Analysis of Solar PV Production Efficiency through R

QTM 2000 – Case Studies in Business Analytics

Silvia Huang, Raymond Huang, Shirley Ying, Xianle Jin
Professor Babak Zafari
December 7th, 2018

"I pledge my honor that I have neither received nor provided unauthorized assistance during the completion of this work."

I. Problem Description

Solar energy, as one of the cleanest and most abundant renewable energy resource available, has been largely developed with the advancement of technology and implemented in different customer segments. Government incentive programs and the reduction in solar energy infrastructure costs have prompted individuals and households to consider adopting this new technology. However, the increasing rate of adoption in solar energy has sparked disruptions in both utility and oil industries, which signifies a shift in customer demand for energy and presents an interesting dynamic in the energy market. To explore more about this technology, our team is particularly interested in its energy production efficiency, which will be valuable information to end customers for decision-making.

When making the decision to install solar panels, a solar power contractor is often hired to work with the individual and to provide information about the amount of space required for installation, module efficiency, system size, and so on. Therefore, based on this available information, the team aims to construct a model to predict the annual photovoltaic (PV) production to better educate customers on the realistic return of implementing solar panels and to help them make informed decisions.

II. Dataset Description

A. Dataset Searching & Cleaning

The team obtained the two raw datasets from *The Open PV Project* portal through the National Renewable Energy Laboratory website.[1] The full Open PV Dataset contains 1,020,813 records and 81 variables. The *Tracking the Sun* Public Data File, which was collected and processed by Lawrence Berkeley National Laboratory based on the full dataset, contains 1,094,909 records and 52 variables. We found that the full dataset and the Public Data File each

[1] https://openpv.nrel.gov/search

contains several predictor variables that were eliminated in the other. Therefore, we performed an inner join using the "inner_join" function in the "dplyr" package[2] to merge the two datasets into one with 80,317 records and 129 variables. Due to the lack of ID numbers in the full dataset, we joined the two datasets based on four primary keys, including Installation Date, System Size, Zip Code, and Installer Name. Although the combination of these four primary keys was quite unique, there were still duplicates generated from this merge. Then, the team excluded duplicated rows and eliminated all columns with no data entries or relevance to the target variable. To this point, the dataset had 49,898 records and 7 variables. A detailed description of each variable can be found in the Data Dictionary shown in Appendix 1.

To deal with missing values, the team decided to not eliminate all records that contained missing values, so we treated the numeric variables and categorical variables separately. For numeric ones, we replaced all missing values with the mean of their respective column. In this way, we did not have to reduce the number of records that we could study on. For categorical variables, we removed all records that had categorical missing values, because whether a solar system is ground-mounted or rooftop and whether it has a DC Optimizer are known facts before producing energy, we did not want to make assumptions regarding these two variables. We also removed all records that had missing values in the Annual Energy Production column, since it is the target variable of interest, we wanted to build the models on real data. As a result, the cleaned dataset has 8,718 records and 7 variables.

B. Descriptive Statistics & Explanatory Analysis

Before building any model, the team examined the descriptive statistics of the dataset using the "summary" function (See Appendix 2). We found that the values of the Annual Energy

[2] http://zevross.com/blog/2014/04/30/mini-post-for-large-tables-in-r-dplyrs-function-inner_join-is-much-faster-than-merge/

Production variable, for example, range from 6 to 35,000,000, which appear to have very extreme values in the dataset. Therefore, we constructed a boxplot for each of the numeric variables to see if there was any outlier (See Appendix 3). We identified and removed three records based on outliers found in the Total Installed Price column and the Annual Energy Production column. Since the original data from the *Open PV Project* were contributed voluntarily from a variety of sources such as solar incentive programs, utilities, installers, and, the general public,[3] this step of removing outliers would help us eliminate potential input errors in data entries. As a result, the final dataset has 8,715 records and 7 variables.

After removing the outliers, the team ran an explanatory linear regression model on the final dataset in order to explain the relationships between the predictor variables and the target variable through coefficients (See Appendix 4). The linear regression formula is Annual Energy Produced = 4.769e+04*Module Efficiency + 1.336e+03*System Size + 1.042e+03*DC Optimizer + 3.659e+04*Annual Insolation - 2.022e-02*Total Installed Price - 1.124e+03*Ground Mounted - 2.554e+04. Module Efficiency has the greatest positive impact on the target variable. One unit of increase in Model Efficiency will increase the annual production of energy by 4.769e+04 kWh on average. This relationship corresponds with our assumption that if the Module Efficiency is high, we expect the annual energy produced to be high. Total Installed Prices has the greatest negative impact on the target variable. One unit of increase in Total Installed Price will decrease the annual energy production by 2.022e-02 kWh on average. This relationship was surprising for us since we expected the annual energy produced to be high if one paid more for the solar system.

C. Exploratory Analysis - Tableau

[3] https://openpv.nrel.gov/about

After conducting the explanatory analysis, we decided to explore more about the relationships among critical variables to gain a better understanding about the project. Through Tableau, we conducted analyses on the final dataset with several additional columns, such as State, Installation Date, and Customer Segment. First, the team wanted to have a clearer picture about how our target variable, Annual Energy Production, relates to the different geographical locations within the U.S. In this way, we were able to gain a concrete idea about the current stage of solar energy development across different states. According to Appendix 5, Massachusetts has drastically more Annual Energy Production than any other state has. Meanwhile, MA is in a leading position on the Annual Energy Production level despite the fact that it has a locational disadvantage due to a lower Annual Insolation. In addition, the Average Annual Energy Production of MA is also lower than that of CA. As a result, the team believes that MA has relatively more solar panels installed in compensation of its lower Average Annual Energy Production and locational disadvantage. Our assumption was also validated after adding the number of records as marks. MA has 5,473 records while CA only has 10 records in the dataset.

After knowing how solar energy production varies between different states, the team tried to explore how key predictor variables can influence the performance of individual solar panel. Therefore, the team plotted the relationships between Module Efficiency, System Size, and Annual Isolation against the Average Annual Energy Production. Based on the explanatory model mentioned above, Module Efficiency should have a larger impact on Annual Energy Production than other variables. However, when we visualized the relationship as shown in Appendix 6, we observed that Module Efficiency mainly ranges from 14% to 16% and does not have a strong positive relationship with the Average Annual Energy Production.

