Selected Supervised Machine Learning Algorithm Evaluations on Classification Tasks

Mingyang Yao
Department of Cognitive Science
University of California, San Diego
m5yao@ucsd.edu

***Abstract*—In this study, a thorough examination was conducted utilizing five distinct machine-learning models applied to four disparate datasets. The principal aim was to employ correlation-based feature selection methodologies and implement parameter adjustments to discern and assess the influence of pertinent features on model outcomes. The selected models encompass Logistic Regression, Decision Tree, Random Forest, K-Nearest Neighbors, and Gradient Boosting Classifier, while the datasets span fields in education, banking, services, and letter recognition.**

The methodological approach herein involves the utilization of correlation coefficients to quantitatively express relationships between features and target variables, thereby facilitating the identification of influential features pertinent to the fitting of targets. The efficacy of this methodology is explored in terms of its impact on enhancing model accuracy, mitigating overfitting, and augmenting generalization across diverse datasets.

***Index Terms*—Machine Learning, GridSearch, Correlation Analysis**

I. Introduction

The classification tasks considered in this investigation possess a wide array of practical applications, particularly in the realms of decision-making and object recognition, spanning an extensive historical continuum. Over an extended period, machine learning engineers have dedicated substantial effort to experimenting with diverse models and selecting features to optimize model performance. This study strategically opts for five widely acknowledged and effective machine learning algorithms, as delineated in the paper titled "An Empirical Comparison of Supervised Learning Algorithms" by Caruana and Niculescu-Mizil, supplemented by my own empirical investigations into the performance of distinct machine learning models on the dataset.

Building upon the foundational insights gleaned from the prior work of Caruana and Niculescu-Mizil, as well as our independent analyses on various datasets, the five machine learning algorithms selected for evaluation in this study are *Logistic Regression, Decision Tree Classifier, Random Forest Classifier, and Gradient Boosting Classifier*. Additionally, we employ encoding techniques to adapt certain continuous regression tasks, ensuring that learning tasks are either inherently binary or are transformed into binary classifications (solely positive and negative). In tandem with presenting all features to the machine learning model, we conduct correlation analysis, identifying features whose correlations with target values exceed a pre-defined threshold, thereby informing the model training process. This methodological approach facilitates the discernment of influential features specific to each target feature set. The effectiveness of this approach is systematically investigated in terms of its impact on model accuracy enhancement, overfitting reduction, and generalization improvement across diverse datasets.

II. Methods

A. Package and Dataset Used

We have used the Python package sci-kit learn as the machine learning algorithm implementations, and for data reading, preprocessing, encoding, and result display, we all use pandas DataFrame to accomplish them. For the visualization of model performance, correlations, and confusion matrix on the test set, we utilized Seaborn and matplotlib.pyplot to make graphs. Some datasets are imported from the *fetch_ucirepo* module of UCI Dataset Repository *ucimlrepo*, and some datasets are downloaded as csv files from the UCI Dataset Repository website. Four datasets used to evaluate in this study are Bank Marketing [2], math data in Student Performance [3], Room Occupancy Estimation [4] as well as Letter Recognition [5].

B. Model Selection

According to the empirical study done by Caruana and NiculescuMizil, decision trees and random forests have generally higher performance in various classification tasks. In addition, we select logistic regression and the K-Nearest Neighbor (KNN) Classifier as two popular but relatively not as good as the previous two models in our evaluations. Finally, sci-kit learn package provides a gradient-boosting classifier, which enables the optimization of the loss function and a combination of multiple decision trees, further reducing the risk of overfitting as depth gets larger [6]. We took advantage of this classifier when evaluating.

C. Dataset Preprocessing and Feature Selection

I initiated the data processing pipeline by initially reading or importing the data into a pandas DataFrame format, subsequently scrutinizing the dataset for the presence of NaN values in its rows. In the event that a column exhibits null values for no more than 30% of the total data points, I systematically exclude all rows with null values in said column.

This process is implemented to preclude any confounding factors, and the imposition of a 30% threshold ensures that the dataset maintains its impartiality throughout the model fitting procedure. Subsequently, I executed One-Hot Encoding on all categorical features, configuring the drop_first parameter to be True. This operation yields a DataFrame comprising both numerical features and one-hot encoded categorical features. It is noteworthy that in the encoding process, one column per categorical feature is discarded to remove redundant information.

