Analyzing Energy Efficiency in Buildings

Rohan Dasgupta

BASIS Independent Silicon Valley, 1290 Parkmoor Avenue, San Jose, 95126, CA, United States.

Corresponding author(s). E-mail(s): rohan.dasgupta69420@gmail.com;

Abstract

Energy efficiency is a crucial issue impacting the lives of people around the world. Over the past few years, we have all encountered the adverse effects of high energy costs in our personal households as well as the global economy. The application of Artificial Intelligence and Machine Learning (AI/ML) on Internet of Things (IoT) data in the context of improving energy efficiency of buildings holds immense potential. This paper aims to utilize the power of AI/ML and IoT to devise predictions and provide recommendations to improve the energy efficiency of buildings. The concept of "smart buildings" has recently become a key focus of AI/ML research, wherein the building collects energy consumption data from devices and sensors to analyze energy usage and provide concrete recommendations for enhancing the energy efficiency of a building. Similar research has been undertaken in improving energy efficiency in smart buildings through retrofitting intervention via IoT components in buildings.

Keywords: Systems Software, Algorithms, Machine Learning, Smart Buildings

1 Introduction

Artificial Intelligence/Machine Learning (AI/ML) can be applied for Internet of Things (IoT) applications given the immense scope and impact the development and integration of these novel predictive approaches can have for society at large. Energy efficiency in buildings is a crucial global issue due to multiple ongoing factors, like global warming that has resulted in extreme weather conditions like extremely hot summers and frigid winters, and geopolitical instability due to events like the ongoing war in Ukraine, which has resulted in unsustainably high energy costs. The aim of the project is to develop predictive and control-based methods for optimizing energy consumption in buildings using novel techniques in supervised learning, energy simulation through the `EnergyPlus`™ program [1], and state-of-the-art deep reinforcement learning (RL) algorithms. Various components of this predictive energy control architecture in Python were implemented via simulation and AI tools like `LazyPredict` [2] and `RLlib` [3], and the final code was run in Google Colab notebooks utilizing the former platforms. The project utilized a DRL and Supervised Learning environment for learning predictive and efficient energy control in buildings obtained from the results of `EnergyPlus`™ simulations, gathering useful data through tables on the r-squared and RMSE) values as well as the time taken to complete its run. The trend is shifting toward "smart buildings" that manage the buildings' general functioning efficiently to minimize energy costs, and energy efficiency is a primary problem that the model is intent on solving. Through the use of IoT sensors and data processing involving AI and ML, the project aims to provide long-term cost efficiencies through the careful calibration of a building's energy infrastructure. In summary, this proposed solution will result in reduction of energy and related maintenance costs necessary to keep the building operational in an optimal, cost-efficient way.

On a personal level, I am motivated to pursue my current project due to the increasingly high energy bills encountered in my own home as well as the ongoing global energy crisis impacting economies around the world. I see this project as an effective and impactful way to utilize my interest in AI/ML and IoT technologies to come up with a smart way to improve energy efficiency of buildings. This project has a crucial real-world impact by coming up with practical solutions to the ongoing energy crisis that continues to affect our world.

2 Methods

I have always been very interested in Mathematics, Statistics, and Computer Programming since middle school, and have taken numerous courses in the field, like AP Calculus BC, AP Statistics, and AP Computer Science A, and I recently took post-AP Multivariable Calculus. To begin with, my project mentor and I came up with a concrete plan to translate my interests into a viable project. The project is broken into two key phases, that of training and testing the data.

We explored different IoT sensor datasets from different building types in Kaggle [4] and Keras [5]. I familiarized myself with Python libraries, and set up the Jupyter Notebook and Anaconda development environments. I evaluated various Machine Learning models and settled on using Reinforcement Learning as the most prominent predictive model for the project. I then evaluated Reinforcement Learning models like Ray RLLib for running Reinforcement Learning simulations.

We explored Linear Regression tools like LazyPredict. We also evaluated multiple simulation tools and chose EnergyPlus for its comprehensive array of building-specific features.

We also chose Energym [6] as a tool for running simulations. We used EnergyPlus to generate test examples and Energym to run the code, utilizing and modifying several variables to manipulate the data, and we also looked into the topic of Multi-Layer Perceptrons (MLPs) through several articles and examples to better understand Reinforcement Learning.

