Machine Learning Final Project

Prediction of Energy Star Score of Washington DC's Buildings

Teammates: Kexin Xu, Zhenghao Zhang, Haiping Guo, Haoran Deng.

I. Introduction

The energy consumption in buildings has significantly increased in the last decade. In 2018, Researchers at MIT estimate commercial buildings account for 20% of all the energy used in the U.S. and concludes that as much as 30% of that energy is wasted. In commercial buildings, energy is an expense of business operation and the productivity of people is the real goal. Thus, we want to tracking energy usage within buildings and build a model to predict the Energy Star Score based on the national energy consumption survey data. Energy Star Score is a percentile measure of a building's energy performance, which is a simple but useful tool to find which building waste more energy and which features are highly correlated with trends in energy waste.

II. Problem Statement

In this problem, we present findings from an exploration of the Washington DC benchmarking dataset which measures more than 30 variables related to energy use for over 1800 buildings. We visualize the data and identify predictors within the dataset for Energy Star Score, also build regression and classification models that can predict the score in the given data. Then test the results of the model and get the error rate. We seek out what variables provide the most information about the score and which model works best. Finally, we analyze which classifier fails and why.

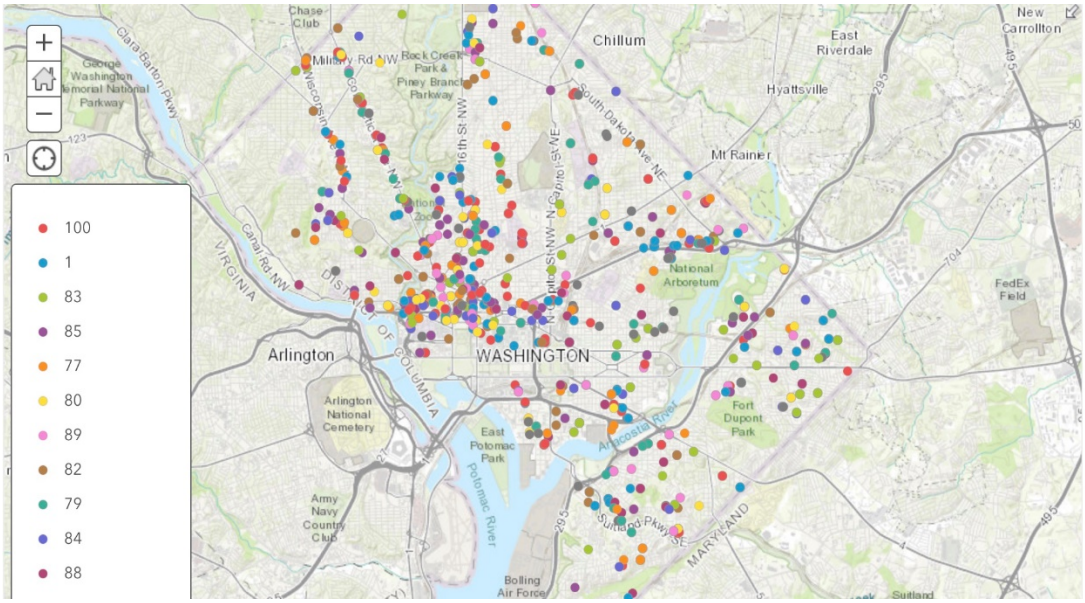
III. Related Work

Many researchers have proposed a methodology for energy consumption prediction or energy waste prediction in buildings. And they developed many solutions on machine learning algorithms like artificial neural networks(ANN), adaptive neuro-fuzzy inference system (ANFIS), support vector machine (SVM), extreme learning machine (ELM), some of which may be too complicated for this problem. Though, in many applications especially in building service engineering, the deep learning methods

worked efficiently as compared to the conventional machine learning approaches due to their slightly deeper architectures and novel learning methods. We will focus on the prediction accuracy of our created model. So we are going to start out with simple models such as linear regression and KNN, and move on to more accurate, complicate models such as Adaboost. And then find which works best and why some classifier fails.

