Predicting Adoption of Solar Power Across the United States

Gabrielle Nyirjesy

Introduction

There is no denying the impact that climate change has had on the planet. According to the Intergovernmental Panel on Climate Change (IPCC) in their Sixth Assessment Report, climate change is a result of human actions, including increases in CO_2 and greenhouse gas emissions, and it is already affecting weather and extreme events worldwide.[1] At this point, the world needs to limit emissions to prevent warming the planet to 2°C; a threshold where extreme events would become even more widespread. In April of 2021, President Biden committed to reduce the United States greenhouse gas emissions by 50-52% by 2030 as part of the country's contribution to the Paris Climate Agreement.[2] The US emits the second most greenhouse gases in the world, so hitting these targets and even going beyond them would help to curb warming effect on the climate. Electricity generation in the US is mainly fueled by coal and natural gas, which is why this area accounts for 25% of the country's greenhouse gas emissions. By aggressively switching to renewable energy sources of electricity, the country can begin to achieve the climate goals and prevent further warming of the planet. Solar is a likely candidate as a renewable energy source because the cost of solar has dropped more than 70% over the last decade and is now competitively priced with other dirtier fuel sources.[3]

This report aims to identify drivers of solar adoption in both the residential and the industrial sectors through a machine learning predictive model. For each sector, the model attempts to predict solar panel system count at the county level using a combination of socioeconomic, policy, and financial data related to the electricity market.

Data

Description

The DeepSolar group at Stanford University created a comprehensive dataset that identified the size, type, and number of solar panels in each county across the United States.[4] The solar panel data was determined through a convolutional neural network (CNN) that used satellite imagery to classify panels and estimate their sizes with precision and recall performance around 90%.[5] The dataset includes information for all 48 contiguous U.S. states. There are 72,537 rows and 168 columns. These correspond with information for 72,537 twelve digit Federal Information Processing System (FIPS) codes for census tract locations across the U.S. The columns included in the dataset fall within five categories:

1. DeepSolar data
    a. Data related to the CNN DeepSolar model, including variables such as total solar area, tile count, etc. for residential and non-residential panels
    b. Source: DeepSolar model
2. U.S. Census data
    a. Data related to the U.S. Census, including variables such as education levels, diversity, income, etc.
    b. Source: ACS 2015

3. Electricity data
    a. Data related to the electricity such as consumption and prices
    b. Source: U.S. Energy Information Administration
4. Weather data
    a. Data related to weather, such as solar radiation, temperature, elevation, etc.
    b. Source: NASA Surface Meteorology and Solar Energy
5. Political data
    a. Data related to political leaning of the area, including variables such as democratic and republican voting percentages in 2012 and 2016
    b. Source: Townhall.com and theguardian.com
6. Policy data
    a. Data related to incentives, taxes, and tariffs around solar energy
    b. Source: NC State Clean Energy Technology Center

The goal of this project is to design two models, one for residential and one for non-residential, to predict the count of solar systems within each county. Two models were developed because it seems likely that residential solar panel adoption and non-residential solar adoption have different drivers.

Data Pre-processing

Categorical Data

First, an exploratory data analysis was conducted to understand the features included in the data. There were only five categorical columns in the data: county, state, transportation electricity price, 2016 voting percentage won by the democratic party, and 2012 voting percentage won by democratic party. Through closer examination, it was found that the transportation electricity price was actually a numeric variable, so it was converted to a float. The two variables related to democratic party win percentage in 2012 and 2016 included only "True" and "False" values. Because categorical variables need to be imputed before being inputted into the machine learning model, these columns were converted to encode "True" as 1 and "False" as 0. The state and county columns were left as categorical because they would not be included in the final model; however, they were relevant for other data pre-processing steps including imputation of missing values.

Train-Test Split

After changing the columns to their appropriate data type, the data was split into a training and testing set. The train_test_split function was used and 20% of the data was used for the testing set and 80% of the data for the training set. This function automatically shuffles the data to add randomization to the sampling. The 80-20 split ensured that the model could be trained with the majority of the data and the performance of the model could be evaluated using unseen data. The aim of this separation was to reduce bias and over-reporting performance of the model metrics.

Missing Values Imputation

Next, the data was examined to identify columns with missing values to determine how best to impute them. Imputation is important for machine learning models because certain machine learning models, such as neural networks, need a complete dataset. Simply removing the columns or rows with missing data would reduce the training data size and potentially impact model performance and introduce bias in to the model.[6]

There were 103 columns that contained missing values. There are several methods for imputation of missing values. Some examples include zero, mean, and median imputation. Zero imputation

would add bias and skew some of the columns like median household income, total area, and education rate. Additionally, many of the columns with missing values had large or small outliers that would effect mean imputation. Therefore, median imputation was implemented. The county level median value, established using the training dataset only, was used to impute missing values for each column. In this way the imputed rows would be within the range of other land areas in the same county. Only the training dataset was used to establish the imputation value so there was no data leakage into the testing dataset.

There were still some variables that had missing values even after this imputation was completed. Most of the remaining variables were missing less than 10 rows; however, transportation electricity price 2012 democratic voting percentage, and 2012 GOP voting percentage were missing almost 5,00 rows. These columns were dropped from the dataset and then any remaining rows that had missing values in other columns were dropped. Lastly, columns that were likely to be uninformative in the model were dropped, such as FIPS, state, and county. These features provide granular information about the rows of data, but do not help to assess the larger trend in solar panel adoption.

Addressing Correlated Features

The next step in the data cleaning process was to remove correlated variables. Removal of correlated features is important because some machine learning models, such as Random Forest, become unstable in the presence of highly correlated features. Moreover, the feature importances of correlated features would be decreased as the importance would be shared across several features rather than attributed to the actual driver of the information. This could lead to incorrect interpretation of the model.[7] In the DeepSolar dataset there were 73 variables that had a correlation coefficient greater than 0.8 with another variable. A function was created to systematically remove variables from the data that were least correlated with the target variable. This was done in a stepwise fashion until 47 columns were removed and the correlation among all remaining variables was less than 0.8.

Normalizing the Data

The distribution of the remaining variables in the dataset were examined to determine the best way to normalize them. Normalization of the data is important so that all features are on a similar scale and follow a Gaussian distribution. The MinMaxScaler, StandardScaler, and PowerTransformer functions were compared. The MinMaxScaler function scales the data to its range by subtracting the minimum from each value and dividing by the maximum minus the minimum for the column. The StandardScaler function scales the data by standardizing it by subtracting the mean and dividing by the standard deviation. After standardization, the data will have a mean of 0 and a standard deviation of 1. The PowerTransformer is a form of logarithmic transformation that can also handle cases where the data is zero.[8] From the histograms of the variables, it was seen that the power transformation on the variables did the best job of removing skewness and standardizing the scale for the variables (Figure 1).