The end goal is to devise a model that will showcase the potential improvements in energy efficiency for a given building and allow the building operator to dynamically adjust their energy consumption, which I achieved by running simulations through collab notebooks and collecting the associated data (i.e. the r-squared and RMSE values).

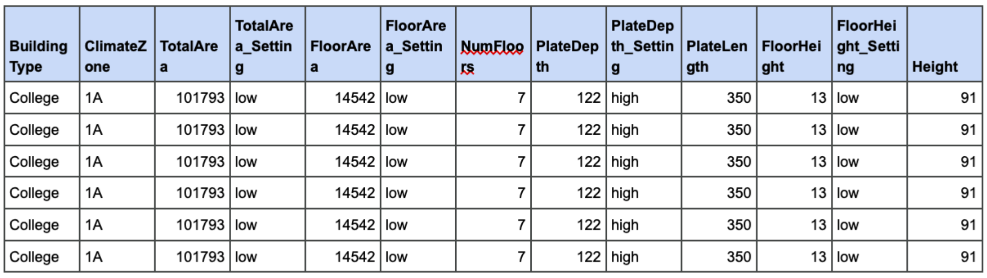
To summarize, one of the key goals of this study was to evaluate the models' performance relative to varying sample sizes of the data (i.e. 100, 1000, 10000, and all samples). This indicates the robustness of the predictive model when limited data is available for training the models. The test set was kept the same in all cases to ensure that comparisons are equal between different models with varying levels of training data.

For this study, the dataset contained key variables such as building type, HVAC type, floor-to-floor height, and number of floors as a representative sample. The building energy data input table from this dataset was randomly sampled down to 256K for computational feasibility.

3 Results

The project showcased a detailed supervised learning analysis of the building. The model used in the project was successful in qualifying the energy efficiency of the building's system and highlighting the correlation between the building's energy parameters and the performance evaluation metrics. The project utilized r-squared and RMSE values for the multiple regression models, as well as the individual times to generate the results. While the expectation was that the average RMSE values would decrease proportionally with the number of samples, the data from Table 4 indicates the opposite trend, as increasing the number of samples resulted in more error in the model's predictions.

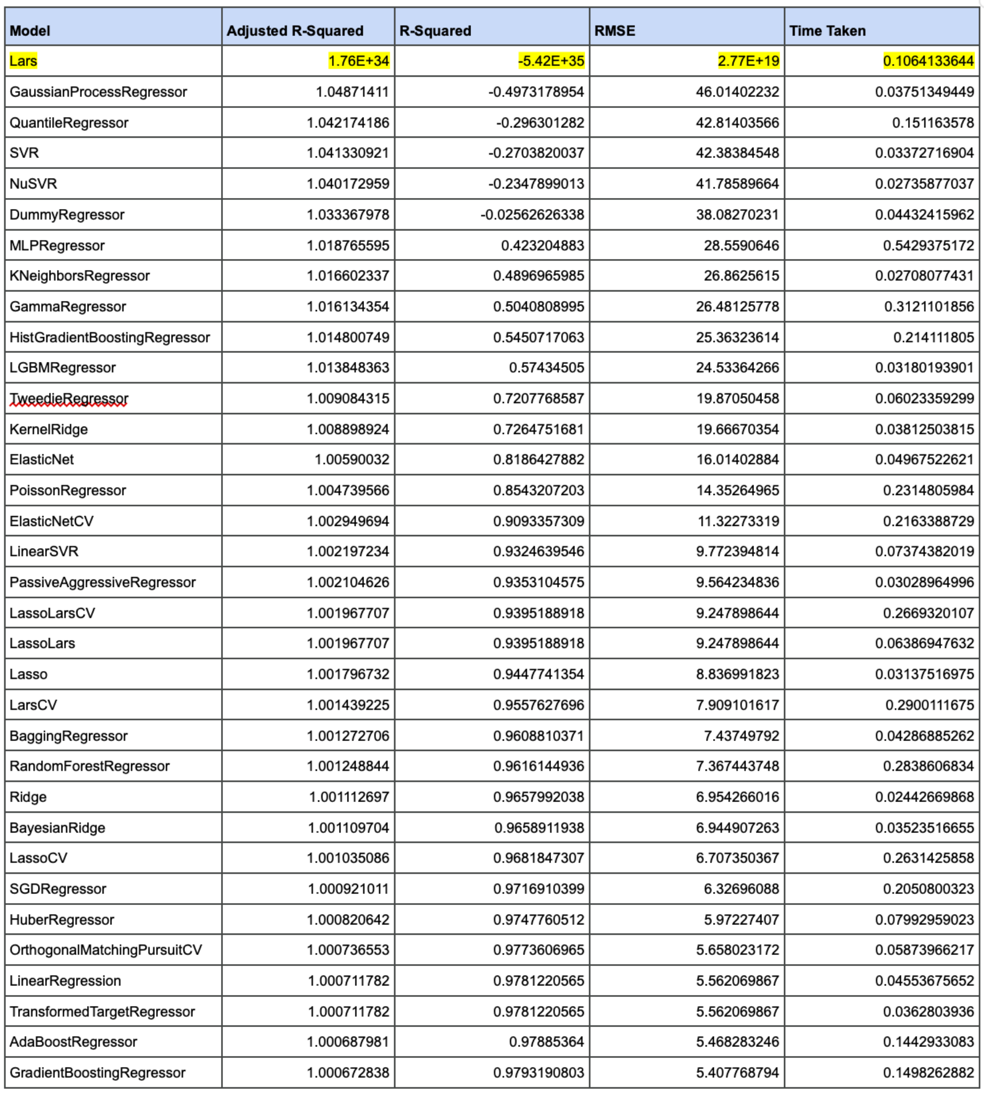
The model's efficiency was strengthened by having the model pre-process and inspect the data. In doing so, the model is able to identify the different parameters (e.g. the light efficiency, the HVAC outputs, the electricity efficiency, etc.) involved in determining the energy usage of the building data and analyze the relationships between them. Additionally, the model also filters out the aspects it deems unnecessary to its algorithm.



