AI/ML for 5G-Energy Consumption Modelling

Predicting energy consumption of different 5G products

Krishna Priya | Rajat Ranjan
krishnapriyakejriwal@gmail.com | rajat5ranjan@gmail.com

Abstract - The paper discusses the energy consumption challenges posed by 5G networks despite their higher energy efficiency compared to 4G. It highlights that over 70% of the energy consumption is attributed to the radio access network (RAN), specifically base stations. To address this issue, the paper emphasizes the importance of optimizing base station parameters and energy-saving methods. The key factors affecting base station energy consumption are identified, such as architecture, configuration parameters, traffic conditions, and energy-saving methods. Accurate energy consumption modeling is considered crucial for more efficient network deployments. The paper concludes by introducing an AI challenge aimed at developing a machine learning-based solution to address these energy efficiency concerns in 5G networks.

1. Introduction

Wireless communication technology has seen remarkable advancements over the years, with the advent of the fifth generation (5G) ushering in a new era of connectivity. 5G networks promise to deliver unprecedented data speeds, ultra-low latency, and support for an ever-expanding ecosystem of Internet of Things (IoT) devices. These advancements bring about new services, technologies, and networking paradigms that have the potential to transform industries and enhance the quality of life for people around the world.

However, with the introduction of 5G, a pressing concern has emerged — the substantial increase in energy consumption associated with these cutting-edge network deployments. While 5G networks are estimated to be approximately four times more energy-efficient than their predecessors, the energy demands they place on operators are roughly three times larger. This is largely due to the necessity for a higher density of network cells to provide coverage at higher frequencies and the increased processing required to accommodate wider bandwidths and a greater number of antennas.

Notably, more than 70 percent of this energy is attributed to the radio access network (RAN), with base stations (BSs) being the primary energy consumers. Data centers and fiber transport, while contributing to energy consumption, represent a smaller share of the overall energy expenditure.

Efforts to address these energy concerns revolve around optimizing base station parameters and implementing energy-saving methods. Achieving this optimization necessitates a profound understanding of how various parameters and methods impact the energy consumption of different base stations. Accurate modeling of energy consumption emerges as an essential element in the pursuit of more energy-efficient network deployments.

This paper delves into the challenges and opportunities presented by 5G network energy consumption. It outlines the factors influencing base station energy consumption, including specific architecture, configuration parameters, traffic conditions, and energy-saving methods. Moreover, it introduces an exciting machine learning challenge aimed at designing innovative solutions to estimate and optimize energy consumption in diverse base station products and configurations. The purpose of this paper encompasses not only the development of energy consumption models but also the achievement of generalization capabilities across different base station products and configurations.

The primary goal of the problem is to develop machine learning models that accurately estimate and optimize the energy consumption of various base station products and configurations, and the primary metric for evaluating the performance of these models is the **Weighted Mean Absolute Percentage Error (WMAPE)**. The metrics which we will be focusing on this paper will be **MAE** and **MAPE**. The challenge aims to encourage solutions that not only provide accurate estimates but also demonstrate the ability to generalize across different scenarios, contributing to more energy-efficient and sustainable 4G and 5G network deployments.

2. Datasets

The dataset used in this context consists of three key datasets, each providing essential information related to 4G and 5G cellular base stations. These datasets are as follows:

Base Station Basic Information

This dataset contains fundamental details about the base stations, encompassing configuration parameters and hardware attributes.

Some of the features includes **BS** - Name of Base Station, **CellName** - Name of Cell, **RUType** - Name of the Radio unit Type, **Mode** - Transmission mode, **Frequency** - Frequency of Cell, **Bandwidth** - Bandwidth of Cell, **Antennas** - Number of antennas of Base station, **TXPower** - Maximum Transmit power of Cell

It has around 1020 Unique Base stations with 4 different CellName values, 12 different RU Types and 3 Modes along with different attributes related to a Base Station.

Cell-Level Data

The cell-level data dataset includes granular hour-level statistics, which cover a range of counters. These counters encompass aspects such as service compliance (e.g., load) and energy-saving methods (e.g., the duration of energy-saving mode activation).

Some of the Features includes

- **Time** - Hourly time in which measurement was collected
- **BS** - Base Station
- **CellName** - Name of Cell
- **Load** - Load of cell (0-1)
- **ESMode(1-6)** - Energy Saving Modes

Energy Consumption Data

The energy consumption data dataset is focused on providing detailed hour-level information regarding the energy consumption of the base stations.

It includes specifications related to the total energy consumed by the base stations, shedding light on their energy efficiency and consumption patterns.

In summary, these three datasets collectively form a comprehensive resource for analyzing and understanding the performance and energy characteristics of 4G and 5G base stations. The base station basic information, cell-level data, and energy consumption data are critical components for research and analysis in the field of network optimization and energy efficiency in the context of emerging cellular technologies.