IV. Dataset

We plan to use the energy and water performance benchmarking data from the largest buildings in the Washington DC. The dataset has 37 columns, among them, 23 columns are index values, many of which have missing values. And one feature is Energy Star score ranging from 1 to 100, which make this a supervised regression task in machine learning. The Energy star heat map of Washington DC is given as follows:

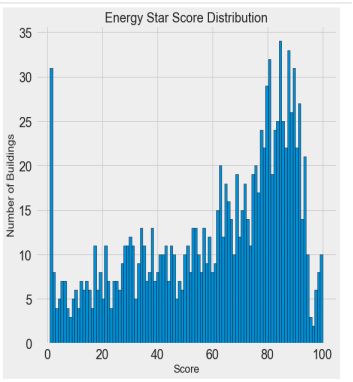


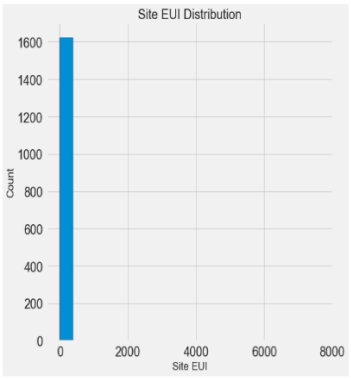
V. Approach

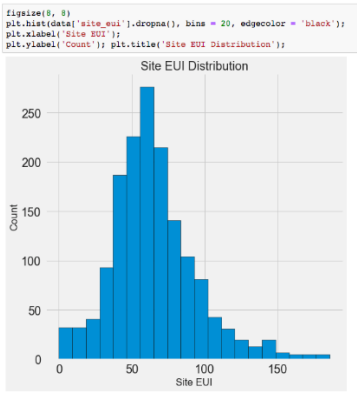
1. Distribution of Energy Star Scores

As we are concerned mainly with the energy star score, a reasonable place to start is examining the distribution of this variable. The first chart to make shows the distribution of this measure across all the buildings in the dataset. But Energy Star Score is a measure based on self-reported energy usage of a building. So, we find the Energy Use Intensity is based on actual energy use as determined by utility. It is straightforward to calculate: take the total annual energy use and divide by the square

footage of the building. The EUI is meant for normalized energy use comparisons between buildings and this measure is likely more objective because it uses actual measure consumption.



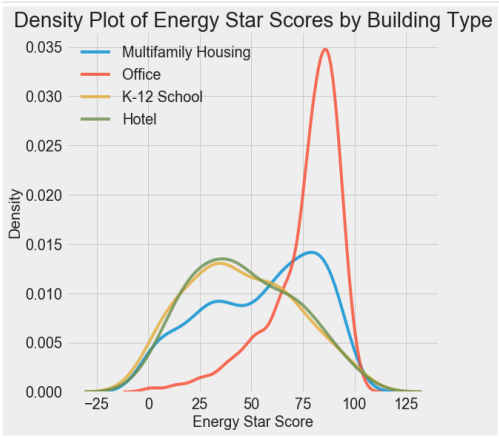


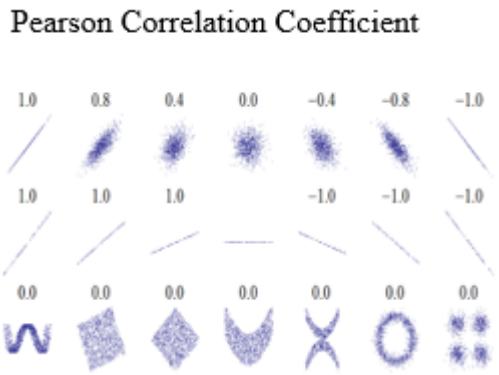


And we find a common problem when dealing with the EUI data is missing values. And we want to be careful about removing information, if a column has a high percentage of missing values, then it probably will not be useful to our model. So we Calculate the portions of missing values for each column and drop the columns which have more than 50% missing values.(df.isnull().sum()). Another problem is the outlier effect, we remove outlier when values which are less than: (first quartile(25%)-3*interquartile range(range of 75%-25%)) Or more than: (third quartile(75%)+3*interquartile) will be removed. While we dropped the columns with more than 50% missing values when we cleaned the data and there are a number of ways to fill in missing data, we will use a relatively simple method, median imputation. We replaces all the missing values in a column with the median value of the column. As we can see, a fter removing outlier, the distribution of site EUI is not suspicious.

