SmartHome Savant: Optimizing Living Spaces with LLM Innovation

Abhilash, Aditya, Ankith, Snigdha

Introduction

This report presents the development and outcomes of three innovative applications designed for the Savant Home system. These applications leverage advanced technologies such as Large Language Models (LLMs) and SHAP analysis to enhance household energy management and device interaction, aiming to improve efficiency, reduce costs, and enhance user experience.

Applications Overview

1. Anomaly Detection in Household Energy Consumption with SHAP
    a. We developed models capable of detecting anomalies in energy consumption and predicting future usage. These models integrate SHAP to elucidate feature importance, assisting in the identification of critical factors influencing energy usage. Recommendations for energy-saving practices are derived from key SHAP features, leveraging insights from LLMs to optimize household energy management.
2. FAQ Model Fine-Tuning
    a. Our team fine-tuned a local LLM using FAQ data specifically curated to address user queries and facilitate access to knowledge base documents.
    b. Additionally, we implemented Retrieval-Augmented Generation (RAG) to derive nuanced insights directly from user manuals and support documentation.
3. Optimal Device Placement
    a. Utilizing the advanced capabilities of Gemini Vision Pro, we provided tailored recommendations for the optimal placement of devices within a room. This application ensures devices operate at peak efficiency and effectiveness, enhancing user experience and device functionality.

Now, let us explore each application in detail.

1. Change Detection and Forecasting in Smart Home

Overview:

In analyzing smart home energy data, we focused on two primary objectives:

1. **Anomaly Detection**: Proactively detecting excessive energy consumption to prevent increased usage fees.

2. **Future Consumption Forecasting**: Utilizing weather data to predict and optimize future energy supply and demand.

Dataset and Preprocessing:

1. We have downloaded the dataset from [14].
2. We dropped the rows if all the columns are NA.
3. For other missing values, we used the 'bfill' strategy to fill up the values.
4. We checked the correlation of various features and dropped the columns that are highly correlated with ratio >= 0.9
5. Observations by EDA:
    a. Energy consumption is low during the day and high at night.
    b. Energy generation is high during the day and low at night.
    c. It is thought that this is because Energy generation is promoted because there are no residents at home during the daytime, and consumption increases at night because the residents return home.
    d. Home office, Fridge, Wine cellar, Living room and Furnace clearly have time-series trends. This is because these appliances need to keep the indoor temperature constant or adjust to a comfortable temperature according to the season.

Case1. Predict Future Energy Consumption:

1. It seems that it is possible to catch change points in the usage tendency of energy consumption from the data of energy consumption.
2. By capturing changes in consumption trends, it may be possible to think out ways to increase energy supply in months when consumption is likely to increase and decrease it in months when consumption is likely to decrease.

What are Change Points? [1]

1. The change point is the point at which the trends in time series data change over time.
2. Outliers indicate a momentary abnormal condition (rapid decrease or increase), while change points mean that the abnormal condition does not return to its original state and continue.

ChangeFinder Overview [2]

1. ChangeFinder is an algorithm used to detect change points.
2. ChangeFinder uses the log-likelihood based on the SDAR(Sequentially Discounting AR) algorithm to calculate the change score.
3. SDAR algorithm introduces a discounting parameter into the AR algorithm to reduce the influence of past data, so that even non-stationary time series data can be learned robustly.

ChangeFinder has two steps of model training:

Training STEP1

1. Train a time series model at each data point using the SDAR algorithm
2. Based on the trained time series model, calculate the likelihood that the data points at the next time point will appear
3. Calculate the logarithmic loss and use it as an outlier score

$$Score(x_t) = -logP_{t-1}(x_t|x1, x2, \ldots, x_{t-1})$$

Smoothing Step

1. Smooth the outlier score within the smoothing window(W).
2. By smoothing, the score due to outliers is attenuated, and it is possible to determine whether the abnormal condition has continued for a long time.

$$ScoreSmoothed(x_t) = (1/W) * \Sigma[(t = t - W + 1)\ to\ t]\ Score(x_i)$$

Training STEP2

1. Using the score obtained by smoothing, train the model with the SDAR algorithm
2. Based on the trained time series model, calculate the likelihood that the data points at the next time point will appear
3. Calculate the logarithmic loss and use it as an change score

