dataset was refined to include approximately 1.03 million rows for each feature.

**3.5 Gradient Descent using Regression**

Gradient descent is an optimization algorithm used to minimize the loss function by iteratively adjusting the model's parameters (weights). It calculates the gradient (or derivative) of the loss function with respect to the parameters and updates the parameters in the opposite direction of the gradient, thereby reducing the error over time.

In this project, gradient descent is used to optimize the model for predicting energy consumption based on outside temperature. The algorithm minimizes the Mean Squared Error (MSE) between the predicted and actual energy consumption values by updating the model’s weights during training. The goal of this project is to find the best-fitting model that accurately forecasts energy consumption, with gradient descent enabling efficient learning from the training data.

**3.5.1 Model Setup and Data Split**

The first step involved setting up a simple regression model to predict household energy consumption (Demand\_kWh) based on Temperature. The dataset was limited to 5,000 samples for computational efficiency. This data was then split into training and testing sets with an 80-20 split, ensuring robust evaluation of the model's performance.

**3.5.2 Gradient Descent with L2 Regularization**

A custom gradient descent optimizer was implemented to minimize the loss function. To prevent overfitting, L2 regularization (Ridge Regression) was incorporated, adding a penalty proportional to the square of the weights. This technique helps in reducing model complexity and improving generalization.

* **Loss Function**: Mean Squared Error (MSE) was used, augmented with a regularization term.
* **Hyperparameters**: The model was trained with a learning rate of 0.01, 200 epochs, and a regularization coefficient of 0.01.

In order to increase the accuray, the model was continuously trained with progressively increasing epoch values. Initially, the model was trained for 200 epochs, allowing the gradient descent algorithm to iteratively minimize the error by updating weights and biases.

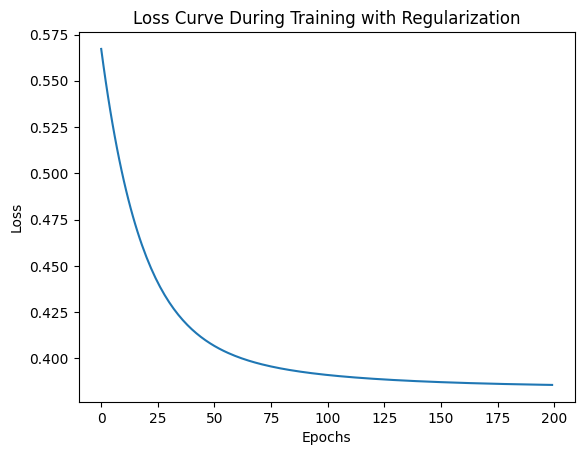
**3.5.3 Model Training and Performance Metrics**

The gradient descent optimizer iteratively adjusted the weights (w) and bias (b) to minimize the regularized loss function. The training process was monitored via a loss curve, which showed a steady decrease in loss, indicating effective learning.

Post-training, the model was evaluated using standard metrics:

* Mean Squared Error (MSE)
* Mean Absolute Error (MAE)
* R-squared (R²)
* Mean Absolute Percentage Error (MAPE)

The model achieved competitive performance on both the training and testing sets, reflecting its ability to generalize.



**Figure 9** Loss Curve During Training With Regularization

**3.5.4 Prediction for New Data Points**

A function was implemented to predict energy consumption for new temperature inputs. Given a new temperature value, the function normalizes the input using the training data's mean and standard deviation. The trained model then predicts energy consumption, which is subsequently denormalized to provide a kWh value.

**3.5.5 Visualization of Predictions**

A scatter plot was used to compare the predicted values with the true values for both training and testing sets. The alignment of points along the identity line indicated a good predictive performance.

This model effectively captures the linear relationship between outside temperature and energy demand, providing utility companies with valuable insights for forecasting and resource management.

**3.6 Gradient Descent using Tensor Flow, A Neural Network Approach**

**3.6.1 Model Setup and Data Split**

The first step in our methodology involved setting up a Neural Network (NN) to predict household energy consumption (Demand\_kWh) based on outside temperature. The dataset was limited to 5,000 samples for computational efficiency. The data was split into **training** and **testing** sets with an **80-20 split**, ensuring robust evaluation of the model's performance.

**3.6.2 Neural Network Design**

The neural network was designed as follows:

* **Input Layer:** 1 neuron representing the temperature variable.