Following this initial preprocessing stage, a correlation matrix is computed, encapsulating the pairwise correlation coefficients among all columns. The correlation coefficients between features and the target variables are then specifically highlighted. The subsequent feature selection endeavor is predicated on a systematic exploration of diverse correlation thresholds. Only those features exhibiting correlations with the target surpassing a user-defined threshold are retained for inclusion in the model training process. This discerning feature selection strategy is designed to effectively isolate salient features, thereby forestalling overfitting concerns, particularly in instances where the volume of data points may be insufficient.

To ensure that the model we will train is binary classification, we manually divide possible target values into two groups if the dataset itself is not for binary classification. The exact methods of divisions and some slight differences in data preprocessing for each dataset will be introduced in the Experiment Section.

D. Fine-Tuning of Parameters

We use the GridSearch class provided by sci-kit learn package to perform 3-fold cross-validation and exhaustively search for the best-performance parameters for the model. For the five models used in this report, the hyperparameter combinations we have searched are listed in the following table. We obtain the best model and hyperparameter combination by directly using *best estimator* attribute of GridSearch, which makes our searching and enumerating hyperparameters significantly easier than manually modifying parameters and tracking the best performance.

Classifier	Hyperparameters
Logistic Regression	C: [0.1, 0.2, 0.4, 0.6, 0.8, 1]
	class_weight: ['balanced']
Random Forest	max_depth: [None, 20, 50]
	min_samples_leaf: [1, 2, 4]
	n_estimators: [50, 100, 200]
Decision Tree	max_depth: [None, 10, 20, 50, 100]
	criterion: ['gini', 'entropy', 'log_loss']
Gradient Boosting	loss: ['log_loss', 'deviance', 'exponential']
	n_estimators: [20, 50, 100]
	max_depth: [3, 5, 10]
KNN	n_neighbors: [1, 3, 5, 7, 10]

TABLE I: Classifier Hyperparameters

E. Model Evaluation

As we are dealing with classification problems, our model training evaluation is based on simple accuracy, calculated using the following formula:

$$accuracy = \frac{\#\ correct\ predictions}{Size\ of\ Test\ Set} \times 100\% \qquad (1)$$

In all experiments in this study, we use the sci-kit learn build-in *accuracy_score* method to calculate the accuracy of each model and hyperparameter combinations.

F. Performance Visualization

We visualize our five-model performance by different test sizes using a joint line chart for each dataset so that we can directly see the change in performance concerning test size and training set size. Also, for each dataset, we visualize the percentages of correct and incorrect predictions as a bar chart as well as a heap map using a confusion matrix for accuracy on the best performance model. Through this process, we can know the performance of our model undoubtedly. For correlation analysis, we use the column of target values in the heat map to visualize the correlations between different features and our targets.

III. Experiments

A. Dataset 1 - Bank Marketing

Within this dataset, the target labels exhibit a binary nature, denoted as 1 and 0. To establish a standardized representation wherein all target labels are characterized as positive and negative, a re-encoding process is undertaken, transforming 0 in the target to -1. Subsequent to this re-encoding operation, correlation analysis is executed on the One-Hot Encoded feature DataFrame. Specifically, features exhibiting correlations with the target greater than or equal to 0.06 are selected for further consideration. The resultant feature list, along with their corresponding correlation coefficients, is detailed below.9 We fit five chosen models. The models' performances are shown here 1.

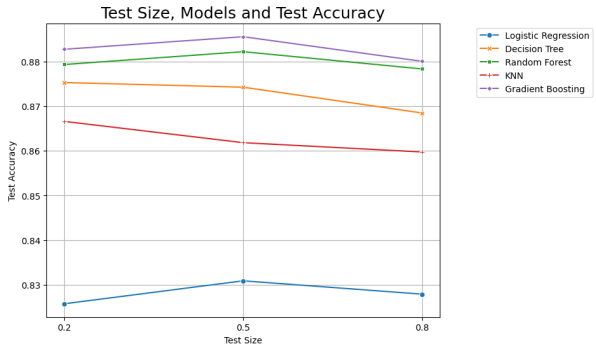

Fig. 1: Model Performances on Dataset 1

The best model for this dataset is the Gradient Boosting Classifier using log loss with a max depth of 3 and 100

estimators. Its accuracy can reach 88% on the 80/20 train-test split case, and the confusion matrix of this model is in the next figure 2.