Building Type	ClimateZone	TotalArea	TotalArea_Setting	FloorArea	FloorArea_Setting	NumFloors	PlateDepth	PlateDepth_Setting	PlateLength	FloorHeight	FloorHeight_Setting	Height
College	1A	101793	low	14542	low	7	122	high	350	13	low	91
College	1A	101793	low	14542	low	7	122	high	350	13	low	91
College	1A	101793	low	14542	low	7	122	high	350	13	low	91
College	1A	101793	low	14542	low	7	122	high	350	13	low	91
College	1A	101793	low	14542	low	7	122	high	350	13	low	91
College	1A	101793	low	14542	low	7	122	high	350	13	low	91

Fig. 1 Sample building energy data.

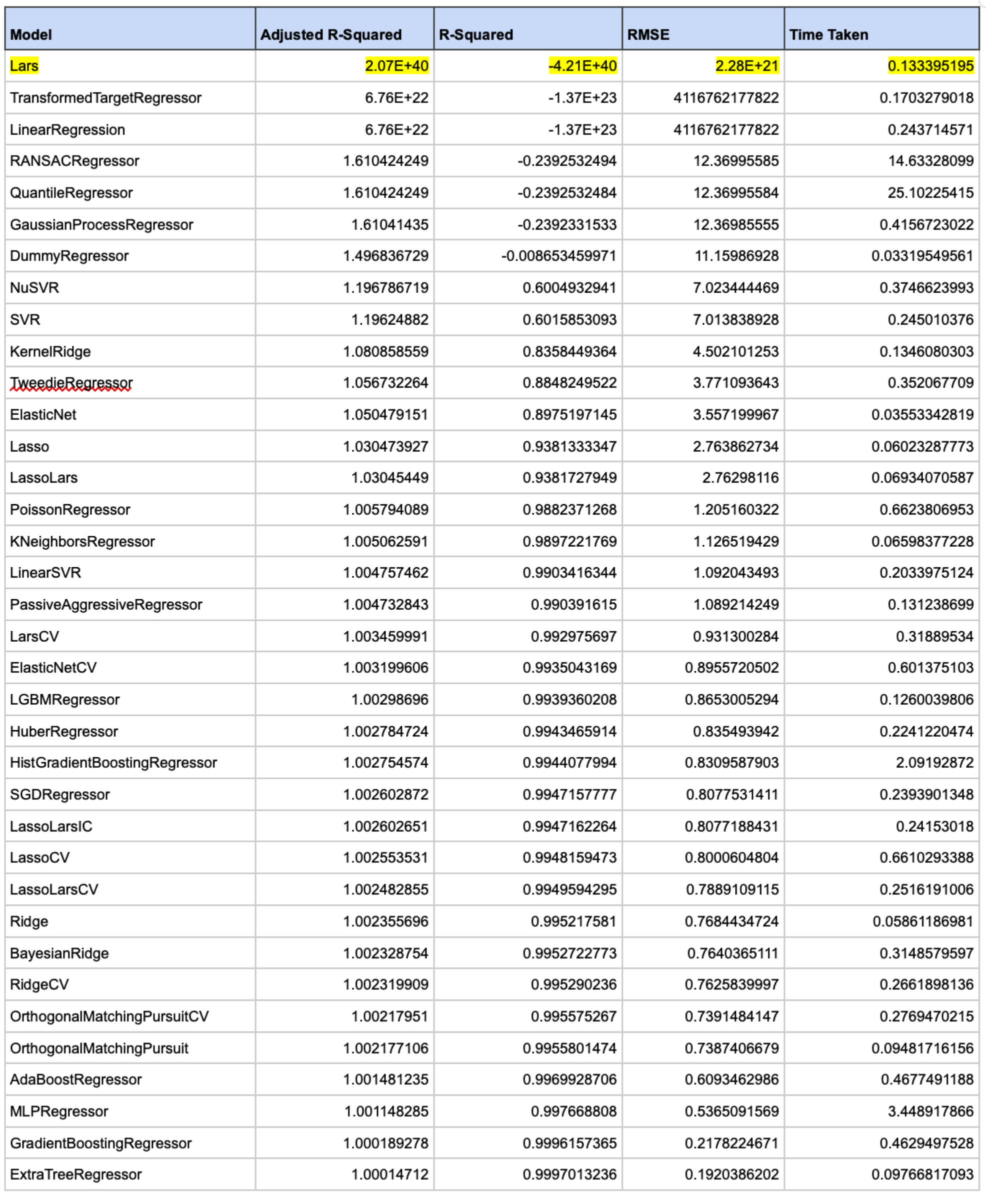
The sample building energy data in Figure 1 above provides the energy consumption of a typical building and captures consumption data from HVAC, Sensors, etc. The machine learning model specifically focuses on categorical variables in order to perform its numeric training and eventually read the regression models, which require numerical data. After processing the categorical variables and the associated keys (column headers), the machine learning model splits the data into independent (x-) and dependent (y-) variables. Later, it is fed the regression models and reports the cross-validation (CV) scores and the RMSE values. Finally, the machine learning model reports the associated r-squared and RMSE values, as well as the time taken to obtain said values, as displayed in Figures 2, 3, and 4. The best-performing model is highlighted in each of these figures.