The input layer consisted of 1 neuron because we are using a single feature—**temperature**—to predict the energy demand. In a regression problem like this one, the input layer typically matches the number of input features. Since only temperature data (a single feature) is being used to predict energy demand, we only need one neuron here.

* **Hidden Layers:** 2 hidden layers, with 64 and 32 neurons, respectively, and ReLU activation to introduce non-linearity.

The two hidden layers were added between the input and output layers to allow the model to capture more complex relationships between the input and the target. Neural networks can model non-linear relationships, which is why we introduce multiple layers.

**First Hidden Layer (64 neurons):** This layer has **64 neurons**, which allows the model to learn a large number of complex features from the input data. The more neurons in a layer, the more features the model can learn, though it also increases the risk of overfitting if the network is too large for the dataset.

**Second Hidden Layer (32 neurons):** The second hidden layer reduces the number of neurons to **32**, which is a common practice to narrow down the learned features as the model moves towards the output. This helps in balancing model complexity and avoiding overfitting.

* **Output Layer:** 1 neuron representing the predicted energy demand (Demand\_kWh).

This neuron is used to predict the **energy demand** (Demand\_kWh) based on the input temperature. Since this is a regression problem (predicting a continuous value), the output neuron does not use an activation function (i.e., the activation function is **linear** by default).

**Activation Function:** ReLU for hidden layers to allow the model to capture complex patterns.

Both hidden layers use ReLU (Rectified Linear Unit**)** as the activation function. ReLU is widely used because it allows the network to introduce non-linearity, enabling it to learn complex relationships in the data. Specifically, ReLU outputs the input if it's positive and zero otherwise:

f(x)=max(0,x)

The use of ReLU also helps with faster training compared to other activation functions like sigmoid or tanh, as it does not suffer from the vanishing gradient problem.

**Optimizer:** Adam optimizer with a learning rate of 0.001 to ensure efficient training.

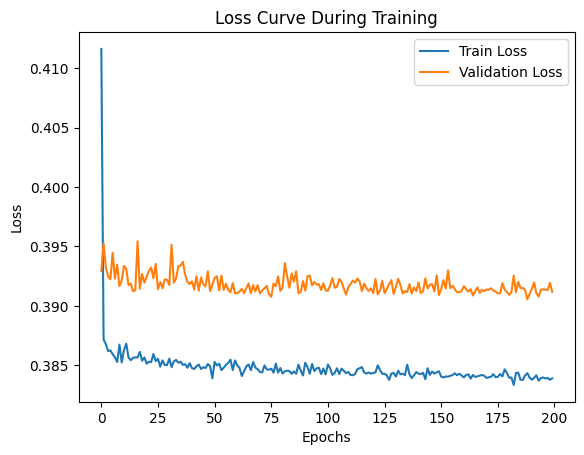
The Adam optimizer was chosen because it adapts the learning rate for each parameter individually and performs well in practice for most tasks. Adam stands for Adaptive Moment Estimation, which combines the advantages of two other extensions of stochastic gradient descent: Momentum (which helps accelerate gradients vectors in the right directions) and RMSProp (which helps with the learning rate). The learning rate for Adam is set at **0.001**, which is a common default that provides a good balance between fast learning and avoiding overshooting the optimal weights.

**Loss Function and Optimizer:**

* **Mean Squared Error (MSE)** is used as the loss function because this is a **regression problem**.
* **Adam optimizer** is used with a learning rate of 0.001 for efficient convergence.

**3.6.3** **Model Training**

* The model is trained for 200 epochs, with each epoch involving a complete pass through the dataset. Using the Adam optimizer, the model adjusts its weights to minimize MSE through backpropagation. The batch size is 32, meaning the dataset is split into batches of 32 samples, with weight updates occurring after each batch. Smaller batch sizes can improve generalization by enabling more frequent updates.
* During the training process, the loss was monitored and plotted to evaluate the model's convergence and performance across the epochs The training loss and validation loss were plotted against the number of epochs. Training Loss represents how well the model is fitting the training data and the Validation Loss is calculated on the validation dataset (a subset of data that is not used for training). This helps track whether the model is overfitting to the training data or generalizing well to new data.