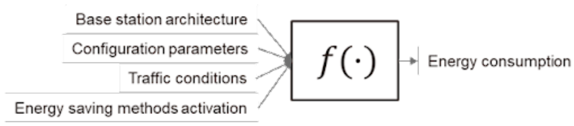

Figure 1. Factors affecting Energy Consumption

2.1 Exploratory Data Analysis and Data Processing

In the pursuit of effective data modeling, several key considerations were made. One critical aspect was the **CellName**, which, despite its variations among individual base stations, exhibited no discernible impact on energy consumption. Consequently, we took **CellName0** for modeling based on local cross-validation results.

Of paramount importance were features such as **RUType**, **Mode**, and other configuration-related parameters. These raw configuration-based attributes proved to be pivotal in understanding and modeling energy consumption patterns.

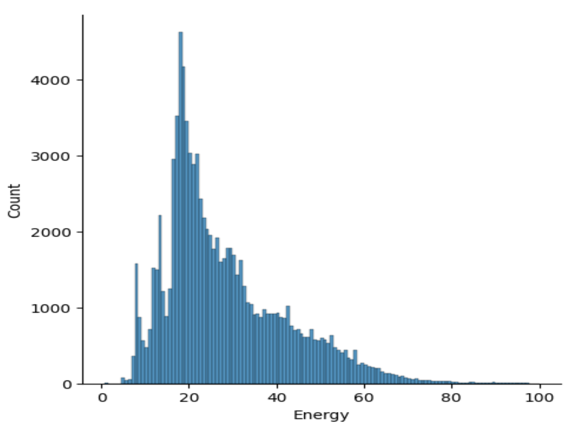

Figure 2. Energy Histogram Plot

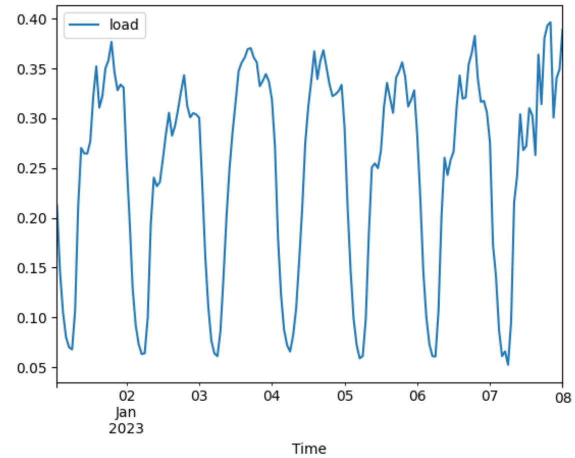

Figure 3. Average Load Values at a Daily Level

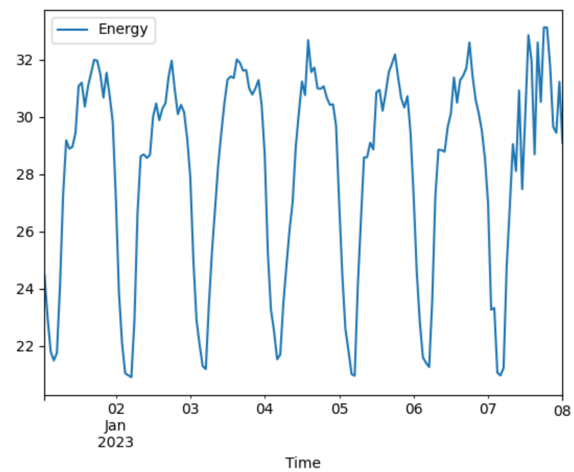

Figure 4. Average **Energy** Values at a Daily Level

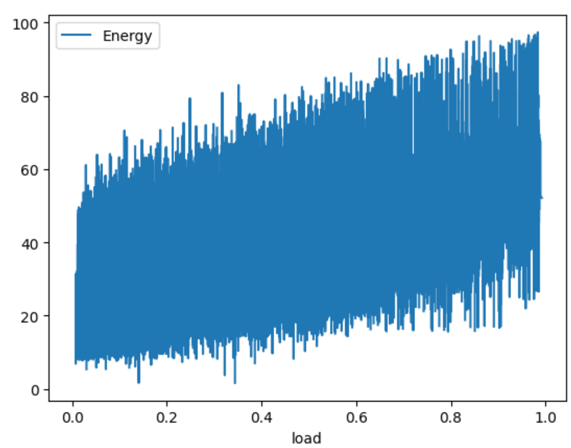

Figure 5. Linear Relationship between Load and Energy

- **Figure 2** presents a notable skewness in the distribution of energy values, hinting at the need for potential data transformations to enhance model performance.
- **Figure 3** and **Figure 4** reveal a pronounced relationship between load and energy consumption, particularly concerning the hour of the day. This suggests the presence of seasonal components associated with these variables, warranting a closer examination of temporal patterns.
- **Figure 5** illustrates a linear relationship between load and energy, offering further insights into the dynamics governing these crucial factors.