Hyperparameter Tuning

1. Discounting parameter *r* (0 < *r* < 1) : The smaller this value, the greater the influence of the past data points and the greater the variation in the change score.
2. Order parameter for *AR order* : How far past data points are included in the model
3. Smoothing *window smooth*: The greater this parameter is, the easier it is to capture the essential changes rather than the outliers, but if it is too large, it will be difficult to capture the changes themselves

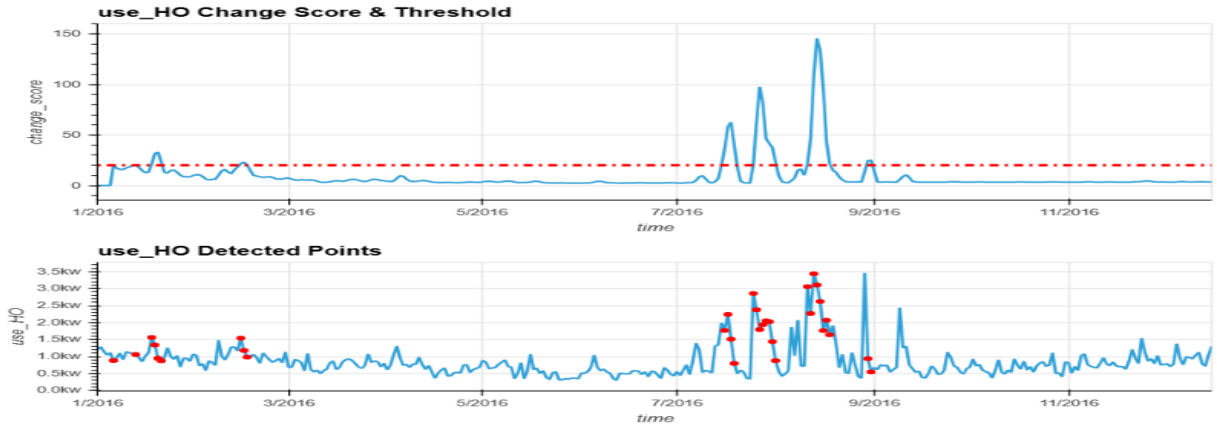

Fig. 1. House Overall Consumption - Detected Points and Threshold

Case2. Predict Future Energy Consumption [3]

- It is considered possible to predict the energy consumption of each appliance from the weather information.
- By predicting energy consumption, it is possible to estimate the amount of required energy supply based on the weather information, which can lead to energy **optimization**.

Model: LightGBM Regressor

Building a time-series regression model with LightGBM, we can predict future energy consumption and understand the relationship between energy consumption and weather information.

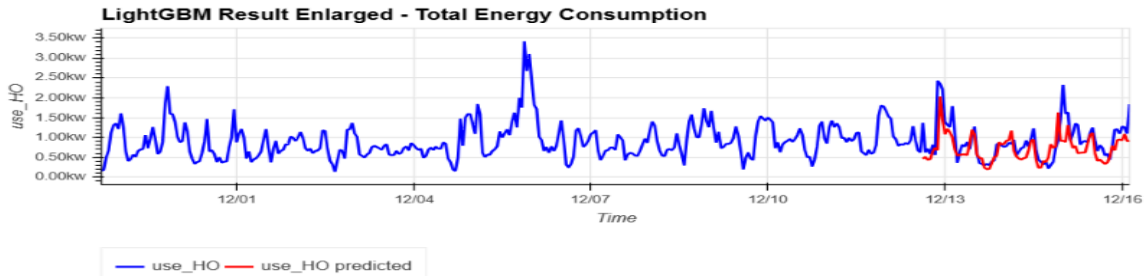

Fig. 2. Total Energy Consumption using LightGBM shown for each date.

EXPLAINABILITY THROUGH SHAP ANALYSIS [4] [5] [6]

- In our SHAP feature analysis, we observed significant impacts on total energy consumption predictions. Positive changes in features like **'weekofyear'**, **'timing'**, **'hour'**, **'dewPoint'**, and **'temperature'** tended to increase energy consumption predictions. Conversely, negative changes in features such as **'weekday'** and **'cloudCover'** were associated with a decrease in predicted energy usage. These insights highlight the nuanced relationship between time-specific elements and environmental conditions with energy consumption patterns.