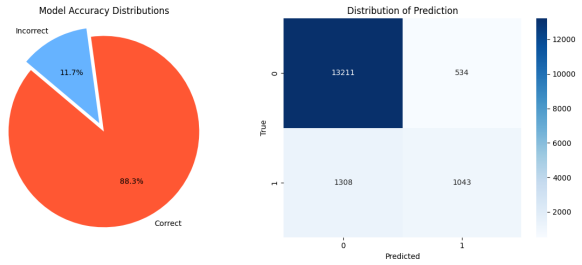

Fig. 2: Best Model Performance of Dataset 1

Based upon the correlation analysis of this dataset, we observed that a substantial proportion of features exhibit modest correlations with the target, falling below the threshold of 0.1. Consequently, even after parameter fine-tuning, the overall accuracy of the model remains below 90%. Furthermore, variations in the size of the test set yield negligible alterations in accuracy, and it's likely caused by the dataset's size(45,211 data points). A training data proportion of 0.2 is still sufficient for the training of a non-underfitting given the ample dataset size. Despite the incorporation of more than 10 features during training, the model does not manifest pronounced overfitting tendencies. Nevertheless, an observable trend towards increased overfitting is noted as the test set size expands II.

B. Dataset 2 - Student Performance

The target of this dataset is students' grades at Grade 3, which are continuous values. To convert our problem to binary classification, we encode students whose scores are greater than or equal to 10 to be 1 ('pass') and -1 ('fail') for all other cases. Then, when selecting one-hot encoded features, we keep all features with at least a 0.1 correlation since there are a relatively large amount of features with high correlations. The features selected and their corresponding correlations are in appendix A10. The model performances on the test set with different sizes are in the graph 3.

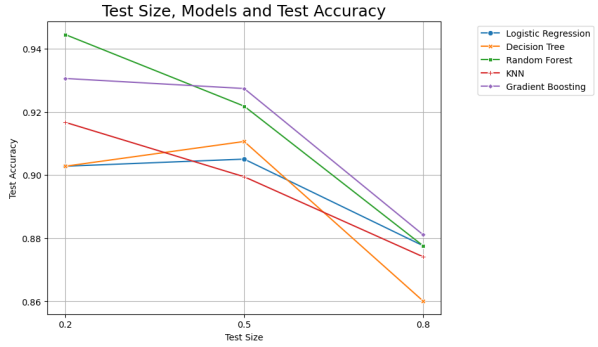

Fig. 3: Model Performances on Dataset 2

The best model is the Random Forest Classifier with a max depth of 20, 100 estimators, and min_samples_leaf of 2. This model produces an accuracy of 94.4% on an 80/20 train-test split case.

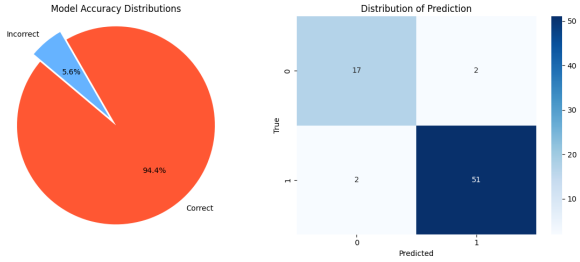

Fig. 4: Best Model Performance of Dataset 2

This dataset gives a relatively good best model accuracy, and the possible reason is that many features have correlations at least 0.1 even 0.15. However, one notable trend is that although the accuracy stays the same for test sizes of 0.2 and 0.5, when the test size increases to 0.8, the performance of all models decreases significantly. This could be caused by the size of the dataset (less than 400 data points in total). Thus, when the training data is too few, the models are all underfitted or overfitted to the training data. Similar to dataset 1, as the test size increases, overfitting is more and more serious as the difference between the accuracy on the training and test set are largerIII.

C. Dataset 3 - Room Occupancy

This dataset also has continuous targets, Room_Occupancy_Count, and possible values are 0, 1, 2, and 3. As more than 70% of data points, we combined the targets to an empty room (value 0) and a non-empty room (values 1, 2, 3) in order to alleviate the imbalance of the data points. Surprisingly, the features in this dataset are exceptionally high since more than 10 features display correlations with a target of more than 0.5, and we just set the correlation threshold to 0.5. The features selected and their correlations are in Appendix A 11. And the overall model performance on various test sizes is in the graph 5.