These initial observations serve as a foundation for the subsequent data modeling efforts, emphasizing the need to address data skewness and the intricate relationship between load, energy, and temporal variations. Such insights are pivotal in crafting accurate and effective models for energy consumption estimation and optimization in the context of 4G and 5G base stations.

2.2 Feature Engineering

The process of feature engineering plays a pivotal role in elucidating the methodology employed to address the problem statement at hand. This section provides an in-depth exploration of the feature engineering steps undertaken, shedding light on the aspects that have contributed to model improvement as well as those that yielded limited impact.

Date Time Features: To capture the temporal dimension of the data, we harnessed the **Time** attribute as a timestamp. From this, we derived static features such as **Hour**, **Dayofweek**, **Date**, and others. These static features enable us to dissect and comprehend the data in various time-based contexts, enriching the feature set.

Periodic Spline Features: Leveraging the concept of periodic splines, we extended our feature engineering to incorporate smoother encoding and enhanced periodicity. Specifically, we utilized features like **Hour** to generate their corresponding spline representations. This approach not only contributes to the refinement of feature encoding but also introduces periodicity into the expanded feature set.

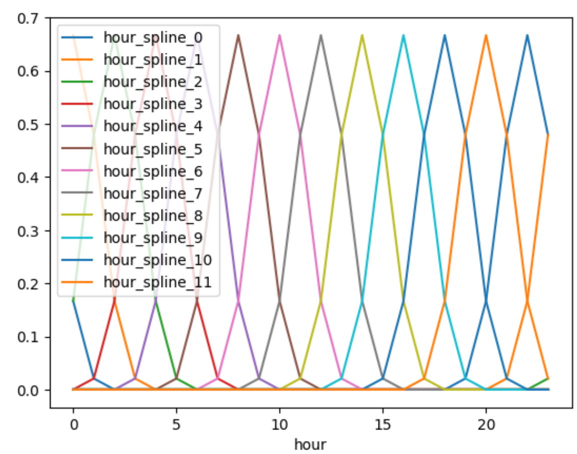

Figure 6. Periodic spline-based encoding for the 'hour' feature

Feature Bins: The introduction of feature bins involved a process of discretization for raw attributes such as **Load**. This discretization allowed us to transform continuous data into discrete values, facilitating a different perspective on the underlying information.

Time-Based Features without Future Data: To address the challenge of target leakage, we took a strategic approach to feature creation. We meticulously engineered time-based features that inherently exclude the use of future data. This involved generating lagged variables for raw attributes like **Load**, **ESModes**, and **Energy**. The time-based lagging, implemented via the shift function with respect to time (T-1, T-2 and T-3), effectively mitigated the risk of leveraging future data values.

Signal Processing: Our feature engineering endeavors extended into the realm of signal processing, drawing from the capabilities of the SciPy library to filter and smooth the data. Two key signal processing techniques were employed:

1. **Savitzky-Golay Filtering ([SG Filter]):** This technique applied polynomial smoothing to our data, enhancing data quality. By specifying the window length and polynomial order, we obtained various derivative orders and a smoothed signal. The resulting features, **load_smooth**, **load_diff**, **load_diff2**, and **load_diff3**, captured different aspects of signal dynamics, reducing noise and uncovering valuable insights from the data.
2. **Second-Order Section [(SOS) Filtering]:** We harnessed Butterworth filtering with second-order section (SOS) design to further enhance our signal processing capabilities. This approach, implemented through the **add_sosfiltfilt** function, utilized fourth and eighth-order Butterworth filters to modify the **load** data. Operating casually and without knowledge of future data points, this filtering method introduced **load_sosfilt** features. Adjusting filter parameters allowed us to tailor the SOS filter to specific research scenarios, enhancing data quality and interpretability. This technique proves invaluable for isolating frequency components and noise suppression while preserving causality in time series data.