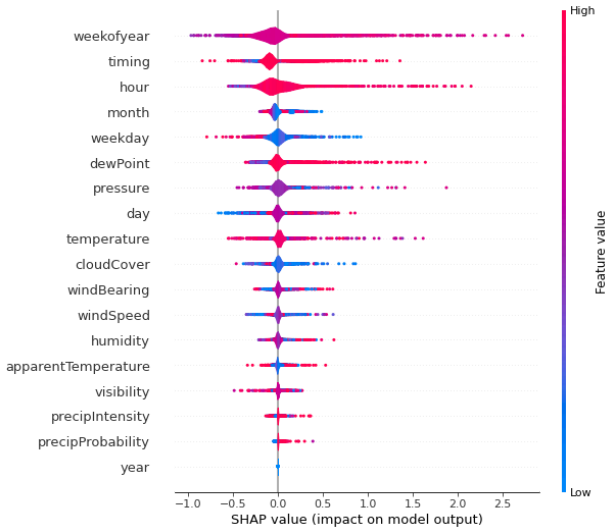

Fig. 3. SHAP values of various features

Target	LightGBM Regressor
Total Energy Consumption	0.282046

Table 1. The total energy consumption that is predicted using LightGBMRegressor.

INTEGRATION WITH LLM

The integration of SHAP values with Large Language Models (LLMs) involved a comprehensive, structured approach:

1. **Feature Identification**:
   a. We determined the input features for SHAP analysis. Typically, in the context of LLMs, these features included tokens, phrases, or embeddings from the input text.
   b. We treated each feature as a player in a cooperative game, where the prediction was the payout, enabling a game-theoretic approach to understanding feature impact.

2. **Computation of SHAP Values**:
   a. We employed the SHAP library in Python to calculate the contribution of each identified feature to the model's predictions. This calculation involved contrasting the predictions made with and without each feature present.

b. For sequence-based models like GEMINI, we utilized adaptations of the SHAP library designed to handle the intricate dependencies between input features, ensuring accurate SHAP value computation.

3. **Visualization and Analysis**:
   a. Visualization tools from the SHAP library were implemented to graphically represent the impact of each feature(as shown in Fig.3). Techniques such as force plots were used, which illustrated how each feature influenced the model's base value (the output without any features) to arrive at the final prediction.
   b. These detailed visualizations were instrumental in helping stakeholders comprehend the model's reasoning process, aligning model predictions with intuitive human understanding.

Conclusion

- There is a certain trend in the amount of energy consumed by each home appliance.
- The ChangeFinder model we created caught trend changes in energy consumption at an early stage.
- It turns out that models built with **LightGBM** can predict future energy consumption.
- It was also found that weather information and time information are very useful for the prediction.

2. RAG - Fine Tuning - Knowledge Bases

Often in the energy domain, we see a lot of documentation involving manuals, usage instructions or energy ratings and limitations. The bigger the system, the more convoluted the documentation and thus more difficulty finding the answer. To address this, we implemented the Retrieval Augmented Generation [7].

We take all the knowledge sources like pdf documents, websites links and other information from the internet and then create a knowledge store. We use a query embedding model and extract the embedding for all the information that we have sentence by sentence. This is stored in a vector database, constituting the knowledge base.

When a query is posed, we take the embedding of the query and then do a vector search in the previous database to find the most matching information pertaining to this using vector similarity. The corresponding sentence should have the answer we are looking for.

Now we have the question and a sentence or group of sentences that has the answer in it. But this answer is not in the format the query expected. This is where the LLM comes into play. We give the query, the set of sentences which contain the answer, the output from our Retrieval step and send it to the LLM model for augmentation. The LLM model will thus generate the output in the format required by the query.