However, System Size has shown a positive relationship with the Average Annual Energy Production (See Appendix 7), which corresponds to the positive coefficient of System Size in the explanatory model. With the increase in System Size, the Average Annual Energy Production increases proportionally. Furthermore, the team observed the relationship between the Average Annual Insolation and the Average Annual Energy Production. According to Appendix 8, when the Annual Energy Production increases, the Annual Insolation does not change accordingly. It has a relatively flat trendline. Overall, these visualizations suggest that System Size has a critical impact on the Annual Energy Production.

D. Data Transformation

After data preprocessing and exploration, the team tested assumptions of normality for the four numeric predictor variables using histograms. Based on the graphs in Appendix 9-1, System Size and Total Installed Price are strongly skewed to the right. In order to measure the skewness, the team used the function of "skewness" in the "e1071" package (See Appendix 9-2). When the skewness value lies above +1 or below -1, the data is interpreted as highly skewed. Therefore, Module Efficiency and Annual Insolation are moderately skewed in comparison to System Size and Total Installed Price. System Size and Total Installed Price both have skewness values that lie significantly above 1, which shows these two variables are highly skewed and need to be transformed. Common transformations for right-skewed data include square root, cube root, and logarithm. The team tested out all three transformations on System Size and Total Installed Price, and compared their skewness values to determine the best transformation (See Appendix 9-3). The logarithm transformation gave the best results. However, the skewness values for System Size and Total Installed Price are still above 1. The team then performed a second round of data transformation upon these two variables. We used square root

transformation and tested the skewness (See Appendix 9-4). System Size became almost normal; however, Total Installed Price had no significant improvement after the first round of data transformation. Then, instead of performing square root transformation on Total Installed Price, the team did logarithm transformation (See Appendix 9-5). Although Total Installed Price is still highly skewed, the team decided to stop transforming since it is already significantly more normal than the original data, and further transformation will not significantly improve its normality. After data transformation of data, the team changed the variables names for easier reference.

III. Predictive Modeling & Validation Testing

A. Linear Regression Model

For the linear regression model, the team first did cross-validation, splitting the dataset into training set, containing 80% of all data, and validation set, containing 20% of all data, to prevent the final model from overfitting. Next, the team built the base model based on the training dataset, which included all predictor variables. However, it was hard to know if all variables are necessary for building the final model, so the team performed variable selections using forward, backward, and stepwise methods to build three other models on the same training dataset. The team tested four models on the validation set and compared their RMSE values. For example, one of the models with the lowest RMSE shows Annual Energy Produced = -517,931*Module Efficiency + 34,643*Annual Insolation + 1,672,428*Total Installed Price + 480,589*Ground Mounted - 3,958,127 (See Appendix 10). With ten different seeds, the team tested the models built on the validation dataset and recorded each round's lowest RMSE value for the final comparison between models.

B. Regression Tree Model

After cross-validation, the team made sure that all categorical variables such as DC Optimizer and Ground Mounted are treated as factors in R. The package used for creating the tree are "rpart," "rpart.plot," and "rattle". After models are created, the team used the accuracy function in the "forecast" to validate our prediction. In the Regression Tree Model, cp values are crucial since any split that does not increase the overall R-Squared value by the cp threshold will be eliminated. The team first ran a default model using the "rpart" function and set the method as "anova," which is specifically for regression. Then, the team created an unpruned model using rpart.control (cp = 0, minsplit = 2, minbucket = 1). The team experimented with different minsplit and minbucket values and manually pruned the tree. Through the "printcp" function, the team found that a cp value below 0.098 will no longer have sufficient impact on xerror (See Appendix 11). Therefore, the control parameter set for manual prune is (cp = 0.098, minsplit=10, minbucket=2). The graphs of three pruning methods are included in Appendix 12. The manually pruned model and the default model both used only System Size as the predictor variable, which may not be optimal in the real world. After iterating ten times with ten seeds, the unpruned tree seems to produce the lowest RMSE in most scenarios. However, the unpruned tree took all the variables into consideration which may potentially be overfitting.

C. k-Nearest Neighbors (k-NN) Regression Model

Before running the k-NN regression model, the team first converted the two binary categorical variables to numeric, because k-NN model is based on distance calculations and only includes numeric variables. Based on our understanding, whether the solar system has a DC Optimizer and whether it is ground-mounted or rooftop would impact the amount of energy produced; therefore, the team chose to convert instead of excluding the two categorical variables. After the conversion, the team re-split the dataset into training and validation datasets.

For building the model, the team first found the k value with the highest accuracy using the "knn.cv" function in the "class" package. This function allowed us to run a leave-one-out cross-validation and to test the accuracy across different k values as shown in Appendix 13. The purpose of this step is to identify a k value to start building the model with, so from k = 1, the team selected k = 3, 5, 9, 11, 25 to build six models and compare their RMSE on the validation dataset. For running a regression model using k-NN, the team chose to use the "knn.reg" function in the "FNN" package instead of the "kNN" function in the "DMwR" package that we learned in class for classification. Out of the ten times the team ran the models with different seeds, while it is true that the model with k = 1 usually had the lowest RMSE, there were several times in which the model with k = 5 or k = 9 had the lowest RMSE. Since k = 1 is an extreme in capturing local structure of the data and may lead to overfitting, for those few times, we chose to use the model with k = 5 and k = 9 to compare with the Linear Regression Model and the Regression Tree Model.

IV. Validation Analysis & Model Evaluation

After running models for all three methods separately with ten different seeds, we selected the most accurate model (i.e. with the lowest RMSE) for each method and for each of the ten seeds. Then, we compared the accuracy of each method based on the RMSE values of their best models (See results in Appendix 14). From the comparison, we concluded that k-NN Regression is the best predictive method for our project. Out of the ten times we ran the models with different seeds, there was a 100% chance that k-NN Regression Model generated the lowest RMSE, which means that its predictive power is independent from the seeds.