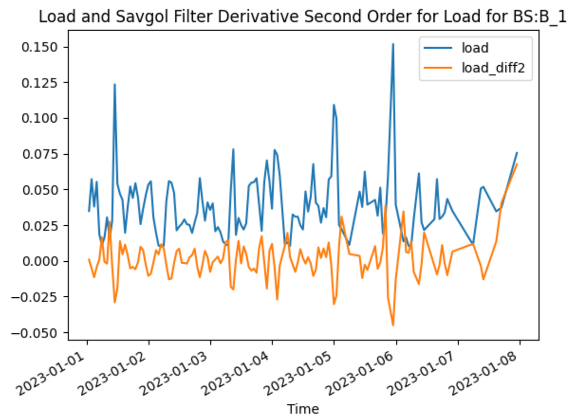

Figure 7. Load and Savgol Filter Derivative Second Order for Load for **B_1**

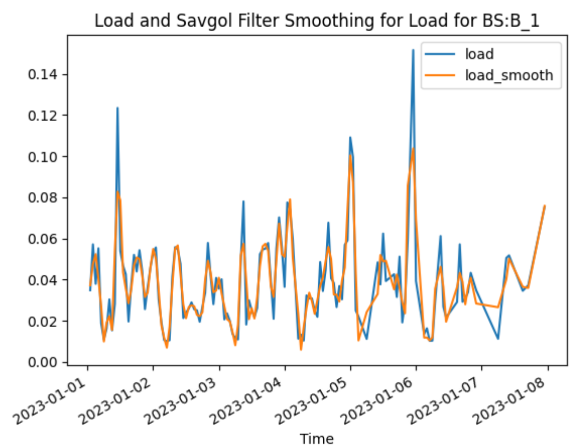

Figure 8. Load and Savgol Filter Smoothing for Load for **B_1**

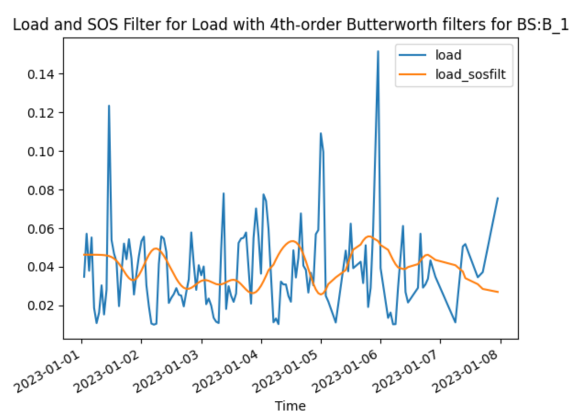

Figure 9. Load and SOS Filter for Load with 4th-order Butterworth filters for **B_1**

It is essential to note that **NONE of the above features nor the signal processed features like Savitzky-Golay filter (SG filter) nor the Butterworth filter (SOS filter) relies on future data**. Instead, they operate within a causal framework, utilizing past and current data exclusively. These techniques are instrumental in processing complex signal dynamics and enhancing the quality and interpretability of our dataset, aligning with the objectives of our research.

NOTE: We also refrained on Using aggregate features as mostly it can include future signals which might have overfitted the model.

2.3 Feature Selection

- First, we identify and remove columns with non-unique values. This is done by checking if the number of unique values in a column is less than or equal to 1.
- Next, we identify and remove columns with a high proportion of missing data (i.e., exceeding 95%) in both the training and test datasets. This step is essential for data quality and accuracy in modeling.
- Finally, we update the list of selected features by excluding the removed features.
- We also refrained from using **BS** feature from our set of raw features such that the generalization is based on inputs rather than base stations.

This feature selection process is vital for improving the quality and efficiency of subsequent modeling and analysis within the context of the research paper.

3. Cross Validation Strategy

3.1 Cross-Validation Based on Problem Statement

In our pursuit of understanding and effectively addressing the problem statement, a profound consideration was given to the choice of cross-validation strategy. Rather than adopting a conventional cross-validation approach, our objective was to align the strategy with the specific goals of the research.

Energy Consumption Estimation: One of the primary objectives was the development of machine learning models capable of accurately estimating the energy consumption of various base station products. This endeavor necessitated the incorporation of critical factors, including engineering configurations, traffic conditions, and energy-saving methods.

Generalization Across Products: It was imperative that the models demonstrate the capacity to generalize their energy consumption estimates to new base station products, even when based on data from existing ones. These models were designed to accurately predict energy consumption for products that had not been previously encountered.

Generalization Across Configurations: Another pivotal objective was to empower the models to predict energy consumption for base stations with diverse configurations. This requirement held even when the training data provided limited information about these specific configurations.

3.2 Best Suited Strategy

In alignment with the aforementioned objectives, we implemented the GroupKfold cross-validation methodology specifically on base stations. This choice was rooted in our pursuit of ensuring that generalization relied on the raw features and parameters of the base stations themselves.

- **Energy Consumption Estimation:** Our approach recognized that the energy consumption for different base stations should inherently vary, given their distinct parameters. This fundamental understanding underscored the need for a strategy that accounted for such variations.
- **Generalization Across Products:** The utilization of energy consumption data from one base station to predict the consumption of an entirely new base station was a key facet of our strategy. This approach effectively achieved generalization across different base station products.
- **Generalization Across Configurations:** Our strategy further extended to making predictions at the root level, considering the specific configuration of each base station. This approach facilitated generalization across diverse configurations, even in scenarios where limited training data was available.