Fig. 5: Model Performances on Dataset 3

The best model on this dataset is the Random Forest Classifier with a max depth of 50, min_samples_leaf of 1, and 100 estimators. This model surprisingly obtains 100%

accuracy on the test set and nearly 100% on the validation set.

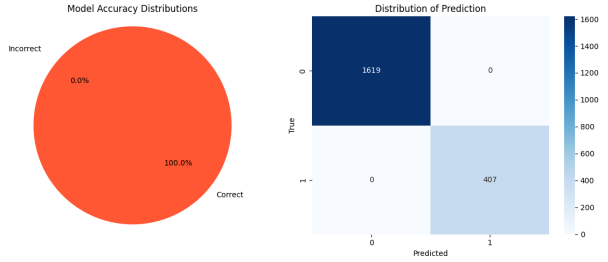

Fig. 6: Best Model Performance of Dataset 3

And also one remarkable phenomenon is that as the test size changes, the accuracy of all models did not change greatly (accuracies of all models in all test sizes are greater than 99%) and the overfitting is negligible in this dataset IV.

D. Dataset 4 - Letter Recognition

Targets in this dataset include 26 letters. Considering both the balanced quantity of positive and negative samples and the practical meaning of our binary splitting, we encode 5 vowel letters to be positive class and all other letters to be negative class. Since this dataset only has 16 features in total and all of them are numerical, we keep all features to adapt to its large data size (around 20000 data points), but we still perform the correlation analysis 12. The Model performance on the test set in various partition rates is in the figure 7.

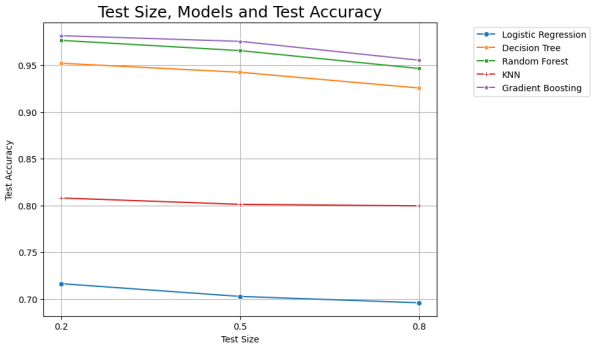

Fig. 7: Model Performances on Dataset 4

On this dataset, the difference in performances of models diverges obviously. All tree-based models (Decision Tree, Random Forest, and Gradient Boosting) outperform other models like linear regression and KNN to a large extent. The possible reason is that the relationship between features and targets is not linear or regular, so tree-based models, which can search out both linear and non-linear relationships, will be much better than logistic regression, which is a linear model, and KNN (sensitive to noisy and outliers and confirmed previous study that KNN does not work well in very high dimension and we have 16 dimensions in this model. [1])

The best model on this dataset is the Gradient Boosting Classifier using log loss with a max depth of 10 and 100 estimators. This model can achieve 98.18% accuracy on the 0.2 test size experiment.

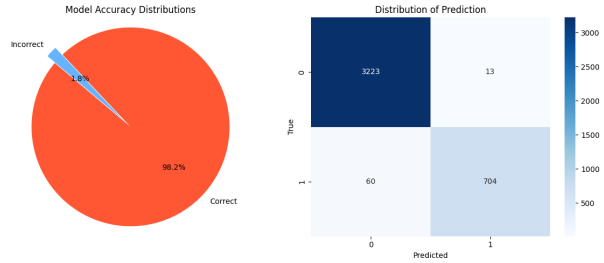

Fig. 8: Best Model Performance of Dataset 4

IV. Conclusion

A. Model Performance

Drawing insights from the fitting outcomes across four datasets characterized by diverse fields and feature-target properties, it becomes evident that the Gradient Boosting Classifier and Random Forest emerge as the optimal models. These models consistently demonstrate superior performance across the majority of datasets, with the sole exception being the Room Occupancy Dataset, where all models yield exceptionally high accuracy. This empirical observation aligns cohesively with the findings of a preceding empirical investigation into distinct machine learning methods, notably delineated in Caruana and Niculescu-Mizil's paper [1]. In their study, both BSTDT (boosted trees) and Random Forest models exhibit the highest mean standardized scores in performance.