After identifying the best model, our team tried to improve its predictive performance by rescaling the data before running the model. We rescaled the training and validation datasets

separately but with the parameters of the training dataset. In doing so, we used the "preProcess" function in the "caret" package, which allowed us to save the transformation specifications and apply them to both training and validation datasets. We saved the rescaled data into new training and validation datasets to prevent confusions. Then, we repeated the process of running the k-NN Regression Model across the same set of k values. To compare the predictive accuracy of the original models and the new models, we chose to focus on MAPE since it is a percentage value and would not be affected by the scale of the data. Surprisingly, the team found that the MAPE values of the rescaled models were much higher than those of the original models (See results in Appendix 15). Therefore, the rescaled transformation did not help improve the model but worsened it. Our team believed that it was because we already transformed some variables before constructing any model, and rescaling may backfire on the predictive power of the model. Some explanations online also suggest that some variables do have more significance in predicting the target variable; rescaling may reduce their significance and thus lead to inaccurate predictions.[4]

V. Conclusion

In conclusion, based on our model, customers can use the predicted Annual PV Production and compare with their current utility usages in order to assist them in the decision-making process. For those that have already installed solar panels, they can adjust settings based on module efficiency or add additional panels to maximize energy production. Last but not least, more policy assistance on solar energy research will be needed to standardize the data collection process. As our data cleaning process has excluded over 99.15% of our original dataset, our final dataset might not be able to become a holistic representation of the real world situation. To

[4] https://stats.stackexchange.com/questions/189652/is-it-a-good-practice-to-always-scale-normalize-data-for-machine-learning

obtain more accurate results regarding solar energy production, a more standardized procedure should be imposed on data collection and upload.

VI. Lessons Learned

This entire project was an invaluable learning experience for the team from an analytic standpoint. From searching for an appropriate dataset to successfully building, running, evaluating, and improving the predictive models, the team was constantly experiencing the iterative process of finding problems and solving problems. For example, when dealing with missing values, we replaced the numeric missing values with the mean of each column and removed all records that contained categorical missing values. During the process, we learned that one single problem can be solved in various different ways, and the most creative solutions may generate the most surprising results. Therefore, if time allowed, we would treat the missing values with different imputation methods, such as the k-NN imputation that we learned online to replace the categorical missing values with their predicted classes. These iterative processes really allowed us to constantly reflect on our approaches and try to improve on them.

In addition, in solving problems and overcoming challenges, the team has to utilize its collective experience and knowledge. The completion and success of this project cannot be achieved with individual effort, but with a dedicated team. The amount of time commitment, perseverance, and patience required brought the team together to strive for the same goal and toward the same direction. For example, our members were constantly collaborating to validate each other's methodologies and brainstorming different solutions to any issue encountered. This collaborative process helped us create an applicable model that represents the team as a whole.

Appendixes

Appendix 1: Data Dictionary

Data Field Name	Units	Description and Key Notes
System Size	kW	The total rated direct-current (DC) output of the module arrays at standard test conditions. These data are generally reported directly by the data provider, but in some cases must be estimated.
Ground Mounted	N/A (Categorical)	If the system is ground-mounted (1) or rooftop (0). Ground-mounted: A system of solar panels that are mounted on the ground on property, rather than on the roof of house. They can be placed anywhere from a few inches to a few feet off the ground.
Module Efficiency	%	The rate of solar panels converting solar energy into electrical energy.
Annual Insolation	$kWh/m^2/day$	The daily average insolation for the Earth is approximately 6 kWh/m^2 = 21.6 MJ/m^2. The output of a photovoltaic panel partly depends on the angle of the sun relative to the panel.
DC Optimizer	N/A (Categorical)	If the system has a DC Optimizer (1) or not (0). A DC optimizer is a DC to DC converter technology developed to maximize the energy harvest from solar photovoltaic. Each optimizer individually tunes the performance of the panel through maximum power point tracking, and optionally tunes the output to match the performance of the string inverter.
Total Installed Price	$	The total installed price for the system, prior to receipt of any incentives.
Annual Energy Production	kWh/yr	The amount of energy produced annually by the PV system. This measurement is represented as kWh per square meter of panel surface.

Appendix 2: Descriptive Statistics

Before removing outliers:

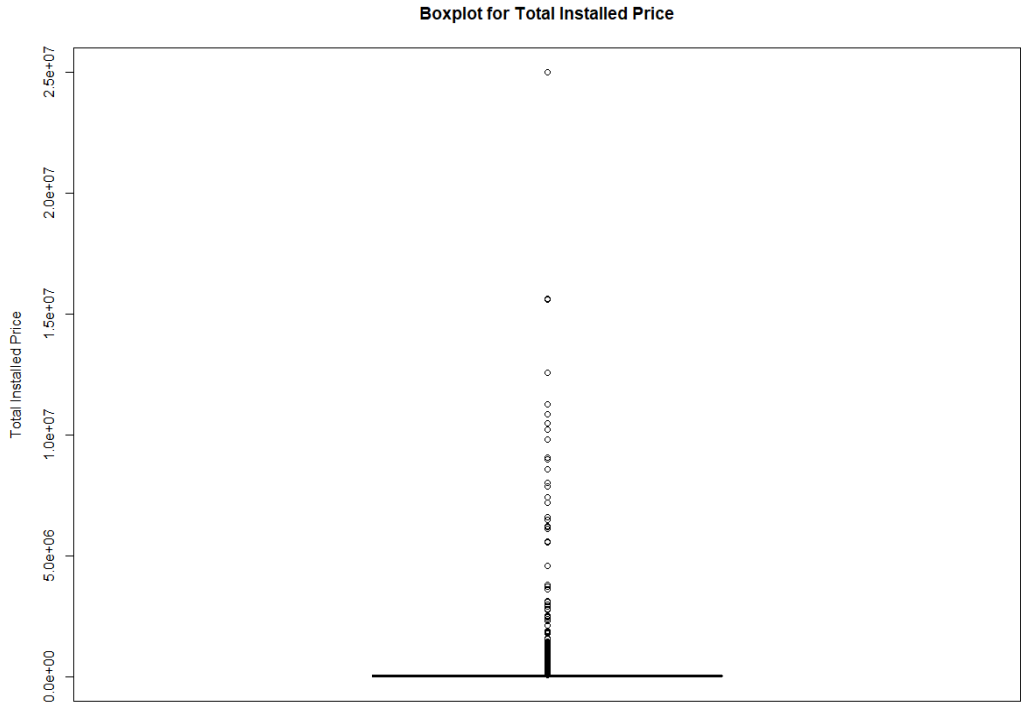
```
> summary(new_innerjoin)
 Module.Efficiency..1  System.Size       DC.Optimizer annual_insolation Total.Installed.Price Ground.Mounted reported_annual_energy_prod
 Min.   :0.0620       Min.   :   0.490   -9999:   0   Min.   :1.335     Min.   :    2064      -9999:   0     Min.   :       6
 1st Qu.:0.1607       1st Qu.:   4.605   0    :5802   1st Qu.:4.112     1st Qu.:   21925      0    :8424     1st Qu.:    4753
 Median :0.1614       Median :   6.500   1    :2916   Median :4.300     Median :   30870      1    : 294     Median :    6712
 Mean   :0.1614       Mean   :  34.068               Mean   :4.268     Mean   :  114675                      Mean   :   43079
 3rd Qu.:0.1614       3rd Qu.:   9.065               3rd Qu.:4.474     3rd Qu.:   42224                      3rd Qu.:    9506
 Max.   :0.2116       Max.   :5998.160               Max.   :6.498     Max.   :25000000                      Max.   :35000000
```