Moreover, our chosen strategy served to encourage the models to regularize their parameters effectively. This enabled them to predict energy consumption with a reliance on the characteristics of individual base stations rather than relying on future data, which was a fundamental aspect of our approach.

4. Modelling

Considering the problem statement, we used several machine learning algorithms with the best suited strategy like XGBoost, LightGBM, CatBoost, ANN etc taking it as an Regression problem statement.
Below we will go over the models which worked in our favor.

4.1 Model Inputs and Output

Input Features:

Instantaneous raw features:

- *load, ESMode1, ESMode2, ESMode3, ESMode5, RUType, Mode, Frequency, Bandwidth, Antennas, TXpower*

Timestamp derived features:

- *day, weekday_number, hour, hour_spline_0, hour_spline_1, hour_spline_2, hour_spline_3, hour_spline_4, hour_spline_5, hour_spline_6, hour_spline_7, hour_spline_8, hour_spline_9, hour_spline_10, hour_spline_11*

Historical features for each BS (If available):

- *load_T-1, ESMode1_T-1, ESMode2_T-1, ESMode3_T-1, ESMode6_T-1, Energy_T-1,*
  load_T-2, ESMode1_T-2, ESMode2_T-2, ESMode3_T-2, ESMode6_T-2, Energy_T-2,
  load_T-3, ESMode1_T-3, ESMode2_T-3, ESMode3_T-3, ESMode6_T-3, Energy_T-3, Time_T-1_hours_elapsed, Time_T-2_hours_elapsed, Time_T-3_hours_elapsed

Binned load features:

- *load_bin*

Smoothed load features:

- *load_smooth, load_diff, load_diff2, load_diff3, load_sosfiltfilt, load_sosfilt*

Output:

- ***Energy***

We started with linear and tree based models bagging and boosting models, however we soon realized that linear models were underfitting and tree based models were overfitting as this specific data presented a challenge of unseen base stations (tree models work on the principle of binning, averaging and improving). Thus we started experimenting with ANNs as mentioned here (N. Piovesan 2022). ANNs were able to generalize well and converge faster.
Finally we stuck with two models for further tuning and improved them.
1. Fast AI based ANN (helps to iterate faster, inbuilt implementation of SHAP inferences etc)
2. Keras based custom ANN (for playing with different activations and embedding dimensions)

Both of them are explained in the later sections.

4.2 Model Architecture

We landed on the ANN as the architecture using 2 different modeling approaches using Fast AI and Keras ANN.
Let's look at its architecture one by one.

4.2.1 Fast AI ANN

Model Architecture:

A ANN architecture with embeddings of categorical features (**RUType, Mode, load_bin**) and dense layers for all other input features.

Model Layers:

It represents a feedforward neural network with five layers. In this architecture, there are **256** neurons in the input layer, **512** neurons in the second layer, **1024** neurons in the third layer, **512** neurons in the fourth layer, and **256** neurons in the output layer.

Data Preprocessing:

All input variables (section 4.1) were standardized and passed to the FastAI ANN.

```
TabularModel (Input shape: 1024 x 3)
============================================================================
Layer (type)         Output Shape         Param #    Trainable
============================================================================
                     1024 x 7
Embedding                                 91         True
----------------------------------------------------------------------------
                     1024 x 3
Embedding                                 9          True
----------------------------------------------------------------------------
                     1024 x 21
Embedding                                 2142       True
Dropout
BatchNorm1d                               102        True
----------------------------------------------------------------------------
                     1024 x 256
Linear                                    20992      True
ReLU
BatchNorm1d                               512        True
----------------------------------------------------------------------------
                     1024 x 512
Linear                                    131072     True
ReLU
BatchNorm1d                               1024       True
----------------------------------------------------------------------------
                     1024 x 1024
Linear                                    524288     True
ReLU
BatchNorm1d                               2048       True
----------------------------------------------------------------------------
                     1024 x 512
Linear                                    524288     True
ReLU
BatchNorm1d                               1024       True
----------------------------------------------------------------------------
                     1024 x 256
Linear                                    131072     True
ReLU
BatchNorm1d                               512        True
----------------------------------------------------------------------------
                     1024 x 1
Linear                                    257        True
----------------------------------------------------------------------------

Total params: 1,339,433
Total trainable params: 1,339,433
Total non-trainable params: 0

Optimizer used: <function Adam at 0x7c322b832b90>
Loss function: FlattenedLoss of MSELoss()

Model unfrozen

Callbacks:
  - TrainEvalCallback
  - CastToTensor
  - Recorder
  - ProgressCallback
```

Figure 10. Fast AI Model Architecture with 5 Layers

Loss Function and Metrics:

The loss function was specified as the mean absolute error (**MAE**) due to the competition's public leaderboard choice and also, **MAPE** as a loss function was leading to underpredictions and was not able to converge as well as MAE, thus the choice of MAE.