Model	Adjusted R-Squared	R-Squared	RMSE	Time Taken
Lars	1.76E+34	-5.42E+35	2.77E+19	0.1064133644
GaussianProcessRegressor	1.04871411	-0.4973178954	46.01402232	0.03751349449
QuantileRegressor	1.042174186	-0.296301282	42.81403566	0.151163578
SVR	1.041330921	-0.2703820037	42.38384548	0.03372716904
NuSVR	1.040172959	-0.2347899013	41.78589664	0.02735877037
DummyRegressor	1.033367978	-0.02562626338	38.08270231	0.04432415962
MLPRegressor	1.018765595	0.423204883	28.5590646	0.5429375172
KNeighborsRegressor	1.016602337	0.4896965985	26.8625615	0.02708077431
GammaRegressor	1.016134354	0.5040808995	26.48125778	0.3121101856
HistGradientBoostingRegressor	1.014800749	0.5450717063	25.36323614	0.214111805
LGBMRegressor	1.013848363	0.57434505	24.53364266	0.03180193901
TweedieRegressor	1.009084315	0.7207768587	19.87050458	0.06023359299
KernelRidge	1.008898924	0.7264751681	19.66670354	0.03812503815
ElasticNet	1.00590032	0.8186427882	16.01402884	0.04967522621
PoissonRegressor	1.004739566	0.8543207203	14.35264965	0.2314805984
ElasticNetCV	1.002949694	0.9093357309	11.32273319	0.2163388729
LinearSVR	1.002197234	0.9324639546	9.772394814	0.07374382019
PassiveAggressiveRegressor	1.002104626	0.9353104575	9.564234836	0.03028964996
LassoLarsCV	1.001967707	0.9395188918	9.247898644	0.2669320107
LassoLars	1.001967707	0.9395188918	9.247898644	0.06386947632