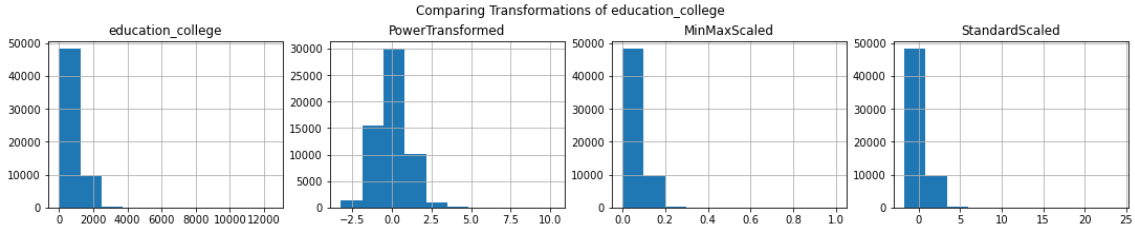

Figure 1. *Comparison of transformation methods on College Education variable.*

Through inspection of several of the variables, it was seen that the Box-Cox method performed better than the Yeo-Johnson method to power transform the variables to have a more Gaussian distribution (Figure 2).

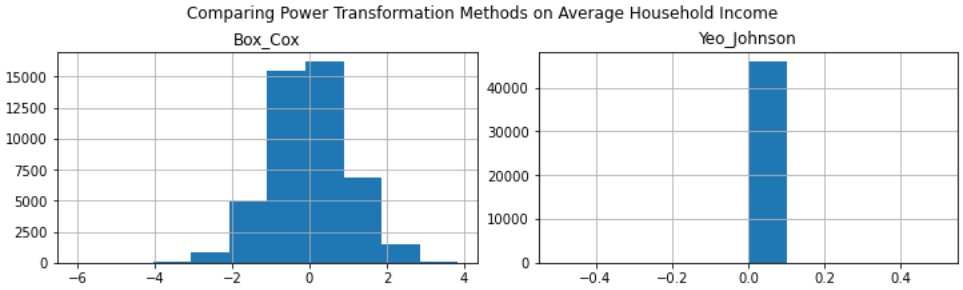

Figure 2. *Box-Cox and Yeo-Johnson transformation of average household income.*

The Box-Cox method uses maximum likelihood estimation to determine the ideal power transformation on the variable and only works on positive values. The Yeo-Johnson method also uses maximum likelihood estimation to determine the ideal power transformation, but it can handle zero and negative values (Figure 3).[9]

$$\text{Box-Cox Transformation: } \Psi(y,\lambda) = \begin{cases} \frac{y^{\lambda}-1}{\lambda}, \ \lambda \neq 0 \\ \log y, \ \lambda = 0 \end{cases}$$

$$\text{Yeo-Johnson Transformation: } \Psi(y,\lambda) = \begin{cases} \frac{(y+1)^{\lambda}-1}{\lambda}, \ y \geq 0 \text{ and } \lambda \neq 0 \\ \log(y+1), \ y \geq 0 \text{ and } \lambda = 0 \\ -\frac{(-y+1)^{2-\lambda}-1}{2-\lambda}, \ y < 0 \text{ and } \lambda \neq 2 \\ -\log(-y+1), \ y < 0 \text{ and } \lambda = 2 \end{cases}$$

Figure 3. *Box-Cox and Yeo-Johnson Transformation Formulas. λ is the power transformer that is determined through maximum likelihood estimation.*[9]

Taking this into account, all variables that were strictly positive were transformed using the Box-Cox method. All other variables were transformed using the Yeo-Johnson method. The power transformer was fit to the training data alone so as to avoid data leakage in the testing data. After the fit, both the training and testing datasets were transformed with the power transformer. As a result, all features were on similar scales and had more Gaussian distributions.

Modeling

Overview

The data was used to develop machine learning models for two objectives:

1. Predicting residential solar system count
2. Predicting non-residential solar system count

The two type of solar systems were split into separate models because the drivers of residential solar system adoption seemed likely to be different than the drivers of non-residential solar system adoption. A simple linear regression of these two target variables (Figure 4) showed that they have a lower correlation coefficient of 0.36. The count of residential and non-residential systems are also on very different scales: 0 - 1,400 for residential and 0 - 400 for non-residential. Therefore, it makes sense to split these type of predictions into two different models.

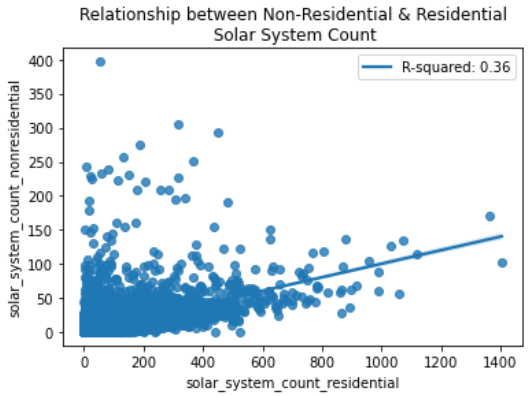

Figure 4. *The simple linear regression for non-residential and residential solar system count shows low correlation between these two target variables.*

Three different modeling methods, Random Forest, XGBoost and a Neural Network, were used to predict each target variable. The goal was to identify the best model for each target variable and hyperparameter tune that model to achieve the optimal performance for predictive accuracy.

Random Forest

The first model attempted was Random Forest. Radom Forest is an ensemble method that uses a combination of decision trees to create a more robust model.[10] Decision trees are weak learners that are subject to large variance and bias.[11] Random Forest is a bagging algorithm. This means that the algorithm fits many decision trees to sampled versions of the training data and creates its final regression by taking the majority vote or average from the trees. The objective function used to train the model was mean-squared error (MSE). The formula for MSE is shown in Figure 5 and will be used in all future models as the objective function:

$$MSE = \sum (y_i - \hat{y}_i)^2 / n$$

Figure 5. *Mean-Squared Error. y_i is the actual value, $\hat{y}_i$ is the predicted value, and n is the number of observations.*

Residential Random Forest Model

To establish a baseline for the Random Forest model, the data was passed through with the default settings. The MSE was 552 and the R^2 value was 0.748. The prediction errors were centered about zero, indicating a good fit (Figure 6). The Random Forest model took ~1.5 hours to complete.