Activation Function: *ReLU*

Private Leaderboard Score: WMAPE - 0.0549

4.2.2 Keras ANN (*Final Model*)

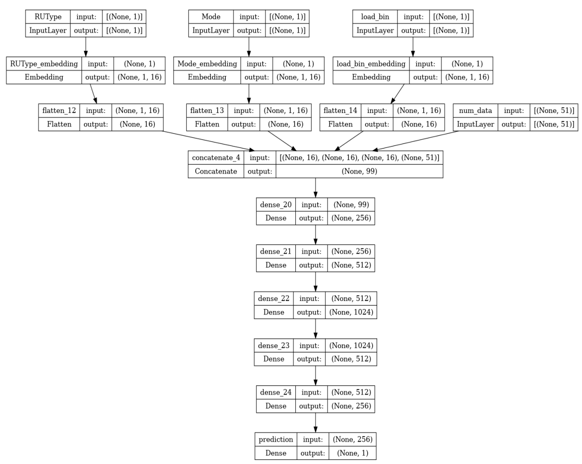

Figure 11. Keras ANN Model Architecture with 5 Layers

Model Input and Architecture:

From the **54** input features mentioned above Section 4.1), 3 categorical features (**RUType, Mode** and **Load_bin**) were converted into embedding layers with 1 input neuron for each and 16 output neuron for each. And the rest 51 features were passed as continuous features with 51 neurons.

Thus input layer consisted of:

16+16+16+51 neurons

Model Hidden Layers:

It represents a feedforward neural network with five layers. In this architecture, there are **256** neurons in the first layer, **512** neurons in the second layer, **1024** neurons in the third layer, **512** neurons in the fourth layer, and **256** neurons in the output layer.

Data Preprocessing:

All input variables (section 4.1) were standardized and passed to the Keras ANN.

Loss Function and Metrics:

Similar to the FastAI model the loss function used was **MAE** with 10 fold cross validation.

Activation Function:

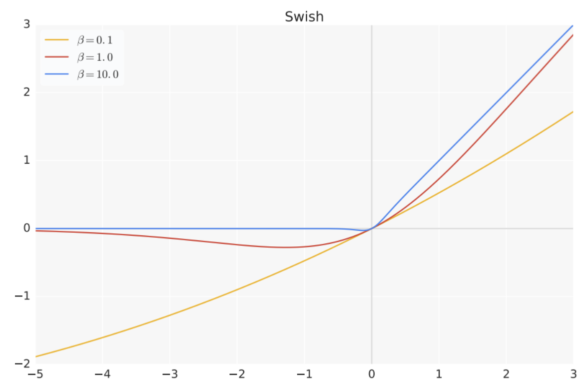

Figure 12. Swish Activation Functions

$$f(x) = x \cdot sigmoid(\beta x)$$

We used the **Swish** Activation function with beta = 1, also called as ***SiLU***: *Sigmoid-weighted Linear Unit.*

In simple terms, the SiLU (Sigmoid-weighted Linear Unit) activation function worked better than ReLU for our tabular regression problem because of its unique properties:

- **Non-Monotonic Behavior**: Unlike ReLU, SiLU doesn't always increase as its input gets larger. It has a global minimum value around -0.28, which means it can inhibit the growth of weights that are too large. This prevents our neural network from learning weight values that are too extreme, which can lead to overfitting.
- **Implicit Regularization**: The global minimum of SiLU serves as a "soft floor" on the weights, effectively acting

as an implicit regularizer. This encourages our neural network to maintain weights at reasonable magnitudes, which is beneficial for generalization and prevents overfitting.

- Similarity to ReLU for Large Inputs: When the input values to SiLU are large, its behavior is similar to ReLU. This means that SiLU can still capture the beneficial features of ReLU when the inputs have large magnitudes.

In essence, SiLU combines the advantages of ReLU (encouraging sparsity and faster convergence) with a self-stabilizing mechanism that prevents extreme weight values. This led to better performance in our tabular regression problem, where controlling the scale of weights was crucial for achieving good results without overfitting.

Private Leaderboard Score: WMAPE - 0.0435

4.3 Model Selection

Finally Keras ANN model was selected as the final model due to its better performance in the local Group K Fold cross validation and public leaderboard. Also the difference between training loss and validation loss of this model was lesser compared to all other other models which showed less overfitting and better generalization.

4.4 Final Submission

The final submission for the competition was a 10 fold mean prediction of Keras ANN model. However the mean prediction of FastAI and Keras ANN also had a good score. The individual model benchmarks are published in the benchmark section.

5. Model Explainability and Interpretation

To enhance the comprehensibility of our model outputs, this section delves into the critical domain of model interpretability, leveraging **SHAP (SHapley Additive exPlanations)** analysis as our primary tool. Our endeavor to understand the model's behavior is bifurcated into two distinct sections, each addressing a specific facet of model interpretability.