RAG

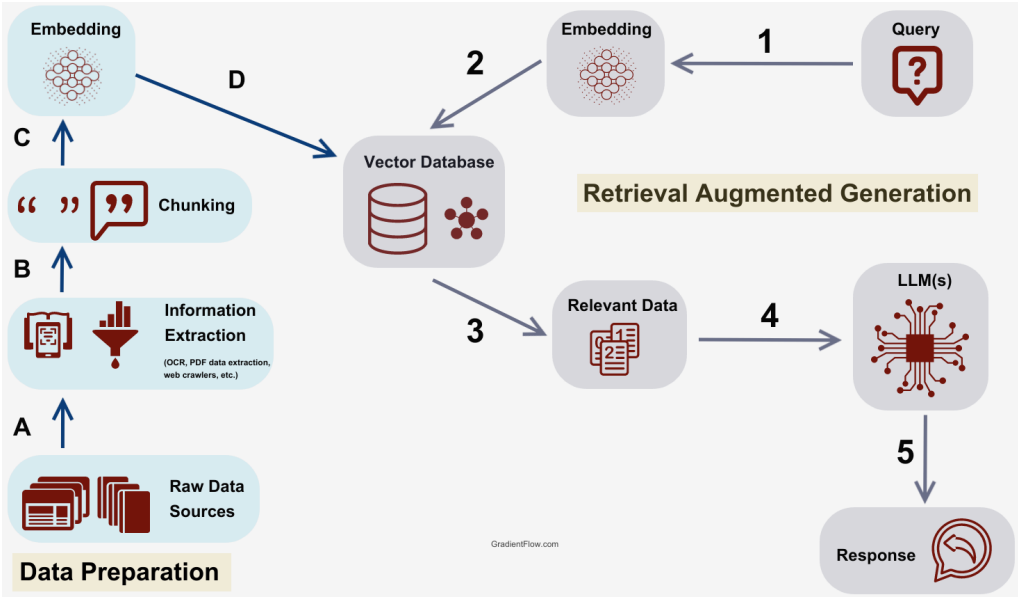

Fig. 4. The work flow of RAG.

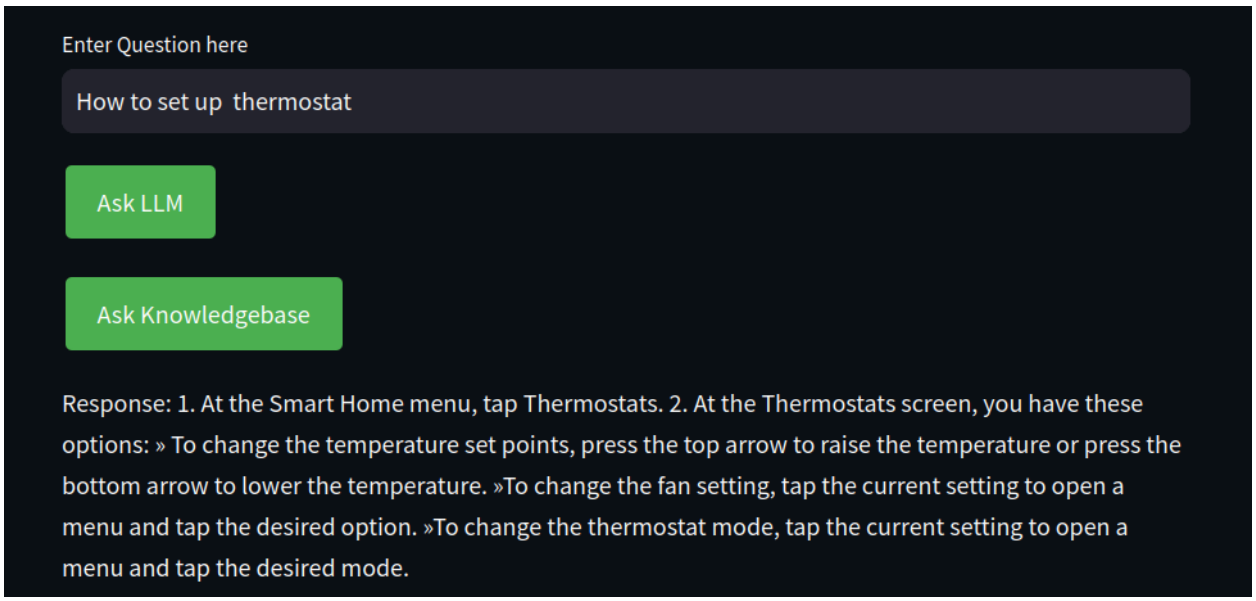

Fig. 5 Example from the Application.

Implementation:

We used both open source and enterprise models to give the user the ability to pick the kind he/she prefers. For Enterprise, we implemented gemini embeddings and added both Gemini and OpenAI implementation. For open source, we used Instructor Model[8] to take out the

embeddings and augmented generation with all of LLama 2, LLama 3, and Gemma. We plan to add Mistral and Phi capability in the future.

Fine Tuning

In the phase of building the knowledge base model, we realized that the FAQ websites on energy systems would have a gold mine of instruction tuned examples that could be used for answering regular questions. If we could somehow embed this information as part of a common model, without the requirement of special knowledge bases, we can add the capability to answer so many questions out of the box, which would be as specific to the product as we need.