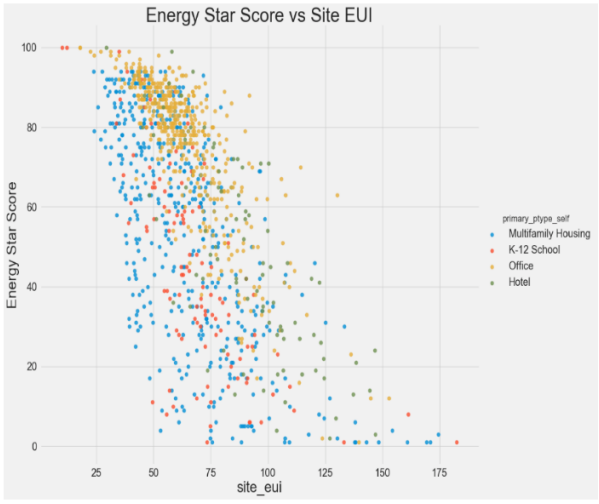
2. Looking for relationships

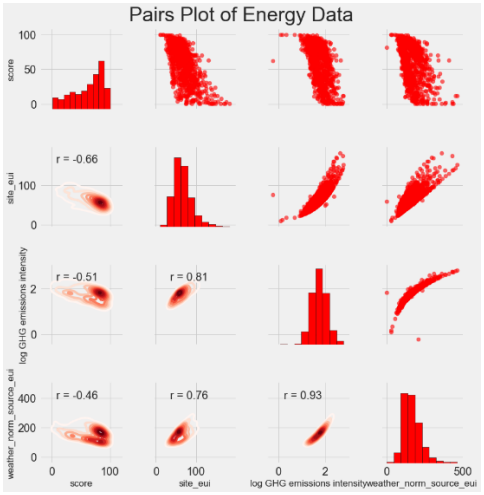
A major part of pre-processing is searching for relationships between the features and the target. Variables that are correlated with the target are useful to a model, so we use the density plot to examine the effect of a categorical variable. Another way to quantify relationship variables is the Pearson Correlation Coefficient. This is a measure of strength and direction of a linear relationship between 2 variables. A score of +1 is a perfectly linear positive relationship and a score of -1 is a negative linear relationship.





To visualize relationships between two continuous variables, we use scatterplots. We can include additional information, such as a categorical variable, in the color of the points. For example, the following plot shows the Energy Star Score vs. Site EUI colored by the building type; and we use the seaborn visualization library to see elationships between multiple pairs of variables:





3. Feature engineering and selection:

In this project, we take the following 2 feature processing steps:

- One-hot encode categorical variables (borough and property use type)
- Add in the natural log transformation of the numerical variables

Add transformed features can help our model learn non-linear relationships within the data. Taking the square root, natural log, or various powers of features is common practice in data science and can be based on domain knowledge or what works best in practice. Here we will include the natural log of all numerical features. As to feature selection, many of the 37 features we have in our data are redundant because they are

highly correlated with one another. In this project, we will use the correlation coefficient to identify and remove collinear features. We will drop one of a pair of features if the correlation coefficient between them is greater than 0.6.

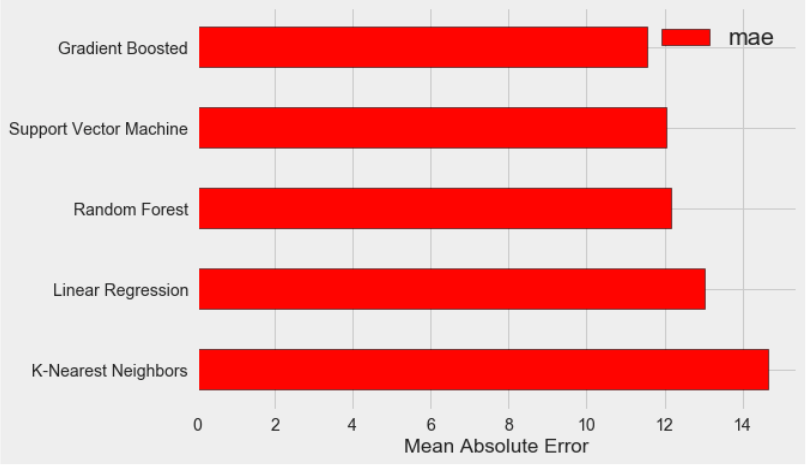
VI. Experiments

1. Implementing machine learning models in Scikit-Learn

After all the work, we spent cleaning and formatting the data, we will use the Scikit-Learn library in Python, which has great documentation and a consistent model building syntax. Here 5 machine learning models are tested (with default parameter) and using the metric: Mean(abs $\sum((Ypred - Ytrue))$ (mean absolute error) to evaluate the accuracy of the model.

```
plt.style.use('fivethirtyeight')
figsize(8, 6)

# Dataframe to hold the results
model_comparison = pd.DataFrame({'model': ['Linear Regression', 'Support Vector Machine',
                                           'Random Forest', 'Gradient Boosted',
                                           'K-Nearest Neighbors'],
                                 'mae': [lr_mae, svm_mae, random_forest_mae,
                                         gradient_boosted_mae, knn_mae]})

# Horizontal bar chart of test mae
model_comparison.sort_values('mae', ascending = False).plot(x = 'model', y = 'mae', kind = 'barh',
                                                            color = 'red', edgecolor = 'black')

# Plot formatting
plt.ylabel(''); plt.yticks(size = 14); plt.xlabel('Mean Absolute Error'); plt.xticks(size = 14)
```

```
(array([ 0.,  2.,  4.,  6.,  8., 10., 12., 14., 16.]),
 <a list of 9 Text xticklabel objects>)
```