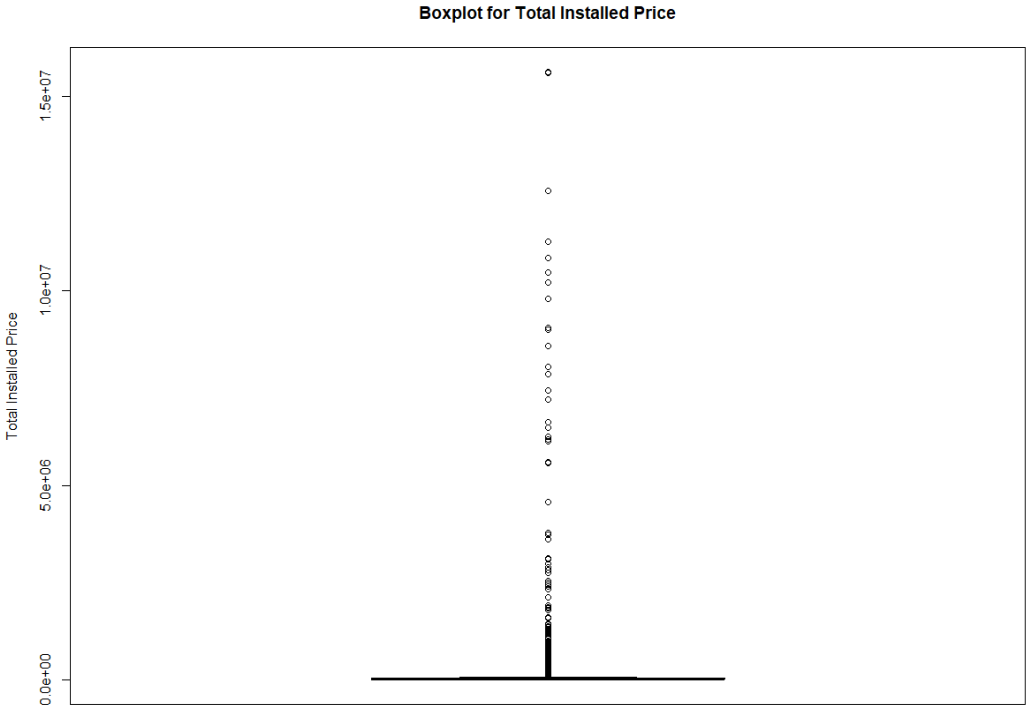
After removing outliers:

```
> summary(new_innerjoin)
 Module.Efficiency..1  System.Size       DC.Optimizer annual_insolation Total.Installed.Price Ground.Mounted reported_annual_energy_prod
 Min.   :0.0620       Min.   :   0.490   -9999:   0   Min.   :1.335     Min.   :    2064      -9999:   0     Min.   :       6
 1st Qu.:0.1607       1st Qu.:   4.600   0    :5799   1st Qu.:4.112     1st Qu.:   21921      0    :8424     1st Qu.:    4753
 Median :0.1614       Median :   6.500   1    :2916   Median :4.300     Median :   30870      1    : 291     Median :    6712
 Mean   :0.1614       Mean   :  32.303               Mean   :4.268     Mean   :  107535                      Mean   :   39077
 3rd Qu.:0.1614       3rd Qu.:   9.065               3rd Qu.:4.474     3rd Qu.:   42224                      3rd Qu.:    9506
 Max.   :0.2116       Max.   :5951.115               Max.   :6.498     Max.   :15625121                      Max.   :7887910
```

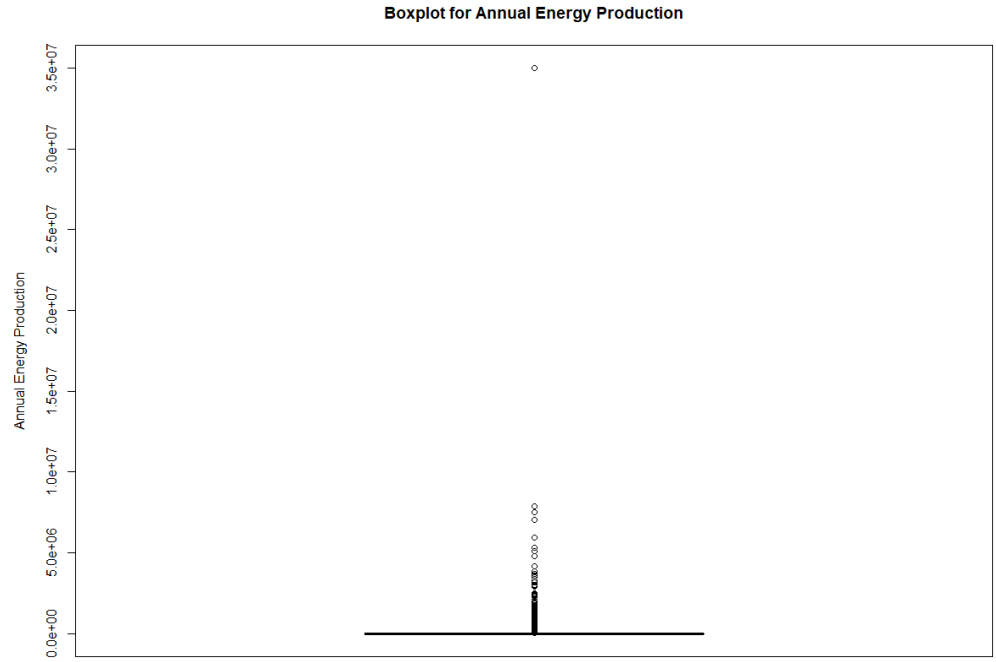
Appendix 3: Boxplots for 2 Numeric Variables