Lasso	1.001796732	0.9447741354	8.836991823	0.03137516975
LarsCV	1.001439225	0.9557627696	7.909101617	0.2900111675
BaggingRegressor	1.001272706	0.9608810371	7.43749792	0.04286885262
RandomForestRegressor	1.001248844	0.9616144936	7.367443748	0.2838606834
Ridge	1.001112697	0.9657992038	6.954266016	0.02442669868
BayesianRidge	1.001109704	0.9658911938	6.944907263	0.03523516655
LassoCV	1.001035086	0.9681847307	6.707350367	0.2631425858
SGDRegressor	1.000921011	0.9716910399	6.32696088	0.2050800323
HuberRegressor	1.000820642	0.9747760512	5.97227407	0.07992959023
OrthogonalMatchingPursuitCV	1.000736553	0.9773606965	5.658023172	0.05873966217
LinearRegression	1.000711782	0.9781220565	5.562069867	0.04553675652
TransformedTargetRegressor	1.000711782	0.9781220565	5.562069867	0.0362803936
AdaBoostRegressor	1.000687981	0.97885364	5.468283246	0.1442933083
GradientBoostingRegressor	1.000672838	0.9793190803	5.407768794	0.1498262882

Fig. 2 Model performance evaluation metrics for 100-sample experiment.

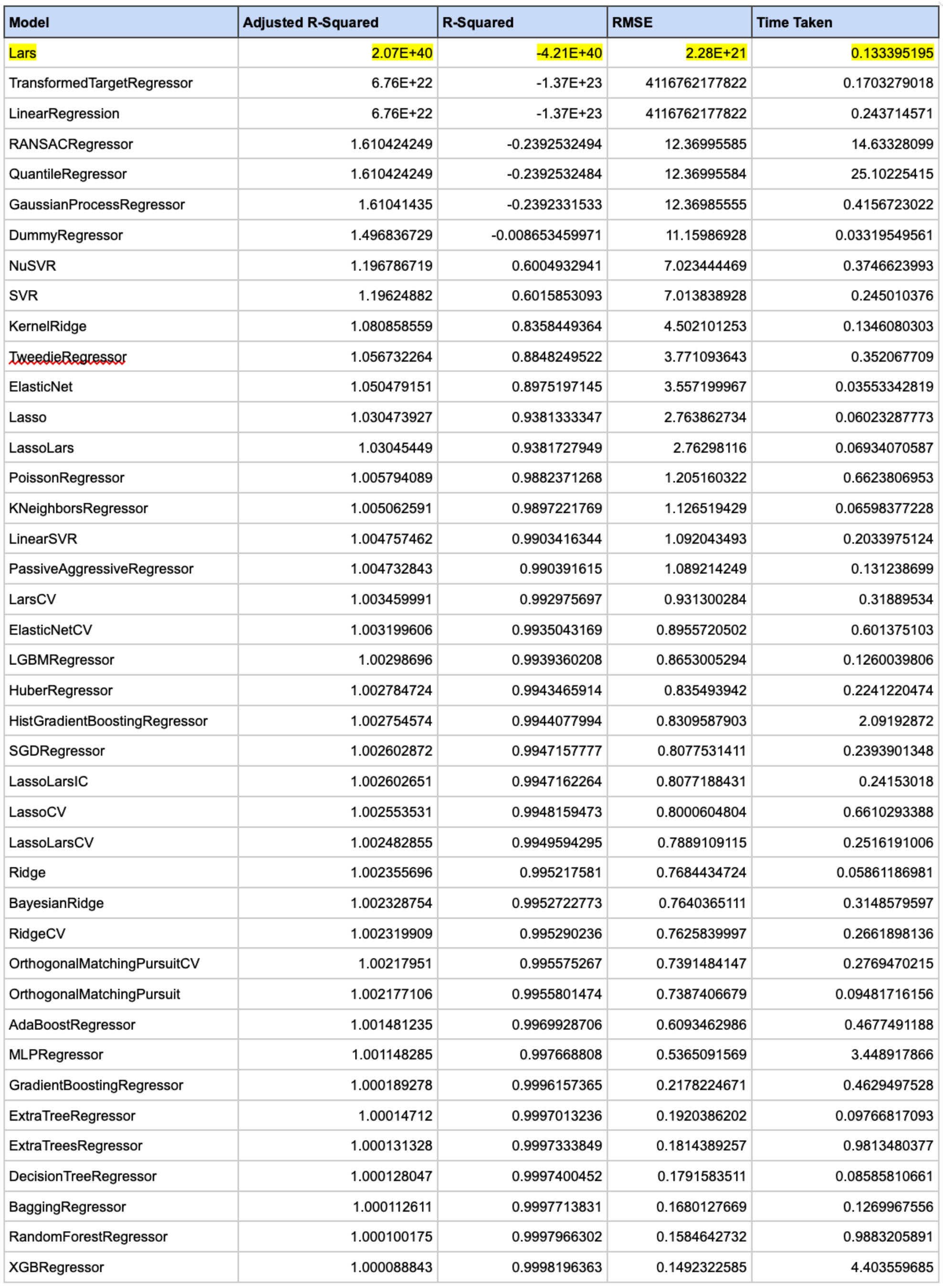
The outputs of the different regression models were collected for 100 samples. The Lars model is highlighted due to holding the highest correlation values, indicating a strong model for the data (Figure 2), as it produces a full piecewise linear path, allowing for a stronger correlation.