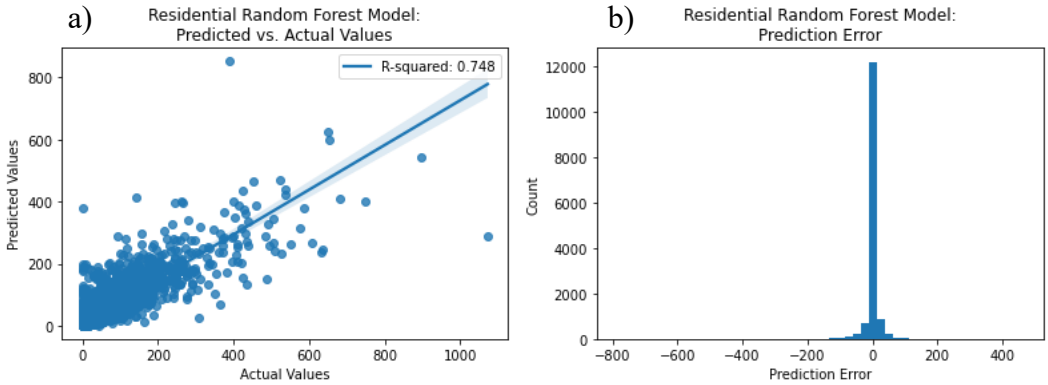

Figure 6. *Assessing the predictions for the residential solar system count Random Forest model with default settings shows a high correlation between predicted and actual values (a) and most prediction errors centered around zero (b).*

Hyperparameter tuning was completed to attempt to improve the model performance. Various settings for number of estimators (n_estimators), maximum features to consider when splitting trees (max_features), maximum tree depth (max_depth), minimum number of samples on which to split (min_samples_split), minimum samples for each leaf (min_samples_leaf), and bootstrapping (boostrap) were compared through a randomized search with 5 cross-validations. The optimal settings were found to be: {'n_estimators': 1000, 'min_samples_split': 5, 'min_samples_leaf': 5, 'max_features': 'sqrt', 'max_depth': 20, 'criterion': 'squared_error', 'bootstrap': False}. With these optimal settings, the MSE was still around 552 and the R^2 value was 0.753.

Residential Random Forest Model Performance			
Model Tuning	**Model Architecture**	**MSE**	**R^2**
None	• default	552	0.748
Random Search for Hyperparameters	• n_estimators: 1000 • min_samples_split: 5 • min_samples_leaf: 5 • max_features: 'sqrt' • max_depth: 20 • bootstrap: False	552	0.753

Figure 7. *Performance of Residential Random Forest Models*

The feature importance for the Random Forest model is shown in Figure 8. The feature importance is an attribute of the model and measures the mean and standard deviation of accumulation of the impurity decrease within each tree.[8]

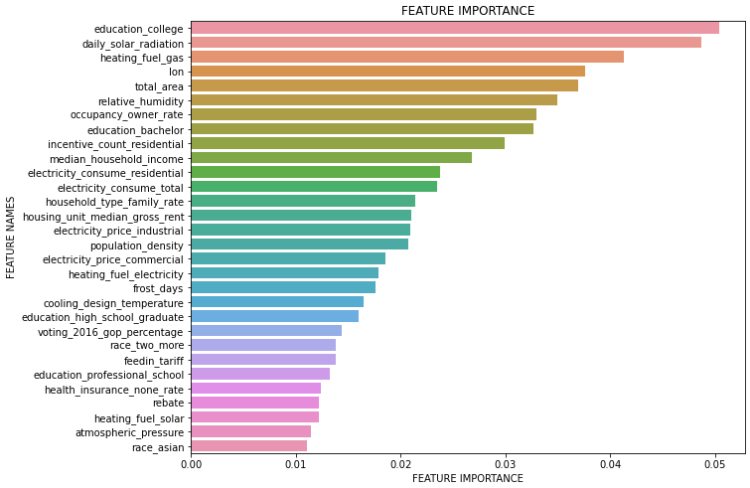

Figure 8. *Feature importance plot from hyperparameter tuned residential Random Forest model.*

From this plot, it seems that college education, daily solar radiation, and heating fuel gas are the most informative features in the model. These features account for almost 15% of the information gain.

Non-Residential Random Forest Model

The non-residential Random Forest model was trained in the same way as above. Results are shown in Figure 9. The initial baseline model using default settings had a MSE of 49.6 and an R^2

value of 0.249. This R^2 value shows low correlation between the predicted values and actual values of solar systems. Only 47% of the FIPS locations have at least one non-residential solar system. This low number of blocks with a solar system may have caused the low R^2 value.

Non-Residential Random Forest Model Performance			
Model Tuning	**Model Architecture**	**MSE**	**R^2**
None	• default	49.6	0.249
Random Search for Hyperparameters	• n_estimators: 1000 • min_samples_split: 5 • min_samples_leaf: 5 • max_features: 'sqrt' • max_depth: 20 • bootstrap: False	50	0.3

Figure 9. *Performance of Non-Residential Random Forest Models*

After hyperparameter tuning, the model MSE remained around 50 and the R^2 was 0.3 (Figures 9 and 10).

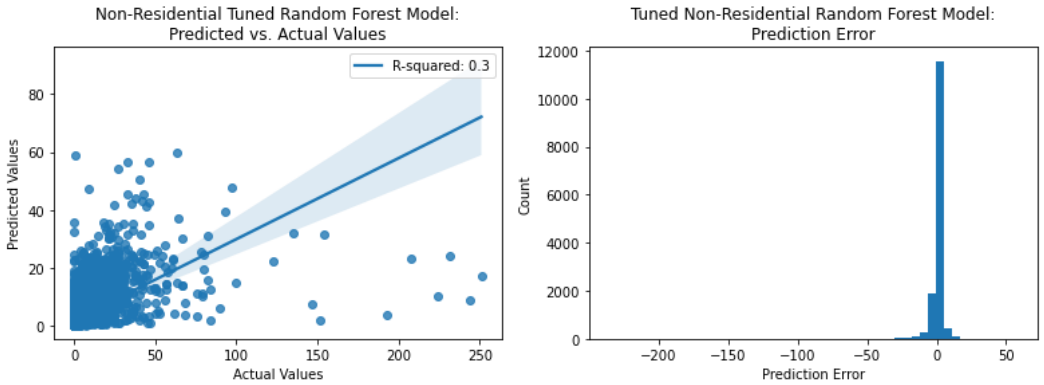

Figure 10. Tuned Non-Residential Random Forest Model shows a low R^2 value between predictions and actual values (a) and error concentrated around 0 (b).