**Figure 10** Training and Validation Loss Curve

**3.6.4 Model Evaluation Metrics:** After training the model, several metrics were used to evaluate its performance on both the training and testing datasets:

* **Mean Squared Error (MSE):** This metric helps quantify the error by computing the squared difference between the actual and predicted values. Lower MSE indicates a better fit to the data.
* **Mean Absolute Error (MAE):** This measures the average magnitude of errors in the predictions, without considering their direction. It is more interpretable than MSE because it is in the same units as the target variable (energy demand).
* **R-squared (R²):** This represents the proportion of variance in the target variable (energy demand) that can be explained by the model. A higher R² indicates a better model fit.
* **Mean Absolute Percentage Error (MAPE):** This provides a percentage representation of how far off the predictions are, on average, from the true values. It is useful for understanding the model's accuracy in relative terms.

These were displayed finally as the output.

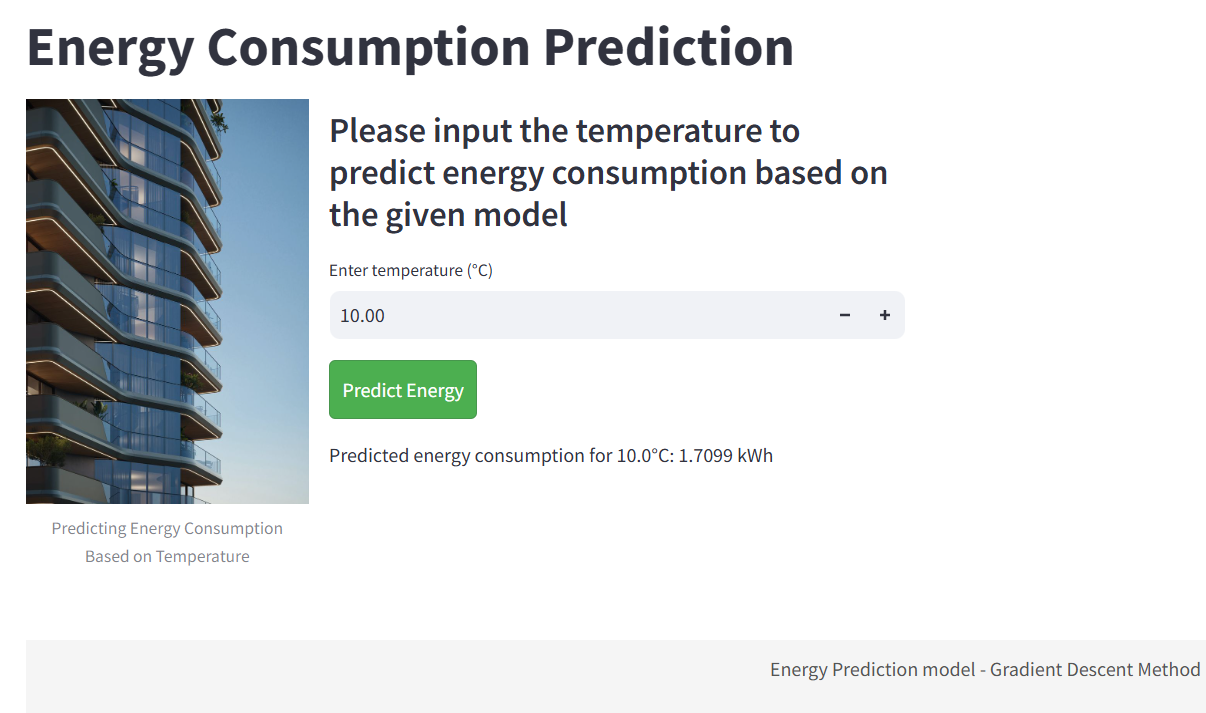
**3.6.5 Visualization and Prediction for User Input**

A scatter plot was used to compare the predicted values with the true values for both training and testing sets. A function was implemented to predict energy consumption for new user temperature input. It displayed the energy consumed along with the temperature.

After implementing the models, the best method was chosen based on the performance metrics. In this way we could produce an efficient model to predict household energy consumption based on outside temperature.

**3.6.6** **Interactive Energy Consumption Prediction Dashboard Using Streamlit**

In our Streamlit application, we display energy consumption predictions in kWh based on user-inputted outside temperatures. This tool, inspired by our research, uses temperature as a single predictor variable to estimate household energy demand, aiming to provide a user-friendly and effective forecasting method. Implementing a gradient descent algorithm, our model adjusts parameters for optimal prediction performance, which is useful for practical applications. Utility companies can leverage such temperature-driven models for improved demand anticipation and resource allocation, enhancing efficiency and supporting data-driven decisions in energy management.