Model	Adjusted R-Squared	R-Squared	RMSE	Time Taken
Lars	2.07E+40	-4.21E+40	2.28E+21	0.133395195
TransformedTargetRegressor	6.76E+22	-1.37E+23	4116762177822	0.1703279018
LinearRegression	6.76E+22	-1.37E+23	4116762177822	0.243714571
RANSACRegressor	1.610424249	-0.2392532494	12.36995585	14.63328099
QuantileRegressor	1.610424249	-0.2392532484	12.36995584	25.10225415
GaussianProcessRegressor	1.61041435	-0.2392331533	12.36985555	0.4156723022
DummyRegressor	1.496836729	-0.008653459971	11.15986928	0.03319549561
NuSVR	1.196786719	0.6004932941	7.023444469	0.3746623993
SVR	1.19624882	0.6015853093	7.013838928	0.245010376
KernelRidge	1.080858559	0.8358449364	4.502101253	0.1346080303
TweedieRegressor	1.056732264	0.8848249522	3.771093643	0.352067709
ElasticNet	1.050479151	0.8975197145	3.557199967	0.03553342819
Lasso	1.030473927	0.9381333347	2.763862734	0.06023287773
LassoLars	1.03045449	0.9381727949	2.76298116	0.06934070587
PoissonRegressor	1.005794089	0.9882371268	1.205160322	0.6623806953
KNeighborsRegressor	1.005062591	0.9897221769	1.126519429	0.06598377228
LinearSVR	1.004757462	0.9903416344	1.092043493	0.2033975124
PassiveAggressiveRegressor	1.004732843	0.990391615	1.089214249	0.131238699
LarsCV	1.003459991	0.992975697	0.931300284	0.31889534
ElasticNetCV	1.003199606	0.9935043169	0.8955720502	0.601375103
LGBMRegressor	1.00298696	0.9939360208	0.8653005294	0.1260039806
HuberRegressor	1.002784724	0.9943465914	0.835493942	0.2241220474
HistGradientBoostingRegressor	1.002754574	0.9944077994	0.8309587903	2.09192872
SGDRegressor	1.002602872	0.9947157777	0.8077531411	0.2393901348
LassoLarsIC	1.002602651	0.9947162264	0.8077188431	0.24153018
LassoCV	1.002553531	0.9948159473	0.8000604804	0.6610293388
LassoLarsCV	1.002482855	0.9949594295	0.7889109115	0.2516191006
Ridge	1.002355696	0.995217581	0.7684434724	0.05861186981
BayesianRidge	1.002328754	0.9952722773	0.7640365111	0.3148579597
RidgeCV	1.002319909	0.995290236	0.7625839997	0.2661898136
OrthogonalMatchingPursuitCV	1.00217951	0.995575267	0.7391484147	0.2769470215
OrthogonalMatchingPursuit	1.002177106	0.9955801474	0.7387406679	0.09481716156
AdaBoostRegressor	1.001481235	0.9969928706	0.6093462986	0.4677491188
MLPRegressor	1.001148285	0.997668808	0.5365091569	3.448917866
GradientBoostingRegressor	1.000189278	0.9996157365	0.2178224671	0.4629497528
ExtraTreeRegressor	1.00014712	0.9997013236	0.1920386202	0.09766817093