From comparison above, gradient boost has a slightly higher accuracy than other models. These results aren't entirely fair because we are mostly using the default values for the hyperparameters. Especially in models such as the support vector machine, the performance is highly dependent on these settings. Nonetheless, from these results we will select the gradient boosted regressor for model optimization.

2. Model Optimization

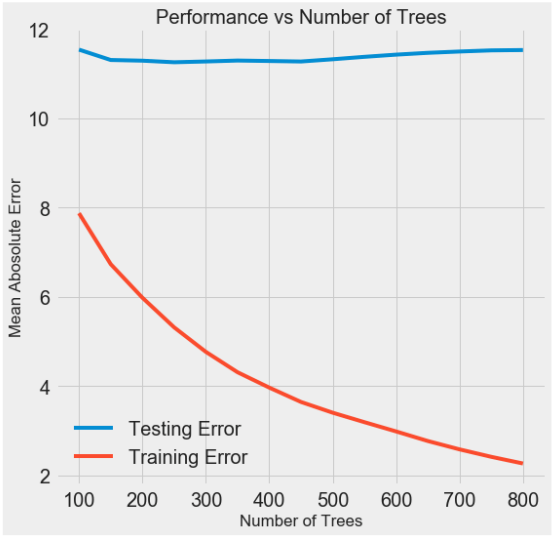
The Gradient Boosted Regression model is an ensemble method, meaning that it is built out of many weak learners, in this case individual decision trees. While a bagging algorithm such as random forest trains the weak learners in parallel and has them vote to make a prediction, a boosting method like Gradient Boosting, trains the learners in sequence, with each learner "concentrating" on the mistakes made by the previous ones. Controlling the hyperparameters affects the model performance by altering the balance between underfitting and overfitting in a model. And we use random search for permuting and combining parameters of gradient boost model from Sklearn package and K-fold cross validation for training and test dataset to select the best combination of parameters for this model. A

```
In [201]: random_cv.best_estimator_

Out[201]: GradientBoostingRegressor(alpha=0.9, criterion='friedman_mse', init=None,
                                    learning_rate=0.1, loss='huber', max_depth=3,
                                    max_features='auto', max_leaf_nodes=None,
                                    min_impurity_decrease=0.0, min_impurity_split=None,
                                    min_samples_leaf=2, min_samples_split=4,
                                    min_weight_fraction_leaf=0.0, n_estimators=500,
                                    presort='auto', random_state=42, subsample=1.0, verbose=0,
                                    warm_start=False)
```

loss = huber
n_estimators = 500
max_depth = 3
min_samples_leaf = 2
min_ samples_split = 4
max_features = 'auto'

One experiment we can try is to change the number of estimators (decision trees) while holding the rest of the hyperparameters steady. This directly lets us observe the effect of this particular setting. When we increase the loops of the mode, both the training and the testing error decrease. However, the training error decreases much more rapidly than the testing error and we can see that our model is overfitting: it performs very well on the training data, but is not able to achieve that same performance on the testing set.

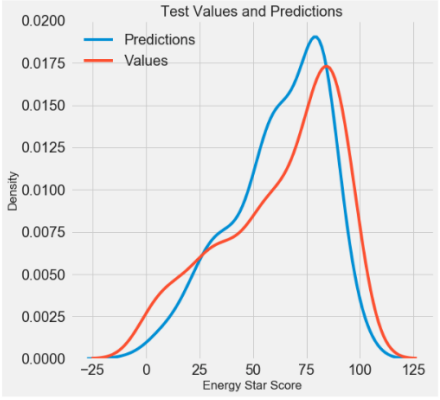
```
# Get the results into a dataframe
results = pd.DataFrame(grid_search.cv_results_)

# Plot the training and testing error vs number of trees
figsize(8, 8)
plt.style.use('fivethirtyeight')
plt.plot(results['param_n_estimators'], -1 * results['mean_test_score'], label = 'Testing Error')
plt.plot(results['param_n_estimators'], -1 * results['mean_train_score'], label = 'Training Error')
plt.xlabel('Number of Trees'); plt.ylabel('Mean Abosolute Error'); plt.legend();
plt.title('Performance vs Number of Trees');
```