The performance of KNN and logistic regression, based on the complete experimental performances in the Appendix, are similar, which in general further confirms the previous study [1], but the slight variance is that among four datasets, KNN performs obviously better on two datasets (dataset 1 and 4) while achieving parity with the other models on the remaining two datasets.

B. Influence of Test Size

Upon examination of model performance across varying test sizes, a discernible trend of overfitting becomes apparent with the enlargement of the test set, consequently diminishing the size of the training set. Across datasets 1, 2, and 4, a consistent pattern emerges, wherein most models exhibit an increase of difference between training set accuracy and test set accuracy as the test size increases. Notably, as the test size reaches 0.8, there is usually a large gap between training set performance and test set performance for the majority of models. A typical example of this phenomenon is evident in the case of the decision tree model applied to dataset 2 with a test size of 0.8, where the training set attains a perfect fit at 100%, while the accuracy on unseen data diminishes to 87%, which is a clear indication of overfitting.

Fortunately, datasets 1 and 4 do not manifest severe overfitting concerns, with the disparity between model accuracy on the training set and test set remaining below 5% across various test sizes and model types. This resilience against overfitting can be attributed to the substantial volume of data points inherent in both datasets. It is thereby inferred that even with the constraint imposed by a test size of 0.8, enough or nearly enough quantity of data points remain available for the models to discern the intricate relationships between features and targets.

C. Correlations and Model Performance

In our evaluation, we examined the relatively highly correlated features and their corresponding correlations for each dataset. As indicated by the model performance in the Appendix and heapmap of correlations, if a dataset has more features with high correlations with the targets, then it is easier to make predictions and classifications, which in turn results in a high model accuracy using the same list of machine learning methods. The typical example is that datasets 2 and 3 both have a few features whose correlations with targets are impressive (many are > 0.15 and > 0.5 respectively). However, this trend is still not clear in our current experiments. One counterexample is datasets 1 and 4. Given they both have large data points and a similar total number of features selected for the model training, although dataset 1 has more features with correlations > 0.1 than that is in dataset 4, and dataset 4 contains several features with extremely low correlations with targets (e.g. y-box has only -0.0024), three tree-based classifiers still performs much better on dataset 4 (tree-related models with best parameters on test size 0.2 gives $> 95\%$ accuracy).

V. Discussion

Through the execution of experiments involving four distinct datasets and diverse models, our overarching findings substantiate prior observations. Despite conscientiously incorporating correlation considerations in the feature engineering stage, the ultimate model performance, and the correlation of selected features, while exhibiting indications of positive associations, fail to demonstrate explicit or linear connections with accuracies across our experimental trials. Notably, the implementation of the GridSearch class for hyperparameter tuning was constrained by time and hardware limitations, restricting our exploration to a maximum of three hyperparameters per model.

To delve deeper into the significance of correlations with targets during feature engineering and to comprehensively assess the strengths and limitations of each model, additional experiments encompassing a broader array of models, datasets, and an exhaustive hyperparameter tuning strategy are imperative. Such endeavors will be instrumental in elucidating the nuanced interplay between feature correlations and model performance, thereby enhancing our understanding of the intricate dynamics inherent in the machine learning processes employed.

VI. Code Availability

The code of model training and graphing are available here

References

[1] R. Caruana and A. Niculescu-Mizil, "An empirical comparison of supervised learning algorithms," Proceedings of the 23rd international conference on Machine learning - ICML '06, 2006. doi:10.1145/1143844.1143865

[2] Moro, S., Rita, P., and Cortez, P. (2012). Bank Marketing. UCI Machine Learning Repository. https://doi.org/10.24432/C5K306.

[3] I. S. Cortez, Paulo. (2014). Student Performance. UCI Machine Learning Repository. https://doi.org/10.24432/C5TG7T.

[4] Singh, Adarsh Pal and Chaudhari, Sachin. (2023). Room Occupancy Estimation. UCI Machine Learning Repository. https://doi.org/10.24432/C5P605.

[5] Slate, David. (1991). Letter Recognition. UCI Machine Learning Repository. https://doi.org/10.24432/C5ZP40.

[6] V. Aliyev, "Gradient boosting classification explained through python," Medium, https://towardsdatascience.com/gradient-boosting-classification-explained-through-python-60cc980eeb3d (accessed Nov. 28, 2023).