An examination of the feature importances for this model showed the most important features were total area, population density, and relative humidity (Figure 11).

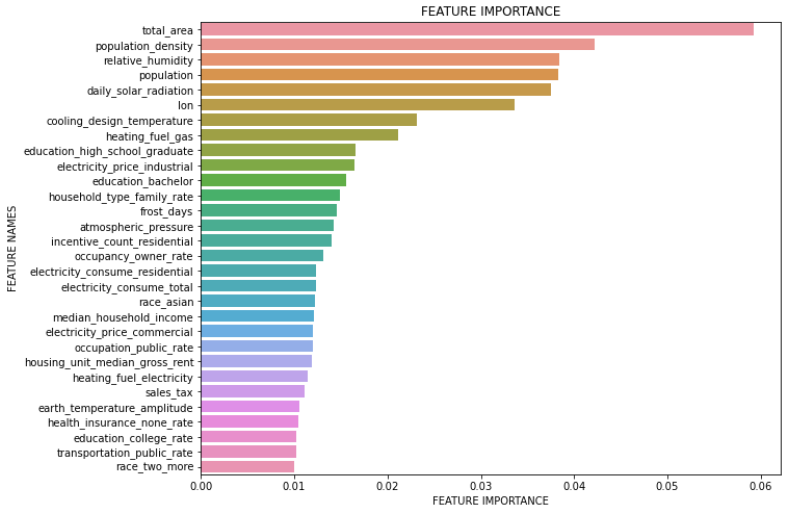

Figure 11. *Feature importance plot for tuned Non-residential Random Forest model.*

XGBoost

The next model attempted was XGBoost. XGBoost is an ensemble method that combines multiple decision trees with gradient boosting. XGBoost sequentially fits decision trees based on the previous tree's residual errors to essentially average over the decision tree results and minimize the loss function and error of the model.[12]

Residential XGBoost Model

The model for the residential solar system was initially trained using 1,000 estimators with a learning rate of 0.01 and all other default settings. The plot of the predicted and actual values on the test dataset show a high correlation between the two, with a R^2 value of 0.768 and a MSE of 503 (Figures 12 and 14).

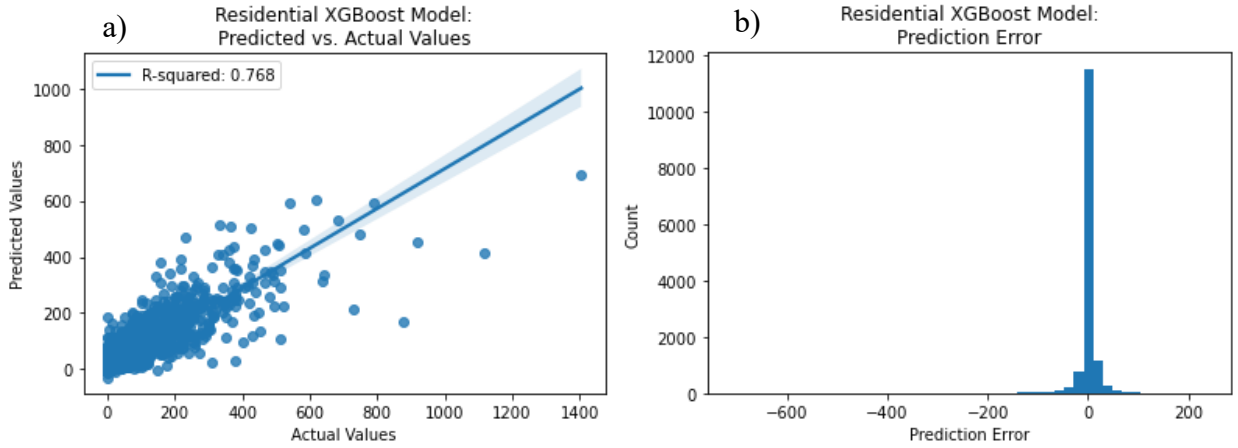

Figure 12. *Assessing the predictions for the residential solar system count XGBoost model with default settings shows a high correlation between predicted and actual values (a) and most prediction errors centered around zero (b).*

To see if it was possible to improve model performance, hyperparameter tuning was implemented using a randomized search over various parameter settings. It was found that the ideal settings were: {'reg_lambda': 2, 'reg_alpha': 0.2, 'objective': 'reg:squarederror', 'n_estimators': 300, 'min_child_weight': 10, 'max_depth': 20 , 'learning_rate': 0.01, 'gamma': 0 , 'colsample_bytree': 0.8, 'booster': 'gbtree'}.

The settings included two new regularization terms: lambda and alpha. These regularization terms help to adjust the model for overfitting and try to remove model complexity by adding a term to tamper the weights of the input features (Figure 13).

Regularization	Loss Function (MSE)
None	$LossFunction = \frac{1}{N}\sum_{i=1}^{N}(\hat{Y} - Y)^2$
L1 regularization (alpha/lasso)	$LossFunction = \frac{1}{N}\sum_{i=1}^{N}(\hat{Y} - Y)^2 + \lambda\sum_{i=1}^{N} \mid \theta_i \mid$
L2 regularization (lambda/ridge)	$LossFunction = \frac{1}{N}\sum_{i=1}^{N}(\hat{Y} - Y)^2 + \lambda\sum_{i=1}^{N} \theta_i^2$

Figure 13. *Comparison of loss function using different regularization terms, $\hat{y}$ is the predicted value, y is the actual value, N is the number of observations, λ is the weight for the parameter, θ is the parameter.*[13]

Lambda regularization is also called L2 regularization or ridge regression. L2 regularization penalizes the sum of square weights and forces them to be small, but not zero. Alpha

regularization corresponds to L1 regularization or lasso regression. L1 regularization shrinks parameters towards zero by minimizing their weights.[13]

The tuned model had an MSE of 518 and an R^2 value of 0.771, which was not an improvement from the model with default settings (Figure 14).

Residential XGBoost Model Performance			
Model Tuning	**Model Architecture**	**MSE**	**R^2**
None	• default	503	0.768
Random Search for Hyperparameters	• reg_lambda: 2 • reg_alpha: 0.2 • n_estimators: 30 • min_child_weight: 10 • max_depth: 20 • learning_rate: 0.01 • gamma: 0 • col_sample_bytree: 0.8 • booster: gbtree	518	0.771

Figure 14. *Performance of Residential XGBoost Models.*

The feature importances for the XGBoost model are shown in Figure 15. From this plot, it appears that the number of residential incentives was by far the most important feature in the XGBoost model, accounting for almost 40% of the information gain. Daily solar radiation, industrial electricity price, sales tax, and residential electricity consumption were the next most important features, all accounting for ~10% of the information gain in the model.