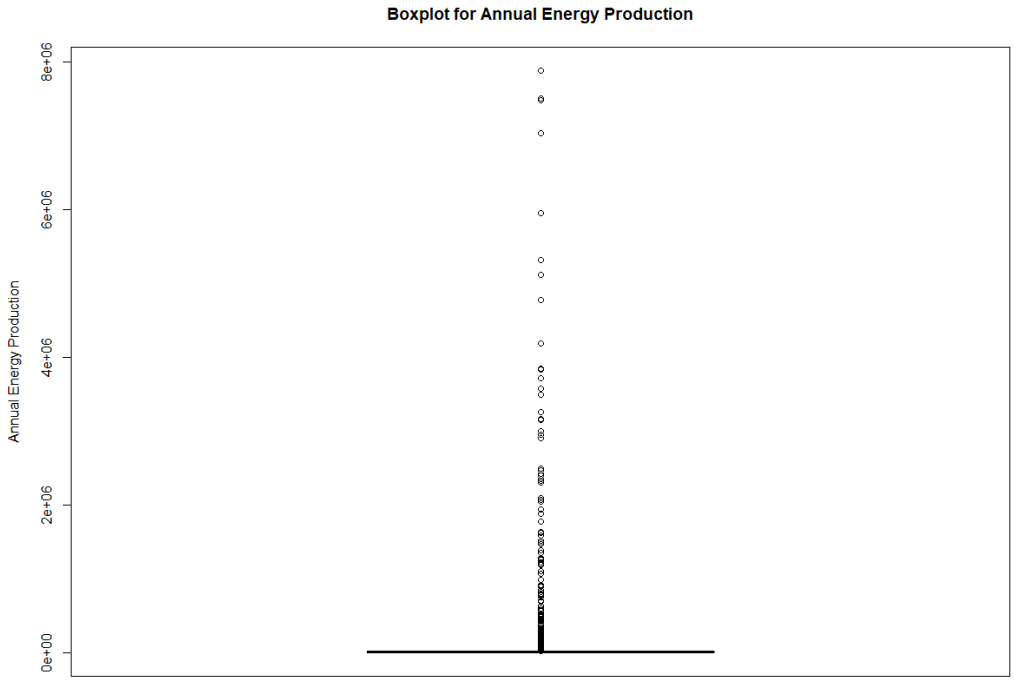
3-1. Total Installed Price: Before (Above) & After (Below) removing outliers