Appendix A
Correlation Figures

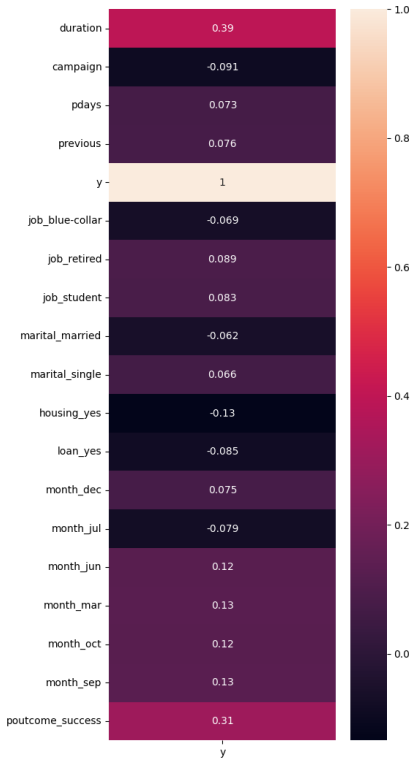

Fig. 9: Correlations for Selected Features in Dataset 1

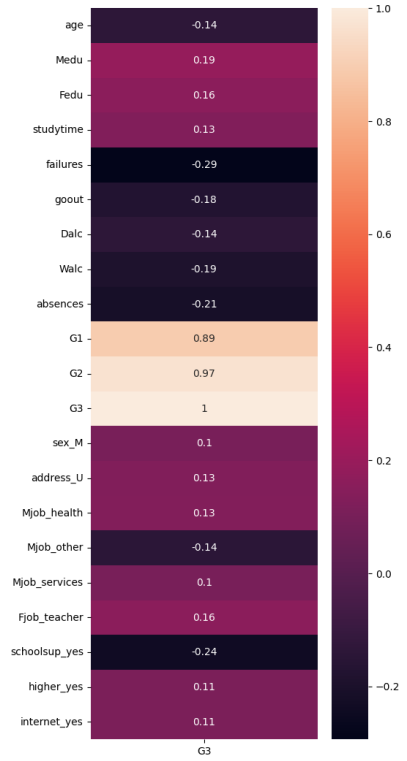

Fig. 10: Correlations for Selected Features in Dataset 2

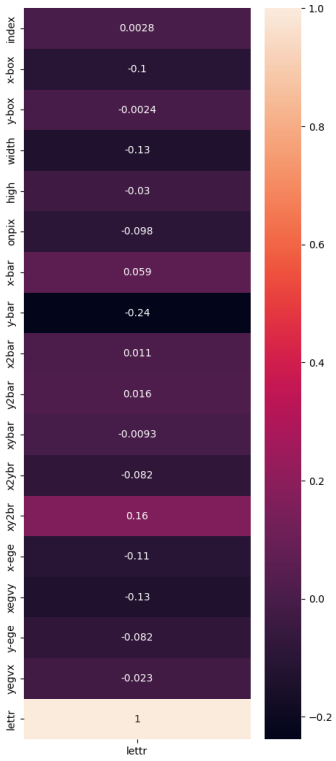

Fig. 12: Correlations for Selected Features in Dataset 4

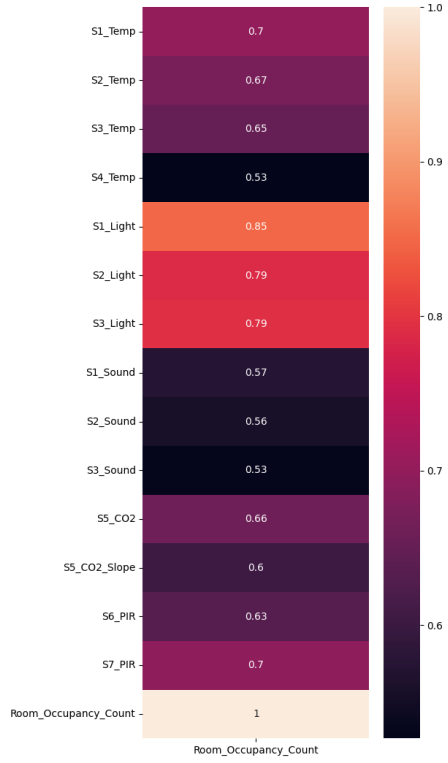

Fig. 11: Correlations for Selected Features in Dataset 3

APPENDIX B
COMPLETE TRAINING AND TEST SET PERFORMANCE

		Accuracy	
Model	**Train**	**Best Validation**	**Test**
Test Size: 0.2			
Logistic Regression	0.828169	0.828479	0.825749
Decision Tree	0.898299	0.873641	0.875291
Random Forest	0.912279	0.881912	0.879329
KNN	0.879815	0.859351	0.866594
Gradient Boosting	0.888514	0.882961	0.882746
Test Size: 0.5			
Logistic Regression	0.827027	0.824852	0.83089
Decision Tree	0.904318	0.870643	0.874254
Random Forest	0.909351	0.879093	0.882207
KNN	0.878472	0.861386	0.861829
Gradient Boosting	0.889593	0.880646	0.885562
Test Size: 0.8			
Logistic Regression	0.826188	0.823237	0.827904
Decision Tree	0.918453	0.861758	0.868481
Random Forest	0.908978	0.874961	0.878344
KNN	0.875272	0.857098	0.859744
Gradient Boosting	0.889562	0.877136	0.880053

TABLE II: Complete Model Performance on Dataset 1

Model	Accuracy		
	Train	Best Validation	Test
Test Size: 0.2			
Logistic Regression	0.915789	0.908772	0.902778
Decision Tree	1.0	0.891228	0.902778
Random Forest	0.982456	0.908772	0.944444
KNN	0.891228	0.884211	0.916667
Gradient Boosting	0.94386	0.898246	0.930556
Test Size: 0.5			
Logistic Regression	0.932584	0.882109	0.905028
Decision Tree	1.0	0.84275	0.910615
Random Forest	0.932584	0.882203	0.921788
KNN	0.898876	0.865348	0.899441