The features were also evaluated using Shapley values, which is a method based on game theory to determine how much contribution of information can be attributed to each feature.[14] The Shapley analysis also attributes high importance to daily solar radiation and residential incentive count, which both are positively correlated with the target, as shown by the high feature values corresponding with high positive impact on the model output.

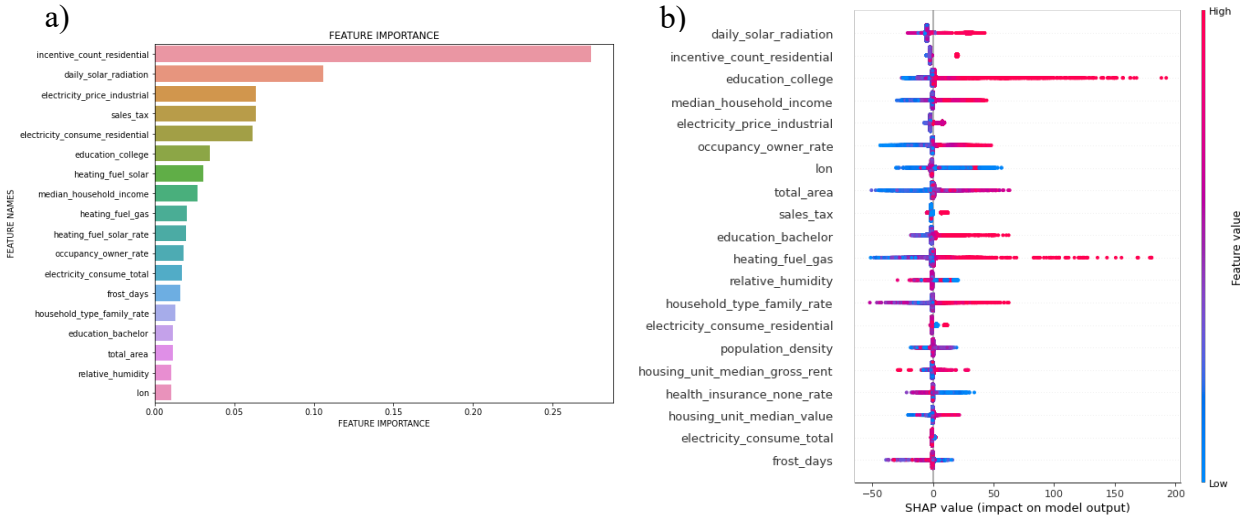

Figure 15. Feature importance from Residential XGBoost model using XGBoost feature importance (a) and SHAP value (b).

Non-Residential XGBoost Model

XGBoost was also used to predict the amount of non-residential solar systems. Again, a model was built with the default settings and compared with a model with hyperparameter tuning. The MSE and R^2 values for both models were 47 and ~0.28, respectively (Figures 16 and 17)

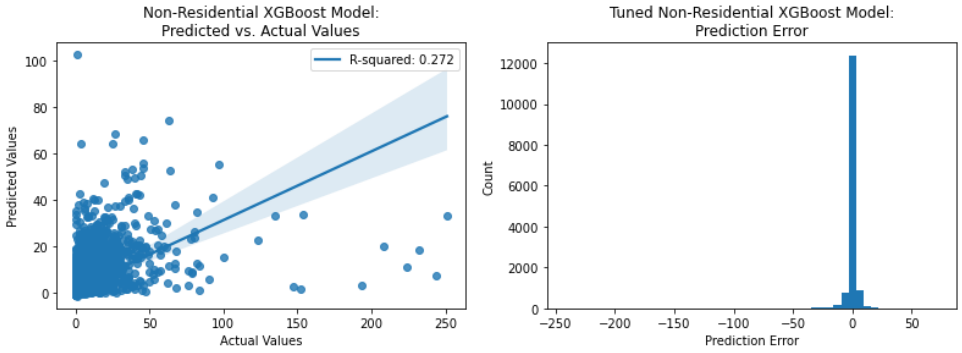

Figure 16. *Plots of Non-Residential XGBoost Model Performance*

The hyperparameter chosen in the random search contained lambda and alpha regularization terms, indicating that the default model may have had overfitting.

Non-Residential XGBoost Model Performance			
Model Tuning	**Model Architecture**	**MSE**	**R^2**
None	• n_estimators: 1000 • learning_rate: 0.01	47	0.272
Random Search for Hyperparameters	• reg_lambda: 3 • reg_alpha: 1 • n_estimators: 300 • min_child_weight: 10 • max_depth: 20 • learning_rate: 0.01 • gamma: 0 • col_sample_bytree: 0.8 • booster: gbtree	47	0.28

Figure 17. *Performance of Non-Residential XGBoost Models*

The feature importance plot for this model attributed 12% of information gain to daily solar radiation, 10% to the industrial electricity price, and 5% to heating fuel solar (Figure 18).

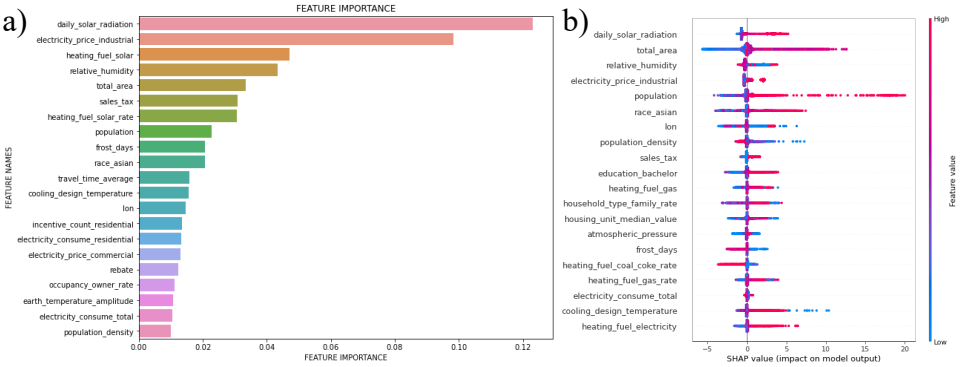

Figure 18. *Feature importance from Non-Residential XGBoost model using XGBoost feature importance (a) and SHAP value (b).*

The Shapley values also ranked daily solar radiation as the most important feature with a high positive relationship with the target variable. The Shapley plot also showed a high positive relationship with total area and a negative relationship with relative humidity. The top five features for both methods did agree; however, there was the small fluctuations as to which feature was ranked in what order for importance.