Fig. 3 Model performance evaluation metrics for 1000-sample experiment.

The outputs of the different regression models were collected for 1000 samples in order to gain more varied/accurate outputs (Figure 3). Again, the Lars model performed the highest, as the model allows for higher correlation due to its generation of a piecewise linear path. The only two models that approached the Lars model's efficiency were TransformedTargetRegressor and LinearRegression. The former generates a linear prediction on a non-linear path and the latter simply tries to find a best fit for the data, regardless of the actual shape, and as a result they don't capture the same level of efficiency as the Lars model.



Model	Adjusted R-Squared	R-Squared	RMSE	Time Taken
Lars	2.07E+40	-4.21E+40	2.28E+21	0.133395195
TransformedTargetRegressor	6.76E+22	-1.37E+23	4116762177822	0.1703279018
LinearRegression	6.76E+22	-1.37E+23	4116762177822	0.243714571
RANSACRegressor	1.610424249	-0.2392532494	12.36995585	14.63328099
QuantileRegressor	1.610424249	-0.2392532484	12.36995584	25.10225415
GaussianProcessRegressor	1.61041435	-0.2392331533	12.36985555	0.4156723022
DummyRegressor	1.496836729	-0.008653459971	11.15986928	0.03319549561
NuSVR	1.196786719	0.6004932941	7.023444469	0.3746623993
SVR	1.19624882	0.6015853093	7.013838928	0.245010376
KernelRidge	1.080858559	0.8358449364	4.502101253	0.1346080303
TweedieRegressor	1.056732264	0.8848249522	3.771093643	0.352067709
ElasticNet	1.050479151	0.8975197145	3.557199967	0.03553342819
Lasso	1.030473927	0.9381333347	2.763862734	0.06023287773
LassoLars	1.03045449	0.9381727949	2.76298116	0.06934070587
PoissonRegressor	1.005794089	0.9882371268	1.205160322	0.6623806953
KNeighborsRegressor	1.005062591	0.9897221769	1.126519429	0.06598377228
LinearSVR	1.004757462	0.9903416344	1.092043493	0.2033975124
PassiveAggressiveRegressor	1.004732843	0.990391615	1.089214249	0.131238699
LarsCV	1.003459991	0.992975697	0.931300284	0.31889534
ElasticNetCV	1.003199606	0.9935043169	0.8955720502	0.601375103
LGBMRegressor	1.00298696	0.9939360208	0.8653005294	0.1260039806
HuberRegressor	1.002784724	0.9943465914	0.835493942	0.2241220474
HistGradientBoostingRegressor	1.002754574	0.9944077994	0.8309587903	2.09192872
SGDRegressor	1.002602872	0.9947157777	0.8077531411	0.2393901348
LassoLarsIC	1.002602651	0.9947162264	0.8077188431	0.24153018
LassoCV	1.002553531	0.9948159473	0.8000604804	0.6610293388
LassoLarsCV	1.002482855	0.9949594295	0.7889109115	0.2516191006
Ridge	1.002355696	0.995217581	0.7684434724	0.05861186981
BayesianRidge	1.002328754	0.9952722773	0.7640365111	0.3148579597
RidgeCV	1.002319909	0.995290236	0.7625839997	0.2661898136
OrthogonalMatchingPursuitCV	1.00217951	0.995575267	0.7391484147	0.2769470215
OrthogonalMatchingPursuit	1.002177106	0.9955801474	0.7387406679	0.09481716156
AdaBoostRegressor	1.001481235	0.9969928706	0.6093462986	0.4677491188
MLPRegressor	1.001148285	0.997668808	0.5365091569	3.448917866
GradientBoostingRegressor	1.000189278	0.9996157365	0.2178224671	0.4629497528
ExtraTreeRegressor	1.00014712	0.9997013236	0.1920386202	0.09766817093
ExtraTreesRegressor	1.000131328	0.9997333849	0.1814389257	0.9813480377
DecisionTreeRegressor	1.000128047	0.9997400452	0.1791583511	0.08585810661
BaggingRegressor	1.000112611	0.9997713831	0.1680127669	0.1269967556
RandomForestRegressor	1.000100175	0.9997966302	0.1584642732	0.9883205891
XGBRegressor	1.000088843	0.9998196363	0.1492322585	4.403559685