For this purpose we picked up a couple of FAQ examples from Government websites and formatted it as a Human Assistant query answer interaction to be used for funtuning.

For fine tuning we used the following:

1. The accelerate library for parallelisation and adaptation to the available config [9]
2. PEFT for limited finetuning- Performance Efficient Fine Tuning [10]
3. Transformer- to both run the pretrained model and to fine tune with other downstream tasks [11]
4. TRL for supervised fine tuning trainer. [12]
5. QLoRA- The quantized Low Rank adapter used to fine tune model by taking it in the quantized format and then fine tuning with the Low Rank Adapters [13]

We used the libraries and did a performance efficient fine tuning of the quantized models for the small dataset we have. This approach is expected to both bring down the size of the model we run every time, as well have answers to some of the specific questions by default.

Although our fine tuned model was giving out nice answers for questions similar to the ones in our dataset, it would never have good answers for extrapolated questions. But we believe with enough data, the effort would be justified for the kind of output the model will give.

3. Optimal Device Placement

Overview:

We implemented multi-modal techniques to identify devices within room images and offer personalized suggestions for optimization. By harnessing the capabilities of LLMs (Gemini Vision Pro) across these diverse usage scenarios, our Smart Home Assistant endeavors to provide intelligent, context-aware assistance tailored to the needs of users, fostering efficiency, convenience, and sustainability within the smart home environment.

Implementation:

We feed a room image with multiple devices to the Gemini Vision Pro. The image is then processed and analyzed to identify the current layout of the devices within the room. The LLM identifies potential problems in the layout, such as suboptimal device placements that could lead to inefficiencies or reduced device functionality and gives device placement recommendations.