Neural Network

Lastly, a neural network was built. Initially the neural network was setup as a single layer with 128 perceptrons. Each perceptron takes the sum of all inputs and their weights and applies a non-linear activation function to determine the output (Figure 19).[15]

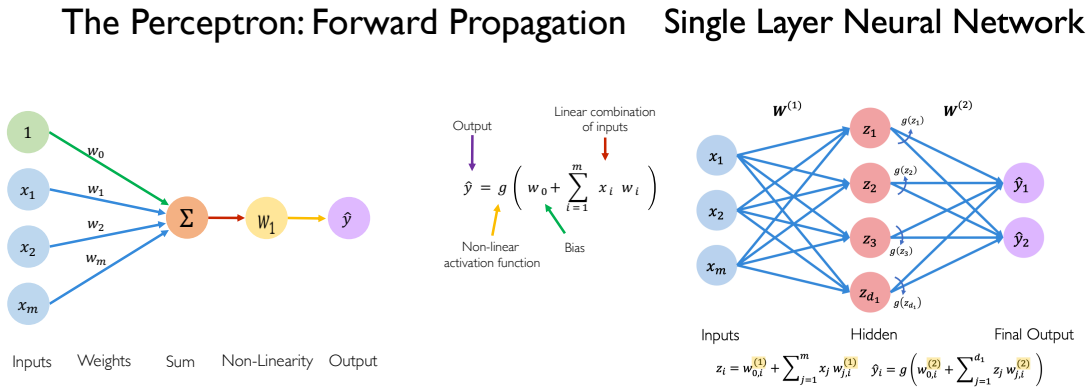

Figure 19. *Perceptron setup and single layer neural network*[15]

The weights for each input are resolved through gradient descent and back propagation using the Chain Rule to multiply derivates. A non-linear rectified linear unit (ReLU) activation function was added to the layer so the model could identify complex relationships between the features. There are several common activation functions used in neural networks, including the sigmoid function, hyperbolic tangent, and ReLU. This model used the ReLU function in order to avoid the issue of the vanishing gradient (Figure 20).[15]

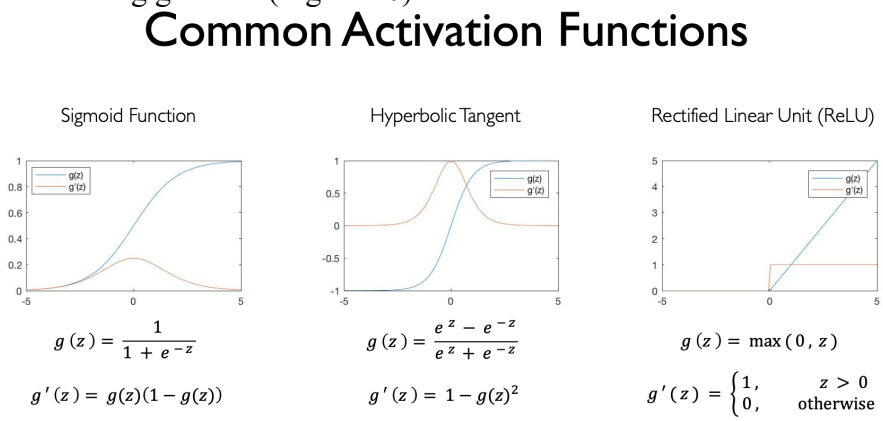

Figure 20*. Common non-linear activation functions used in neural networks.*

The model also incorporated an Adam optimizer to dynamically adapt and optimize the learning rate for the model so as to avoid local minima. Lastly, early stopping was employed to prevent overfitting in the models.

Residential Neural Network

The single layer neural network for the residential model had 13,441 parameters. The MSE for the initial model was 681 and the R^2 value was 0.695 (Figure 21).

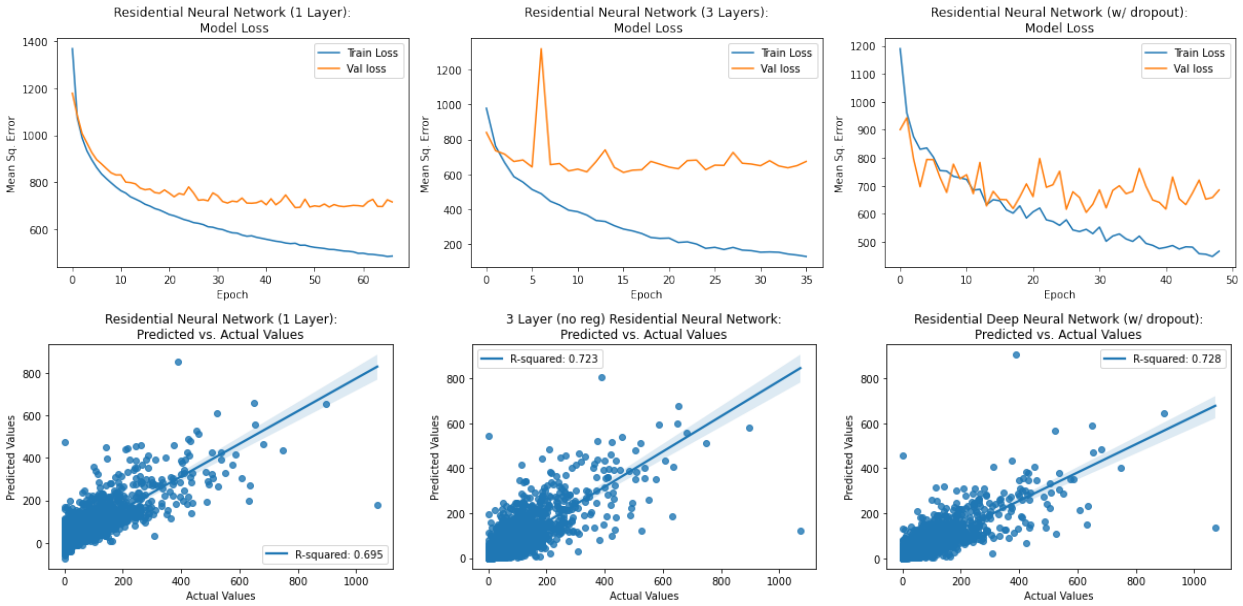

Figure 21. *Neural Network loss and prediction vs. actual values for each model architecture.*

To try to improve model performance, additional hidden layers were added to the neural network. All of the hidden layers had a ReLU activation function and 128 units. The 3 layer neural network had a lower MSE of 593; however, the loss plot indicated some overfitting occurred because the training loss and validation loss diverged quickly at around 3 epochs (Figure 21).