Model Understanding and Features of Existing Base Stations

The first section of our model interpretation exercise concentrates on comprehending the behavior of the model in the context of existing base stations. These base stations, with a historical presence in the training dataset, provide us with vital insights into their energy consumption patterns. By delving into the features associated with these established base stations, we aim to gain a comprehensive understanding of how the model accounts for their specific characteristics.

Model Understanding and Features for New Base Stations

The second section of our interpretability exploration is dedicated to model understanding concerning new base stations. These base stations, for which we lack historical energy consumption data, present a unique challenge. To address this, our model must extrapolate its understanding based on the available information. By investigating the features related to these uncharted base stations, we aim to unravel how the model navigates the intricacies of unobserved scenarios.

It is noteworthy that this division in our model explainability analysis aligns closely with the fundamental objectives set forth in our cross-validation strategy, as elucidated in **Section 3.2**. This approach ensures that our model's interpretability efforts are intricately linked to the core goals of the problem statement.

5.1 Model Explainability for Existing Base Stations

Interpretation for Importance of features for Existing Base stations which have a time based history of Energy consumptions on an hourly basis. This will summarize and give us an indication towards points **Energy Consumption Estimation**

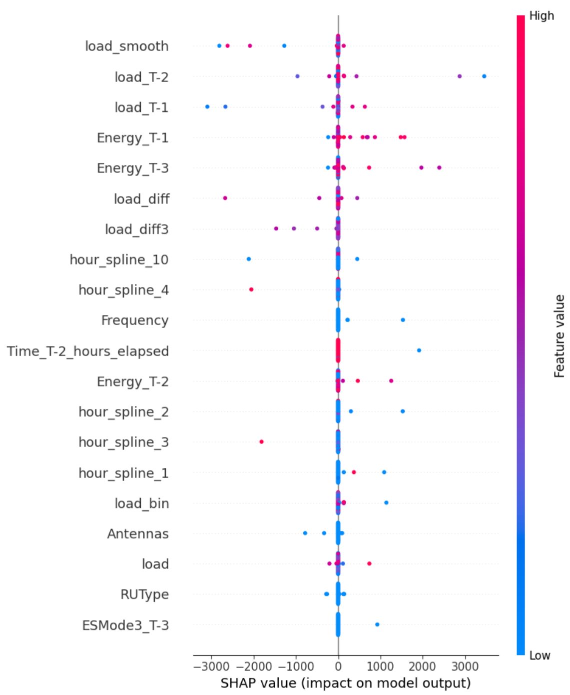

Figure 13.SHAP summary plot for **BS 143**

Lets understand some insights about our model that which Feature is important for **Base Station <u>143</u>** which had around **44** samples

- Smoothing features have a very **HIGH** impact on the model, since **load_smooth** with high and low values have a good impact.
- **HIGH** and **LOW** values of **load_T-1,load_T-2 and load_diff** have a very large impact on model output, suggesting the Energy is increasing based on Lagged values of Load.
- The interesting part of this analysis is **the importance of Energy_T-1, Energy_T-2 and Energy_T-3** for which **HIGH** values for **Energy_T-1** has a very good impact on model performance suggesting very good correlation.
- Considering the **hour_spline** features which are also indicating good model insights.

<u>NOTE:</u> The SHAP interpretation is done on **BS-143** which had around **24 samples**, the results may vary accordingly if we try to look for another BS

5.2 Model Explainability for New Base Stations

Interpretation for Importance of features for NEW Base stations which don't have a history. This will summarize and give us an indication towards **Generalization Across Products and Generalization Across Configurations**. There are around **97 Base stations** which don't have a history and are totally new Base stations with different configurable parameters. This approach can give a good explanation on which features are dependent.

Lets understand some insights about our model that which Feature is important for **Base Station <u>1016</u>** which HAVE NO History in Train data regarding their Energy consumption and are completely new Base stations