Fig. 4 Model performance evaluation metrics for 10000-sample experiment.

The outputs of the different regression models were collected for 10000 samples in order to gain more varied/accurate outputs (Figure 4). As predicted, the Lars model performed the highest due to its piecewise linear structure, with the TransformedTargetRegressor and LinearRegression models trailing behind, as the number of samples had exceeded the models' threshold.

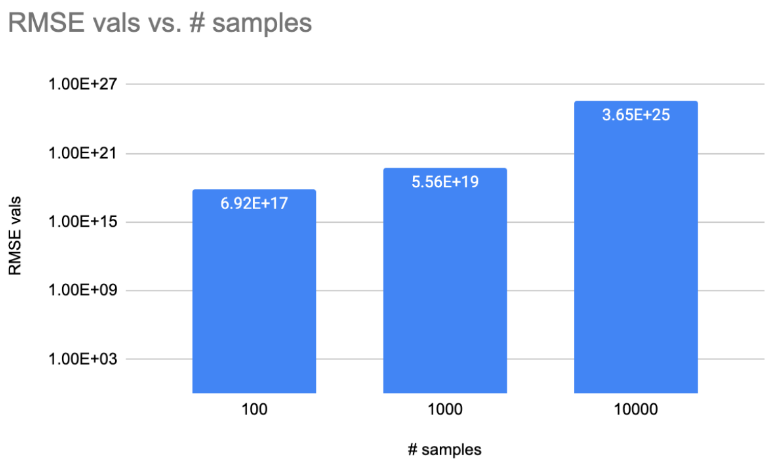


Fig. 5 Least Angle Regression (LARS) RMSE vs. # samples bar chart.

The average RMSE values tend to increase as the sample size increases (Figure 5). This is likely due to the increase in samples generating more error in the model's predictions. It is possible that the sheer number of samples and data within them exceeded the LARS model's threshold.

4 Conclusion

To summarize, this data and its analysis come from an exhaustive supervised learning study. The system performed an exhaustive analysis of the energy consumption of a building. The key takeaway was that a successful model was created that qualified the levels of energy efficiency in buildings and provided accurate readings. The model also reported the r-squared and RMSE values for each regression model, as well as the time taken to produce the results for each model. One interesting observation from the data in Figure 1 plotting the average RMSE values against the number of samples is that the average RMSE values tended to increase with the number of samples, indicating that increasing the number of samples produced more errors in the model's predictions, the opposite of what was initially expected.

As we can infer, the data outputs from these models can be further enhanced in addressing the broader goal of providing concrete recommendations for potential appliances involved in the structure of the building. Therefore, the model can be strengthened with more precise data, significantly larger sample sizes, detailed readings, and interaction with real-time data from actual physical appliances.

Acknowledgments. I would like to thank my Mentor, Ryan Sander, for guiding me through the concepts and making this project possible.

References

[1] Crawley, D. B. *et al.* EnergyPlus: creating a new-generation building energy simulation program. *Energy Build.* **33**, 319–331 (2001).

[2] Pandala, S. R. Lazypredict. https://github.com/shankarpandala/lazypredict (2022).

[3] Liang, E. *et al.* RLlib: Abstractions for distributed reinforcement learning 3053–3062 (2017).

[4] Bass, B., Curtis, L. & McNally, P. Universal design space building energy simulation (2021). URL https://www.kaggle.com/dsv/2741357.

[5] Chollet, F. keras. https://github.com/fchollet/keras (2015).

[6] Scharnhorst, P. *et al.* Energym: A building model library for controller benchmarking. *Appl. Sci. (Basel)* **11**, 3518 (2021).