3-2. Annual Energy Production: Before (Above) & After (Below) removing outlier

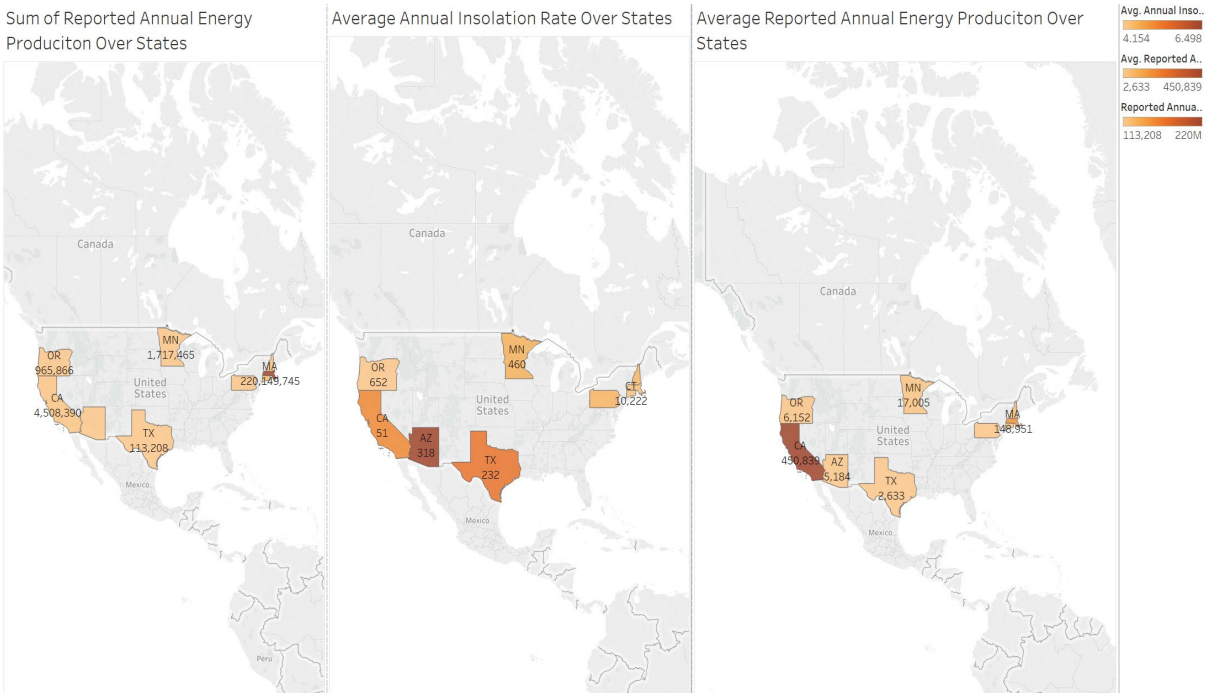




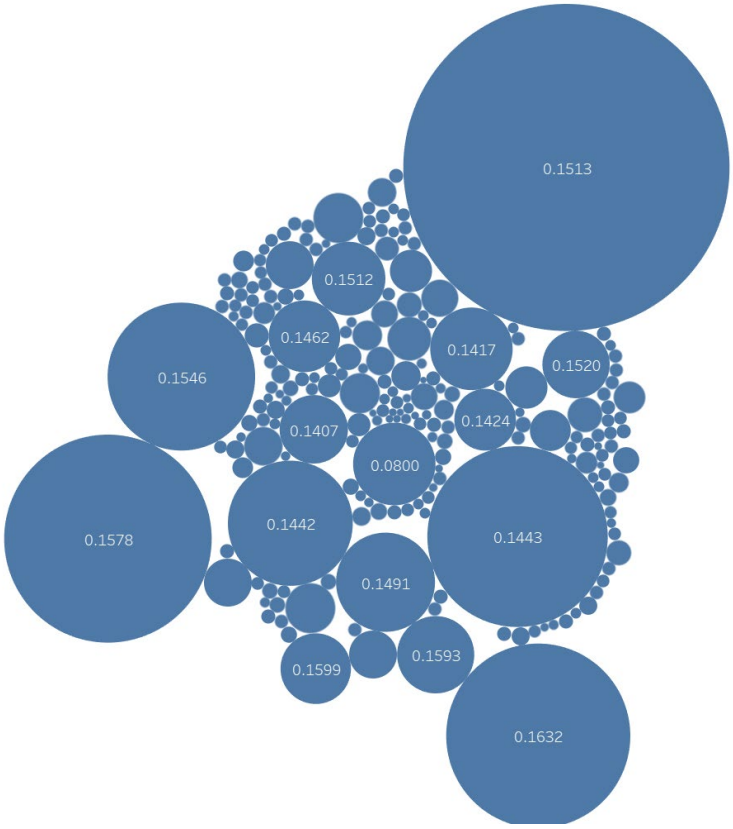
Appendix 4: Explanatory Linear Regression Model on the Whole Dataset

```
> reg_exp=lm(reported_annual_energy_prod~.,new_innerjoin)
> summary(reg_exp)

Call:
lm(formula = reported_annual_energy_prod ~ ., data = new_innerjoin)

Residuals:
    Min      1Q  Median      3Q     Max
-916247    -363     864    1694 3263857

Coefficients:
                      Estimate Std. Error t value Pr(>|t|)
(Intercept)          -2.554e+04  8.968e+03  -2.848  0.00441 **
Module.Efficiency..1  4.769e+04  3.840e+04   1.242  0.21425
System.Size           1.336e+03  4.184e+00 319.425  < 2e-16 ***
DC.Optimizer1         1.042e+03  9.517e+02   1.095  0.27347
annual_insolation     3.659e+03  1.302e+03   2.810  0.00497 **
Total.Installed.Price -2.022e-02  1.447e-03 -13.980  < 2e-16 ***
Ground.Mounted1       -1.124e+03  2.761e+03  -0.407  0.68395
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 40910 on 8708 degrees of freedom
Multiple R-squared:  0.982,     Adjusted R-squared:  0.982
F-statistic: 7.932e+04 on 6 and 8708 DF,  p-value: < 2.2e-16
```

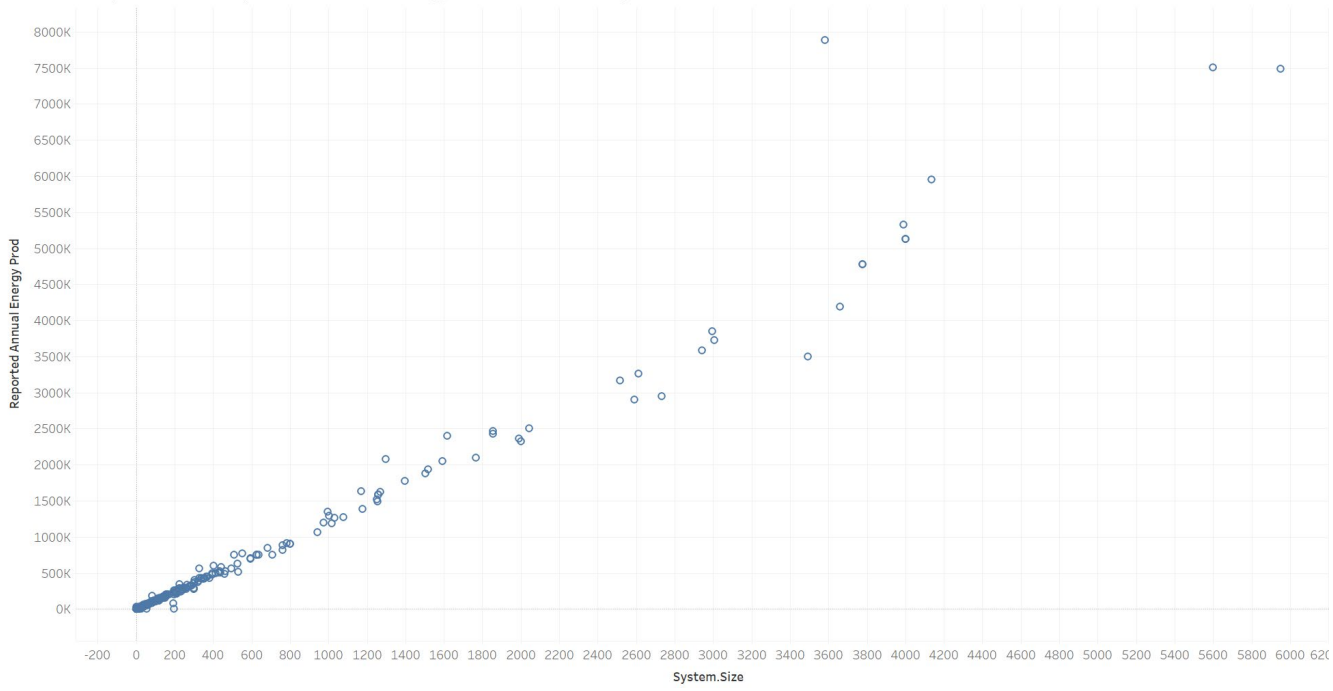
Appendix 5: Sum of Annual Energy Production/Average Annual Insolation/Average Annual Energy Production across States