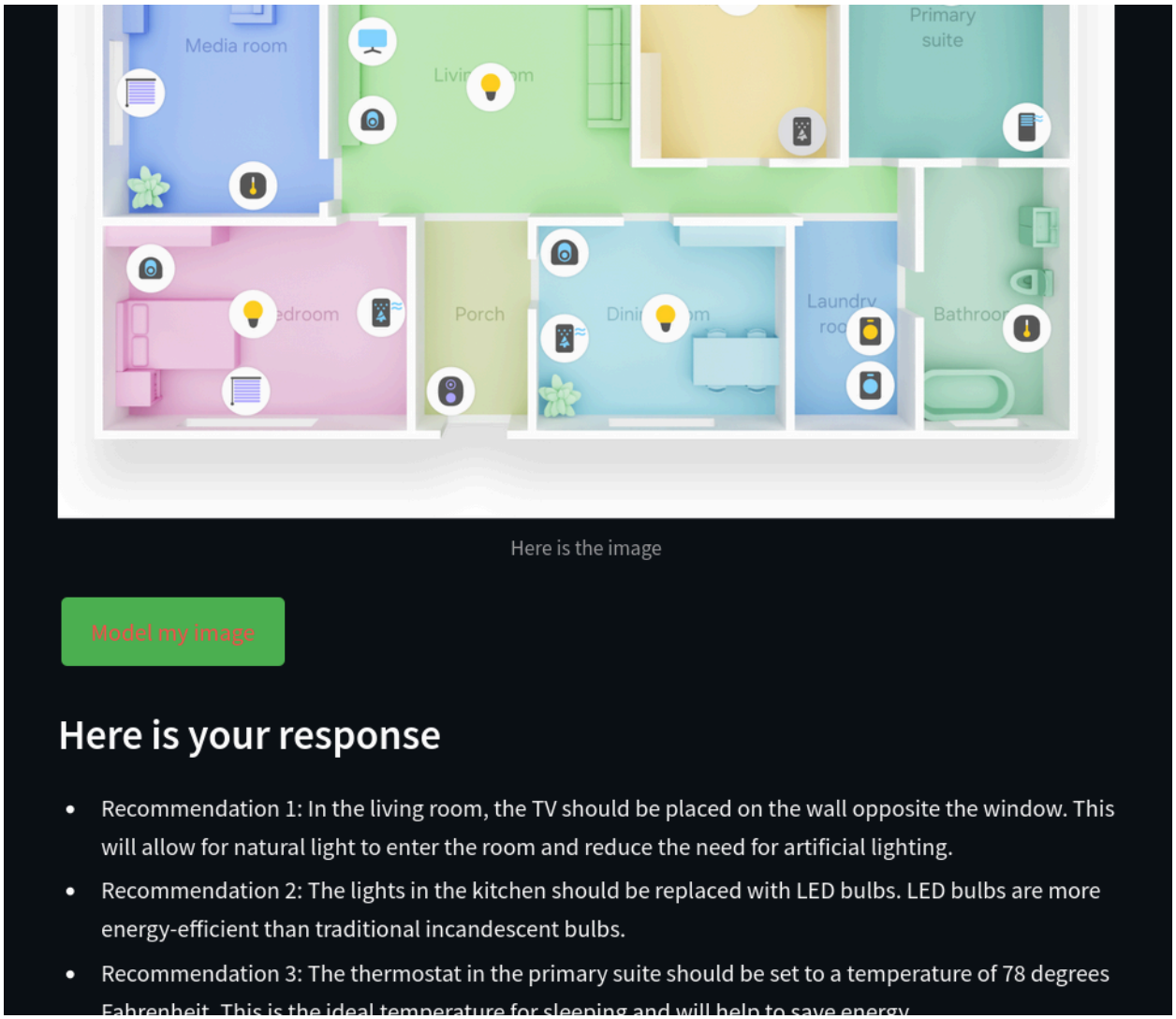

Fig. 6. Optimal Device Placement Recommendations Example for a Given Image Layout.

Professor's Remarks and Future Work:

Professor Liu was interested in the image modeling part of the project. Here we are taking the image and with prompt embedding and conversation chain buffers, we are making the LLM model to give out the suggestions along with the explainability. There is no training involved here. But we believe this can be worked on. Although there is no technical document outlining the information on how google does it, we can make an educated guess on how other Large Multimodal developers have done it, take independent encoders for each modality to generate embeddings, but how they set up the cross attention with the decoder is something we have no knowledge about. But we believe masked image modeling, similar to how its done with texts can be improved upon and with a very specific dataset, can be made to learn the layout and its corresponding recommendations.

With usage of better LLMs, in this project we have implemented different LLMs and have compared their relative performances. In the forecasting and anomaly detection model, we have used LLMs for explainability extension. After running different models, we have seen similar results across different models. What this implies is that for this purpose, the comparative model performance doesn't matter. Further we believe that for explainability purposes we can actually build a smaller model and then augment it with this capability, rather than running the whole LLM. This would extend for the RAG model as well. We can use a simpler model to build the augmentation, which answers questions in the context.

With fine tuning, we saw that models like LLama 2 were giving out good results, but the gemma model wasn't performing to a good extent. But we would withhold from commenting anything on this since our dataset was very small and a proper reasoning for the behavior cannot be reached.

SOURCE CODE

https://github.com/abhihash01/SmartHome-Savant-LLM

REFERENCES

[1] ChangeFinder Library Usage. Available at: http://argmax.jp/index.php?changefinder

[2] Yamanishi, K., & Takeuchi, J. (2002). A Unifying Framework for Detecting Outliers and Change Points from Time Series. IEEE Transactions on Knowledge and Data Engineering.

[3] Malekzadeh, M. (Date). Smart Home Data Processing: Weather vs Energy.

[4] Lundberg, S. M., & Lee, S.-I. (2017). A Unified Approach to Interpreting Model Predictions. NIPS. Available at: http://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions.pdf

[5] Lundberg, S. M., et al. (2018). Consistent Individualized Feature Attribution for Tree Ensembles. ArXiv. Available at: https://arxiv.org/pdf/1802.03888.pdf

[6] SHAP Library Documentation. Available at: https://shap.readthedocs.io/en/latest/api.html

[7] Retrieval Augmented Generation https://arxiv.org/abs/2312.10997

[8] Instructor XL embedding model. https://huggingface.co/hkunlp/instructor-xl

[9] Accelerate https://huggingface.co/docs/accelerate/index

[10] PEFT- Performance Efficient Fine Tuning

[11] Transformers https://huggingface.co/transformers/v3.0.2/model_doc/auto.html

[12] SFT trainer https://huggingface.co/docs/trl/main/en/sft_trainer

[13] QLoRA https://huggingface.co/blog/4bit-transformers-bitsandbytes

[14] https://www.kaggle.com/datasets/taranvee/smart-home-dataset-with-weather-information/data