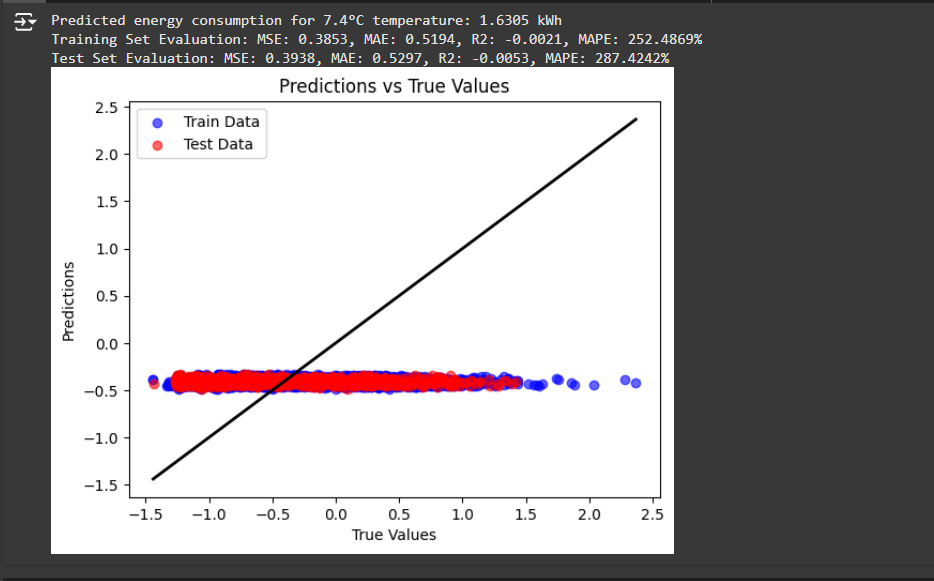
**Figure 11** Front EndInteractive Energy Consumption Prediction

**4 Results and Discussions**

The regression model provided the following evaluation metrics:

* Training Set Evaluation:
* MSE: 0.3853
  + MAE: 0.5194
  + R²: -0.0021
  + MAPE: 252.49%
* Test Set Evaluation:
  + MSE: 0.3938
  + MAE: 0.5297
  + R²: -0.0053
  + MAPE: 287.42%

The model's performance is quite poor, as indicated by the negative R² values, which suggest that the model does not capture the variance of the target variable at all. The high MAPE values further highlight significant errors in the predicted energy consumption compared to the actual values. Despite this, the model is able to predict the energy consumption for specific temperatures, e.g., 7.4°C resulted in a predicted value of 1.6305 kWh, though this prediction lacks accuracy due to the overall poor model fit.



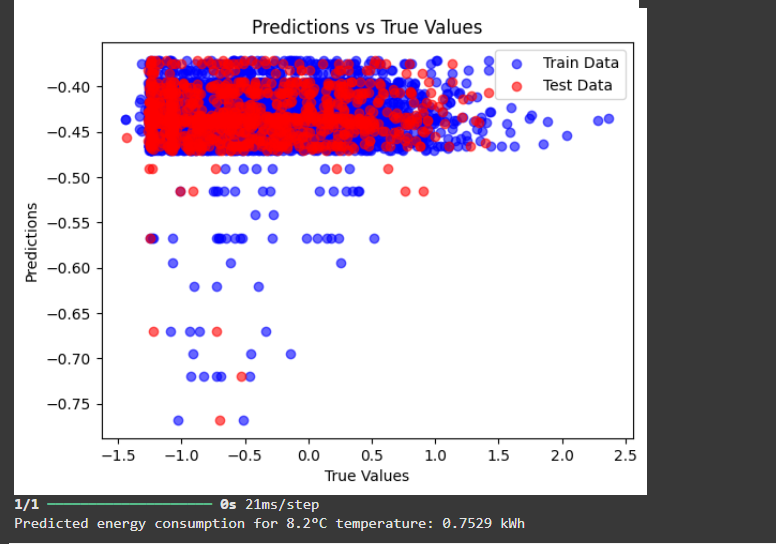
**Figure 12** Plot of Test Data and Data Predicted Using Regression