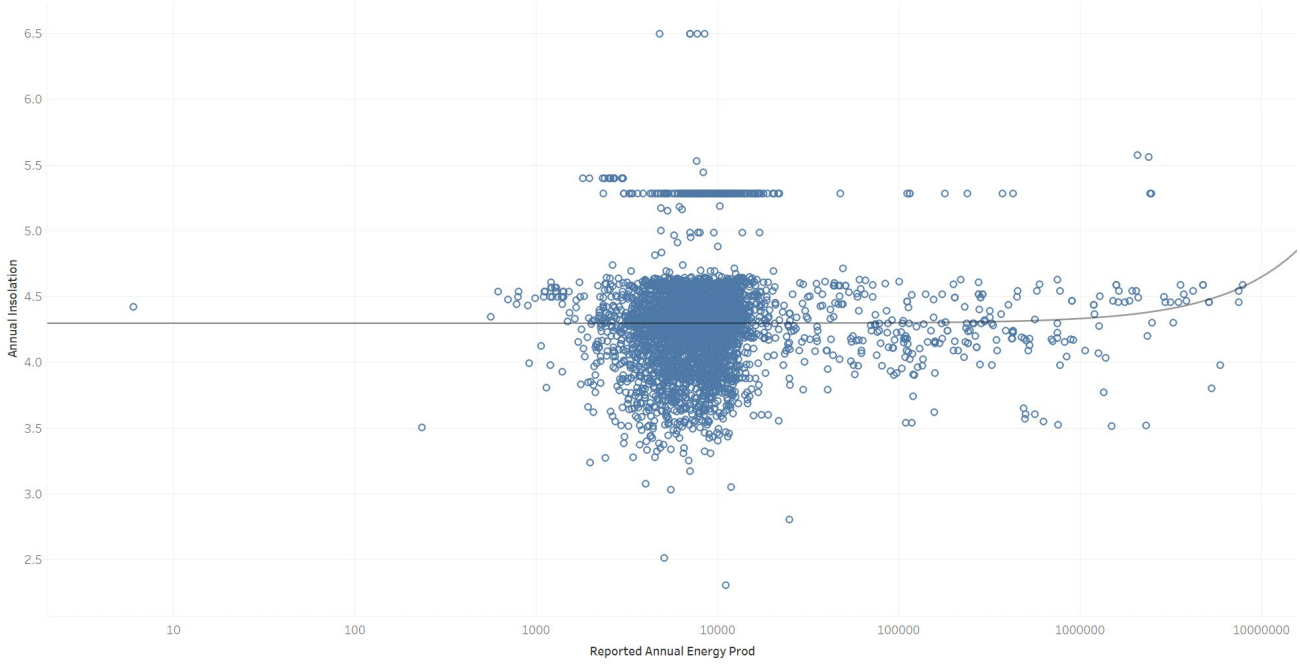
Appendix 6: Module Efficiency over Average Annual Energy Production