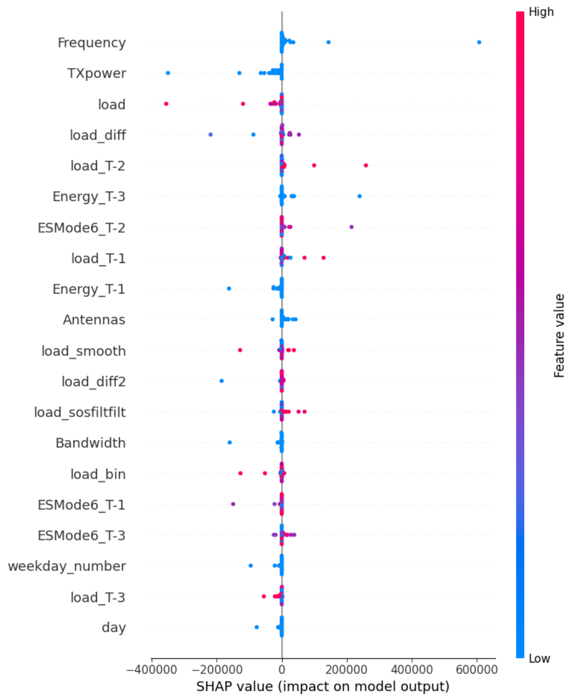

Figure 14. .SHAP summary plot for **BS 1016**

- **LOW** values of **Frequency** have high impact on the model output, which can suggest that low Frequency have low Energy consumption
- **LOW** values of **TXPower** tends to have not much impact towards the model, thought suggested as a good indicator
- **HIGH** values of **Load** tend to have **LOW** impact (though good importance), but **HIGH** values of **load_T-2** and **load_diff** have a very impact on model output, suggesting the Energy is increasing based on Lagged values of Load.
- Since, these are new Base stations and doesn't have any **Energy consumption or (value=0 for Lagged Variables), therefore we can see that the values of all Energy Lagged values such as Energy_T-1, Energy_T-2 and so on have very low Values and not much predictive** in terms of raw features like Frequency and TXPower. ***This particular is the difference between the NEW BS and EXISTING BS.***
- We can also see some of the smoothing variables for load such as **load_diff2** and **load_sosfiltflit** are also having a good impact on models with higher values suggesting even good predictive ability.
- **ESMode** lag variables are holding good impact on model

6. Conclusion

In this competition, we focussed on creating a generalized model that achieves the Objectives as mentioned in **Section 3.1**, which provides predictions on Base Stations where we have historical data along with physical configurations and also for the Base Stations where we only have information regarding its physical configurations. So, instead of making two separate models for both scenarios, through our experiments we came to the conclusion that ANN models are better suited for the above generalization task which targets both paradigms of the problem statement respective of the real world scenario.

7. Future Work and Analysis

Energy Efficiency Optimization: Future research can focus on developing more advanced algorithms and methodologies for optimizing the energy efficiency of 5G networks. This could involve the exploration of novel techniques for reducing energy consumption in radio access networks (RAN) and base stations (BS).

Machine Learning Advancements: As machine learning plays a pivotal role in estimating and optimizing energy consumption, future work could delve into more sophisticated machine learning models. For eg. Experimenting with DNN Architectures or improving model scores with other activation functions.

Dynamic Network Configuration: Investigating dynamic network configurations that adapt to varying traffic conditions and user demands could be a promising avenue. Machine learning algorithms that can dynamically adjust base station parameters in real-time to optimize energy consumption while ensuring quality of service are worth exploring.

Resource Allocation Strategies: Future research can delve into resource allocation strategies that balance the trade-off between energy consumption and network performance along with Location dependent features. This could include features like traffic load, user density, and environmental factors.

Real-world Deployments and Testing: Validating the proposed solutions in real-world 5G network deployments and conducting extensive testing is crucial. Future work can involve collaborating with network operators and conducting large-scale trials to assess the practicality and performance of energy-efficient strategies.

Ethical and Societal Considerations: As 5G technology becomes more deeply integrated into society, future research may examine the ethical and societal implications of energy-efficient 5G networks. This could encompass issues related to privacy, data security, and equitable access to efficient 5G services.

These pointers for future work and analysis can provide a roadmap for researchers and industry professionals to continue exploring and addressing the complex challenges and opportunities presented by 5G network energy consumption and efficiency.

Appendix:

Model Scores:

1. **FastAI ANN**:
    a. *Local GroupKFold OOF MAPE:* ***0.029***
    b. *Public Leaderboard:* ***0.0558***
    c. *Private Leaderboard:* ***0.0549***
2. **Keras ANN**:
    a. *Local GroupKFold OOF MAPE:* ***0.026***
    b. *Public Leaderboard:* ***0.0434***
    c. *Private Leaderboard:* ***0.0435***
3. **Mean Weighted Ensemble**:
    a. *Local GroupKFold OOF MAPE:* ***0.0259***
    b. *Public Leaderboard:* ***0.0474***
    c. *Private Leaderboard:* ***0.0470***

References:

1. McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, *5*(4), 115–133.
2. Howard, Jeremy & others. (2018). fastai.GitHub. Retrieved from https://github.com/fastai/fastai
3. Chollet, F., & others. (2015). Keras. GitHub. Retrieved from https://github.com/fchollet/keras
4. Agarap, A. F. (2018). Deep learning using rectified linear units (relu). *ArXiv Preprint ArXiv:1803.08375*.
5. Ramachandran et al., (2017). Swish: a Self-Gated Activation Function. *ArXiv Preprint ArXiv:1710.0594.*