Gradient Boosting	0.949438	0.876554	0.927374
Test Size: 0.8			
Logistic Regression	0.915493	0.860507	0.877622
Decision Tree	1.0	0.875	0.86014
Random Forest	1.0	0.888889	0.877622
KNN	0.929577	0.90157	0.874126
Gradient Boosting	1.0	0.804348	0.881119

TABLE III: Complete Model Performance on Dataset 2

Model	Accuracy		
	Train	Best Validation	Test
Test Size: 0.2			
Logistic Regression	0.998025	0.997655	0.996051
Decision Tree	1.0	0.99963	0.998519
Random Forest	1.0	0.99963	1.0
KNN	0.999753	0.999753	1.0
Gradient Boosting	1.0	0.99963	0.998519
Test Size: 0.5			
Logistic Regression	0.998025	0.997038	0.997038
Decision Tree	1.0	0.99842	0.998815
Random Forest	1.0	0.999803	0.999605
KNN	1.0	1.0	0.999408
Gradient Boosting	1.0	0.99921	0.999013
Test Size: 0.8			
Logistic Regression	0.997531	0.996049	0.996051
Decision Tree	1.0	0.997531	0.997038
Random Forest	1.0	0.999506	0.999383
KNN	1.0	0.997531	0.999136
Gradient Boosting	1.0	0.998519	0.998272

TABLE IV: Complete Model Performance on Dataset 3

Model	Accuracy		
	Train	Best Validation	Test
Test Size: 0.2			
Logistic Regression	0.695937	0.696437	0.71625
Decision Tree	0.996437	0.948562	0.95225
Random Forest	1.0	0.967625	0.97675
KNN	0.80825	0.801625	0.808
Gradient Boosting	1.0	0.975375	0.98175
Test Size: 0.5			
Logistic Regression	0.6978	0.6988	0.7026
Decision Tree	1.0	0.9374	0.9426
Random Forest	1.0	0.9597	0.9658
KNN	0.8107	0.8045	0.8012
Gradient Boosting	1.0	0.9671	0.9757
Test Size: 0.8			
Logistic Regression	0.69075	0.690499	0.695812
Decision Tree	1.0	0.9115	0.92575
Random Forest	1.0	0.942751	0.946688
KNN	0.81275	0.80825	0.799625
Gradient Boosting	1.0	0.950001	0.9555

TABLE V: Complete Model Performance on Dataset 4