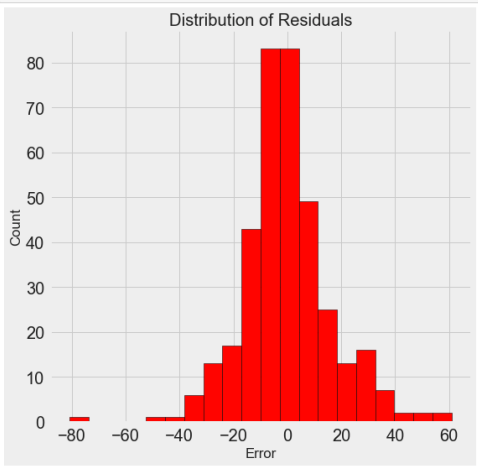


We always expect at least some decrease in performance on the testing set (after all, the model can see the true answers for the training set), but a significant gap indicates overfitting. We can address overfitting by getting more training data, or decreasing the complexity of our model through the hyerparameters.

Once we have the final predictions, we can investigate them to see if they exhibit any noticeable skew. On the left is a density plot of the predicted and actual values, and on the right is a histogram of the residuals:

```
figsize(8, 8)

# Density plot of the final predictions and the test values
sns.kdeplot(final_pred, label = 'Predictions')
sns.kdeplot(y_test, label = 'Values')

# Label the plot
plt.xlabel('Energy Star Score'); plt.ylabel('Density');
plt.title('Test Values and Predictions');
```





VII. Discussion

In this project. We have following findings. First, Energy star scores in offices, dormitories, and non-refrigerated warehouses are higher than in senior care communities and hotels. Second, the Site Energy Use Intensity, the Electricity Intensity, and the natural gas usage are all have bad effect on the Energy Star Score. Last, Gradient boost trained on the training data was able to have an average absolute error of 10 points on a hold-out testing set, which was significantly better than baseline measure. If have data for a new building, our trained model could accurately infer the Energy Star Score.

VIII. Conclusion

Back to the question: Is there a model could predict the Energy Star Score of a building and what feature has the most significant effect on the score?

The conclusion is yes. We found that the EUI, the floor area and Building Type are important features for energy star score. Also, we created a Gradient Boosted model to predict the Energy Star Score of a building, and apply random search to improve model and could get accurately prediction.

References

[1] Department of Energy & Environment. Energy Benchmarking Disclosure[Online].Available:https://doee.dc.gov/page/energy-benchmarking-disclosure

[2] William Chung, Y.V. Hui, Y. Miu Lam, *Applied Energy,* Elsevier Ltd, 2016, vol. 83, issue 1, pp. 1-14
Available:https://www.sciencedirect.com/science/article/pii/S0306261904002028

[3] Constantine E. Kontokosta, "Predicting Building Energy Efficiency Using New York City Benchmarking Data," *ACEEE Summer Study on Energy Efficiency in Buildings*, vol.4, pp.163-174, 2012
Available:https://aceee.org/files/proceedings/2012/data/papers/0193-000114.pdf

[4] Seyedzadeh, S., Rahimian, F., Glesk, I. et al. "Machine learning for estimation of building energy consumption and performance: a review", *Visulization in Engineering,* vol.6, no.5, 2018
Available:https://link.springer.com/article/10.1186/s40327-018-0064-7

[5] Giovanni Tardioli, Ruth Kerrigan, Mike Oates, James O'Donnell, Donal Finn, "Data Driven Approaches for Prediction of Building Energy Consumption at Urban Level," *Energy Procedia*, vol. 78, 2015
Available:https://www.sciencedirect.com/science/article/pii/S1876610215024868

[6] Li Qiong, Ren Peng, Meng Qinglin. Prediction model of annual energy consumption of residential buildings. In: Proceedings of the IEEE international conference on advances in energy engineering; 2010. p. 223–6.
Available:https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5557576

[7] NYC Energy and Water Benchmarking. New York City Energy & Water Performance Map Available: https://serv.cusp.nyu.edu/projects/evt/

[8] Muhammad Fayaz, Dohyeun Kim. A Prediction Methodology of Energy Consumption Based on Deep Extreme Learning Machine and Comparative Analysis in Residential Buildings, Electronics, vol. 7, issue 10, 2018

Available: https://www.mdpi.com/2079-9292/7/10/222/htm