There are several methods to address overfitting in a neural network, including early stopping, dropout, and equal weighting across layers. Early stopping was already established in the model, so a dropout parameter of 10% was added between the second and third hidden layers. The best model included the 3 hidden layers with the ReLU activation function (Figure 22).

Residential Neural Network Model Performance			
Model Architecture	**Parameters**	**MSE**	**R^2**
• 1 hidden layer (128 units, ReLu activation)	13,441	681	0.695
• 3 hidden layers (128 units each, ReLU activation)	46,465	616	0.723
• 3 hidden layers (128 units each, ReLU activation) • 10% dropout between layers 2 and 3	46,465	626	0.728

Figure 22. *Performance of Residential Neural Network Models*

The Shapley values for the model with 20% dropout were assessed and showed that frost days, daily solar radiation, and housing unit median value were the most important predictive features (Figure 23).

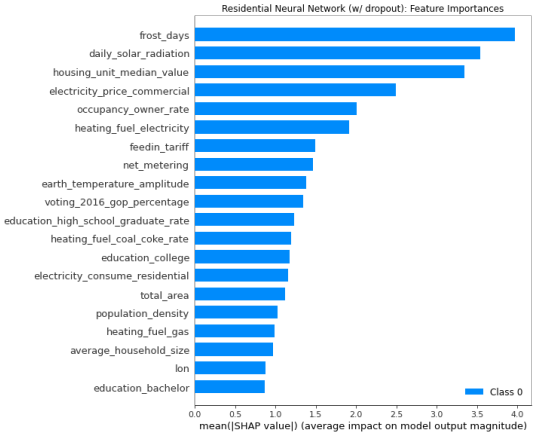

Figure 23. Shapley values for *Residential Neural Network with dropout.*

Non-Residential Neural Network

Several architectures for the non-residential neural network were also implemented. The single layer (128 unit) model with a ReLU activation function had a MSE of 51 and a R^2 value of 0.249 (Figures 24 and 25). Adding 2 additional hidden layers did not improve the model performance. Both of the initial models showed signs of overfitting in the model loss plots. Therefore, a neural network with 3 hidden layers and a 20% dropout between layers 2 and 3 was also trained. This model showed less signs of overfitting and had a MSE of 50 and an R^2 value of 0.241.

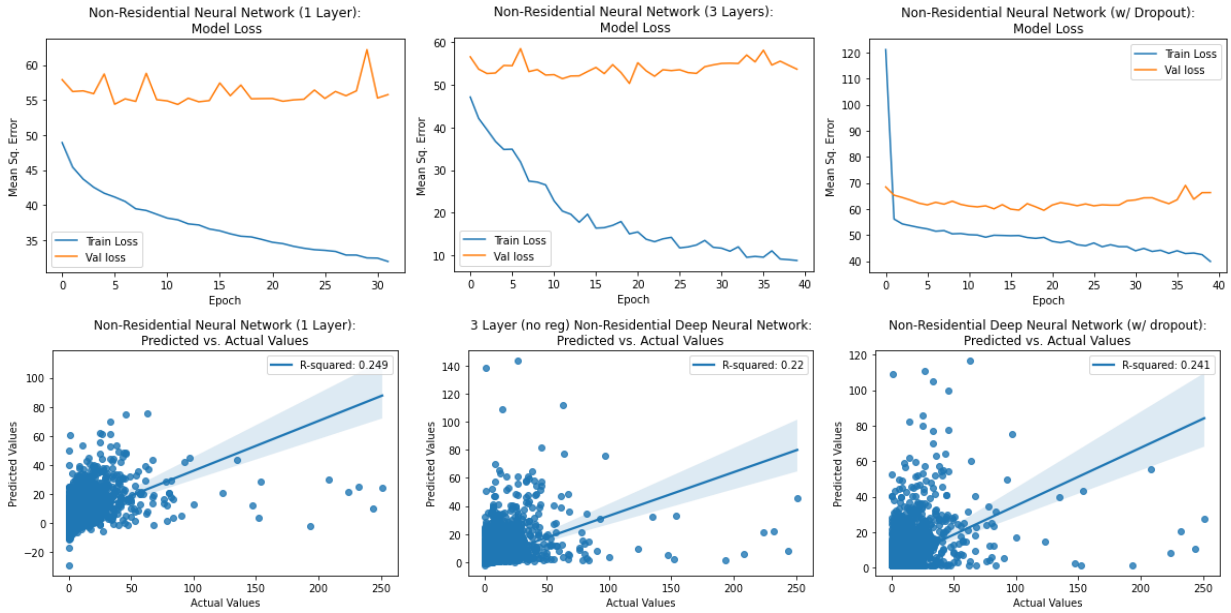

Figure 24. *Neural Network loss and prediction vs. actual values for several non-residential model architectures.*

Non-Residential Neural Network Model Performance			
Model Architecture	**Parameters**	**MSE**	**R^2**
• 1 hidden layer (128 units, ReLU activation)	13,441	51	0.249
• 3 hidden layers (128 units each, ReLU activation)	46,465	53	0.22
• 3 hidden layers (128 units each, ReLU activation) • 20% dropout between layers 2 and 3	46,465	50	0.227

Figure 25. *Performance of Non-Residential Neural Network Models*

The Shapley values for the neural network model with dropout is shown in Figure 26. Population density, frost days, and total area were the top features in predicting solar system count.

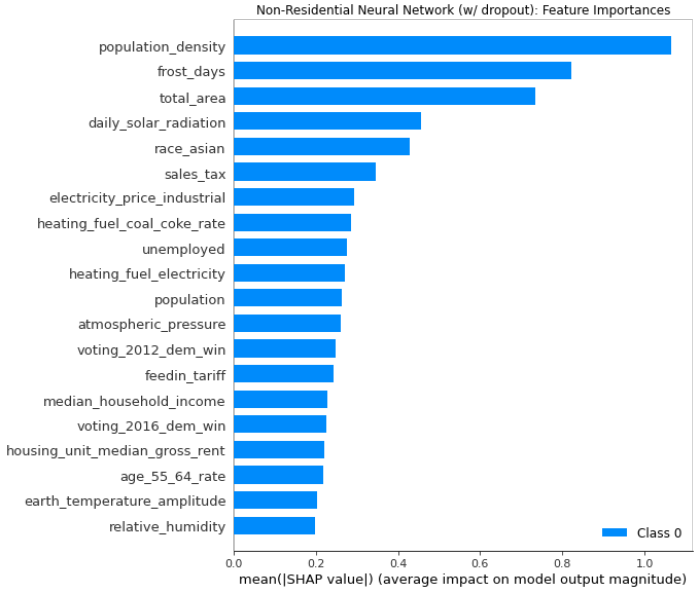

Figure 26. *Shapley values for Residential Neural Network with dropout.*

Conclusion

The best performing residential model was XGBoost. This model had an MSE of 503 and R^2 value for the predictions vs. actual values of 0.769. The top important features in this model were residential incentive count, daily solar radiation, and college education (Figure 27).