Neural Network Model Performance

For the neural network, the evaluation metrics were:

* Training Set Evaluation:
  + MSE: 0.3832
  + MAE: 0.5149
  + R²: 0.0032
  + MAPE: 7.19%
* Test Set Evaluation:
  + MSE: 0.3912
  + MAE: 0.5245
  + R²: 0.0014
  + MAPE: 7.75%

The neural network showed a significant improvement over the regression model. The positive R² values suggest that the neural network was able to capture some relationship between temperature and energy consumption. Additionally, the MAPE values (7.19% for the training set and 7.75% for the test set) indicate that the predictions were more accurate compared to the regression model. The model also demonstrated a reasonable prediction for energy consumption at 8.2°C (0.7529 kWh), showcasing its ability to generalize better than the linear regression approach.



**Figure 13** Plot of Test Data and Data Predicted Using Neural Network

Loss Curve Analysis: The loss curves during training demonstrated consistent improvement, with the training loss decreasing steadily and the validation loss remaining stable, suggesting that the model was learning effectively without overfitting.

**5 Conclusion**

This project focused on predicting household energy consumption based on outside temperature using linear regression and neural network models.

The regression model performed poorly with low R² and high MAPE, indicating it couldn't capture the complex, non-linear relationship between temperature and energy demand.

The neural network model significantly outperformed the linear regression model, showing better prediction accuracy with lower MSE, MAE, and MAPE, and higher R². This indicates its ability to model the non-linear relationship more effectively.

The neural network’s ability to handle non-linear relationships makes it more suitable for real-world applications where multiple factors influence energy consumption beyond just temperature**.**

In conclusion, this project demonstrates the power of artificial intelligence and machine learning, particularly neural networks, in predicting energy consumption patterns and offers valuable insights for improving forecasting accuracy in energy management systems

**6 Future Scope**

The project highlights the importance of selecting appropriate models for predictive tasks. Further research could involve refining the neural network by experimenting with different architectures, hyperparameters, and incorporating additional variables that influence energy consumption (such as humidity, time of day, or historical energy usage patterns).

Expanding the dataset to include more diverse temperature ranges and data points could also help improve the model's performance, especially for extreme conditions that were not captured in this study.

**7 References**

[1] Mouatasim, A. E. (2020). Fast gradient descent algorithm for image classification with neural networks. *Signal Image and Video Processing*, *14*(8), 1565–1572. https://doi.org/10.1007/s11760-020-01696-2

[2] Mohammed, S. A., Al-Haddad, L. A., Alawee, W. H., Dhahad, H. A., Jaber, A. A., & Al-Haddad, S. A. (2023). Forecasting the productivity of a solar distiller enhanced with an inclined absorber plate using stochastic gradient descent in artificial neural networks. Multiscale and Multidisciplinary Modeling Experiments and Design, 7(3), 1819–1829. https://doi.org/10.1007/s41939-023-00309-y

[3] Mustapha, A., Mohamed, L., & Ali, K. (2020). An overview of gradient descent algorithm optimization in Machine Learning: Application in the ophthalmology field. In Communications in computer and information science (pp. 349–359). https://doi.org/10.1007/978-3-030-45183-7\_27

[4] Unrolled projected gradient descent for multi-spectral image fusion. (2019, May 1). IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/abstract/document/8683124

[5] A comparative analysis of Gradient Descent-Based optimization algorithms on convolutional neural networks. (2018, December 1). IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/abstract/document/8769211

[6] Ganie, A. G., & Dadvandipour, S. (2023). From big data to smart data: a sample gradient descent approach for machine learning. Journal of Big Data, 10(1). https://doi.org/10.1186/s40537-023-00839-9

[7] Tian, Y., Zhang, Y., & Zhang, H. (2023). Recent advances in stochastic gradient descent in deep learning. Mathematics, 11(3), 682. <https://doi.org/10.3390/math11030682>

[8] Jihad, A. S., & Tahiri, M. (2018). Forecasting the heating and cooling load of residential buildings by using a learning algorithm “gradient descent”, Morocco. Case Studies in Thermal Engineering, 12, 85–93. https://doi.org/10.1016/j.csite.2018.03.006

[10] Prediction of hourly cooling energy consumption of educational buildings using artificial neural network. (n.d.). CORE Reader. https://core.ac.uk/reader/296919160