XGBoost also performed the best for the non-residential model. This model achieved a MSE of 47 and R^2 of 0.28 (Figure 27). The top important features in this model were daily solar radiation, industrial electricity price, and heating fuel solar.

Target Sector	Method	Architecture	Best MSE	R^2	Top Important Features
Residential	Random Forest	• n_estimators: 1000 • min_samples_split: 5 • min_samples_leaf: 5 • max_features: 'sqrt' • max_depth: 20 • bootstrap: False	552	0.753	Education college, daily solar radiation, heating fuel gas
	XGBoost	• n_estimators: 1000 • learning_rate: 0.01	503	0.769	Incentive count residential, daily solar radiation, college education
	Neural Network	• 3 hidden layers (128 units ea ch, ReLU activation)	616	0.723	Frost days, daily solar radiation, housing unit median value

Target Sector	Method	Architecture	Best MSE	R^2	Top Important Features
Non-Residential	Random Forest	• default	49.6	0.249	Total area, population density, relative humidity
	XGBoost	• reg_lambda: 3 • reg_alpha: 1 • n_estimators: 300 • min_child_weight: 10 • max_depth: 20 • learning_rate: 0.01 • gamma: 0 • col_sample_bytree: 0.8 • booster: gbtree	47	0.28	Daily solar radiation, industrial electricity price, heating fuel solar
	Neural Network	• 3 hidden layers (128 units ea ch, ReLU activation) • 20% dropout between layers 2 and 3	50	0.227	Population density, frost days, total area, daily solar radiation

Figure 27. *Summary of best model performance by type and sector.*

The non-residential models tended to have low correlation between predicted values and actual values. In the future, this model could be improved by performing an initial classification to determine if a FIPS location has any non-residential solar systems before implementing the regression model on only those locations with non-residential systems. This additional step would help to address the class imbalance where most counties do not have any non-residential solar systems.

Further improvements could also include implementing more extensive hyperparameter tuning using a grid search over all settings rather than a random sample. Grid search is a computational and time intensive process and therefore was not within the scope of this project.

The models could be implemented to identify areas likely to have a large amount of residential solar systems and work to establish the sales pipeline in those locations. Additionally, from the residential model it is seen that the number of residential incentives are highly correlated with the number of solar systems in a county. This information could be used to convince politicians to add more incentives to promote solar power adoption at the residential level.

Code

Source code for this project can be found on Github at https://github.com/gnyirjesy/Solar-Panel-Area-Prediction.

Sources

1. IPCC, 2021: Climate Change 2021: The Physical Science Basis. Contribution of Working Group I to the Sixth Assessment Report of the Intergovernmental Panel on Climate Change [Masson-Delmotte, V., P. Zhai, A. Pirani, S.L. Connors, C. Péan, S. Berger, N. Caud, Y. Chen, L. Goldfarb, M.I. Gomis, M. Huang, K. Leitzell, E. Lonnoy, J.B.R. Matthews, T.K. Maycock, T. Waterfield, O. Yelekçi, R. Yu, and B. Zhou (eds.)]. Cambridge University Press. In Press.
2. Kidd, D. (2021, June 16). *US regulatory barriers to an ambitious Paris Agreement Commitment - Environmental & Energy Law Program*. Harvard Law School. Retrieved December 21, 2021, from https://eelp.law.harvard.edu/2021/04/us-paris-commitment/
3. Solar Energy Industry Association. (2021). *Solar Industry Research Data*. SEIA. Retrieved December 21, 2021, from https://www.seia.org/solar-industry-research-data
4. *The deepsolar project*. Home - DeepSolar. (2018). Retrieved December 21, 2021, from http://web.stanford.edu/group/deepsolar/home
5. Yu, J., Wang, Z., Majumdar, A., & Rajagopal, R. (2018). DeepSolar: A machine learning framework to efficiently construct a solar deployment database in the United States. *Joule*, *2*(12), 2605–2617. https://doi.org/10.1016/j.joule.2018.11.021
6. Singhal, S. (2021, June 21). *Imputation techniques: What are the types of imputation techniques*. Analytics Vidhya. Retrieved December 21, 2021, from https://www.analyticsvidhya.com/blog/2021/06/defining-analysing-and-implementing-imputation-techniques/
7. Toloşi, L., & Lengauer, T. (2011). Classification with correlated features: Unreliability of feature ranking and solutions. *Bioinformatics*, *27*(14), 1986–1994. https://doi.org/10.1093/bioinformatics/btr300
8. Pedregosa, F. a. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research, 12*, 2825-2830.
9. Kjytay. (2021, February 19). *The box-cox and Yeo-Johnson transformations for continuous variables*. Statistical Odds & Ends. Retrieved December 21, 2021, from https://statisticaloddsandends.wordpress.com/2021/02/19/the-box-cox-and-yeo-johnson-transformations-for-continuous-variables/
10. Breiman, L. (1999, September). *Random forests - statistics at UC Berkeley*. Retrieved December 21, 2021, from https://www.stat.berkeley.edu/~breiman/random-forests.pdf
11. Gentine, P. (n.d.). *Regression Tree* [PowerPoint presentation]. Retrieved from Columbia University Courseworks.
12. Chen, T., & Guestrin, C. (2016). XGBoost. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. https://doi.org/10.1145/2939672.2939785
13. Pykes, K. (2021, December 14). *Fighting overfitting with L1 or L2 regularization: Which one is better?* neptune.ai. Retrieved December 21, 2021, from https://neptune.ai/blog/fighting-overfitting-with-l1-or-l2-regularization
14. Shapley, Lloyd S. "A value for n-person games." Contributions to the Theory of Games 2.28 (1953): 307-317.
15. Gentine, P. (n.d.). *Neural Networks* [PowerPoint presentation]. Retrieved from Columbia University Courseworks.