Appendix 7: Relationship between Annual Energy Production and System Size

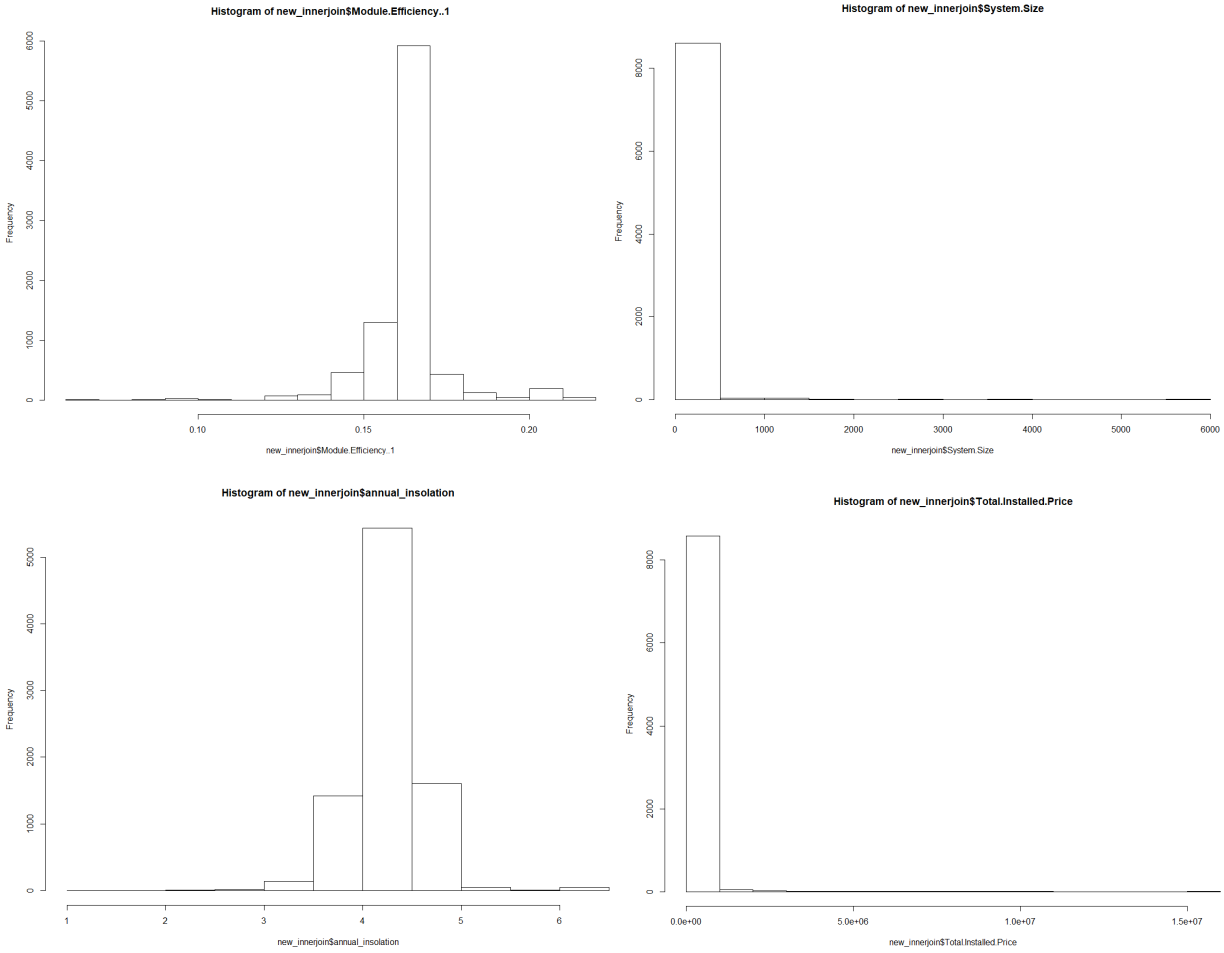


Appendix 8: Relationship between Annual Insolation and Annual Energy Production



Appendix 9: Data Transformation Tables

9-1. Histograms of Four Numeric Predictor Variables



9-2. Skewness Comparison

Variable	Module Efficiency	System Size	Annual Insolation	Total Installed Price
Skewness	0.2116959	14.9736	0.7988914	14.0541

9-3. Data Transformation Round 1

Transformation	System Size (Skewness)	Total Installed Price (Skewness)
Original Data	14.9736	14.0541
Cube Root	6.539633	6.650128
Log	**2.610688**	**3.066911**
Square Root	8.600298	8.615218

9-4. Data Transformation Round 2

Transformation	System Size (Skewness)	Total Installed Price (Skewness)
Square root	0.1032942	2.685144

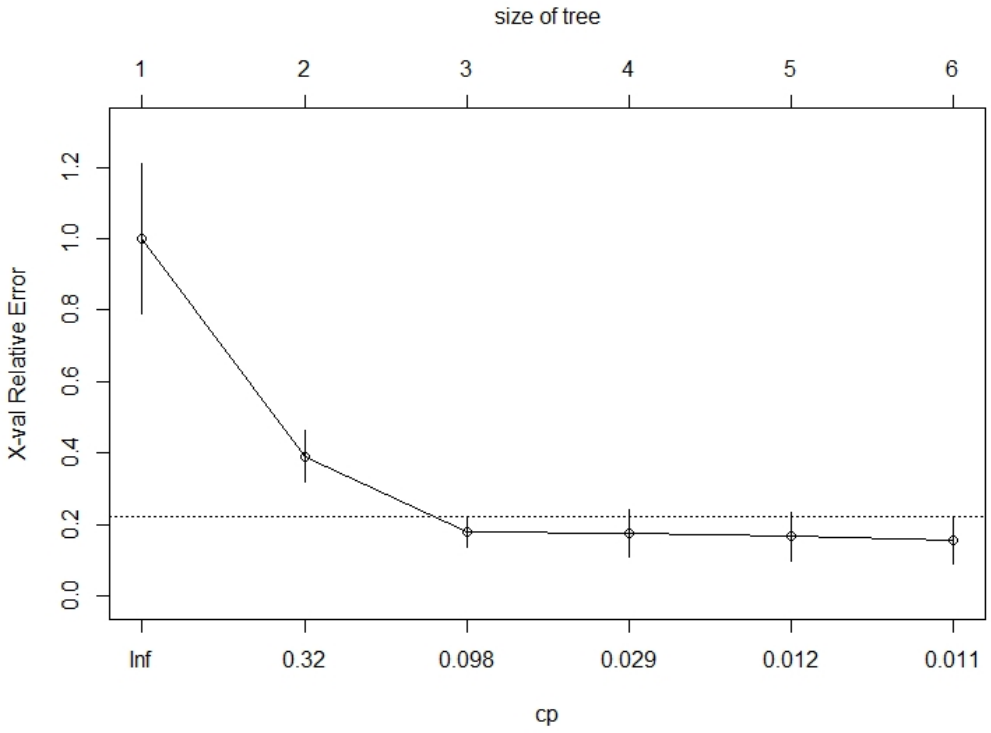
9-5. Re-apply data transformation on Total Installed Price

Transformation	Total Installed Price
Log	2.317743

Appendix 10: Variable Selection: Stepwise Model

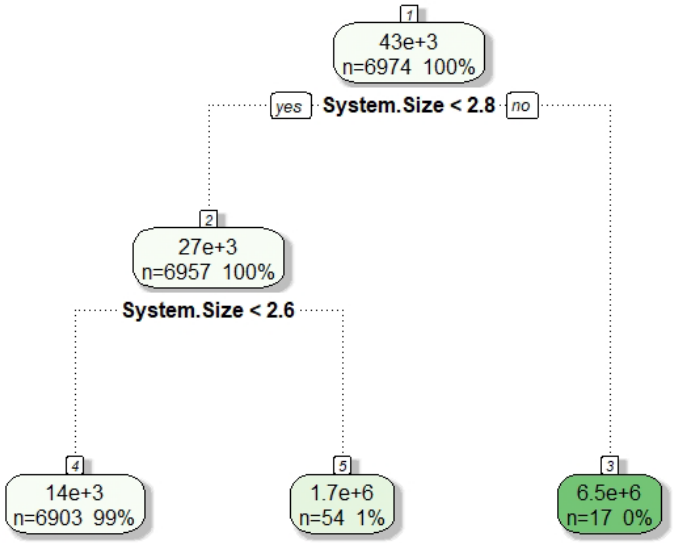
```
> summary(reg_both)

Call:
lm(formula = Annual.Energy.Prod ~ Module.Efficiency + Annual.Insolation +
    Total.Installed.Price + Ground.Mounted, data = training)

Residuals:
    Min      1Q  Median      3Q     Max
-876746  -51347   -3140   51285 6666183

Coefficients:
                     Estimate Std. Error t value Pr(>|t|)
(Intercept)           -3958127     127786 -30.975  < 2e-16 ***
Module.Efficiency      -517931     267318  -1.938 0.052723 .
Annual.Insolation        34643       8999   3.850 0.000119 ***
Total.Installed.Price  1672428      45761  36.547  < 2e-16 ***
Ground.Mounted1         480589      18620  25.810  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 254300 on 6967 degrees of freedom
Multiple R-squared:  0.3196,    Adjusted R-squared:  0.3192
F-statistic: 818.2 on 4 and 6967 DF,  p-value: < 2.2e-16
```

Appendix 11: Graph of cp

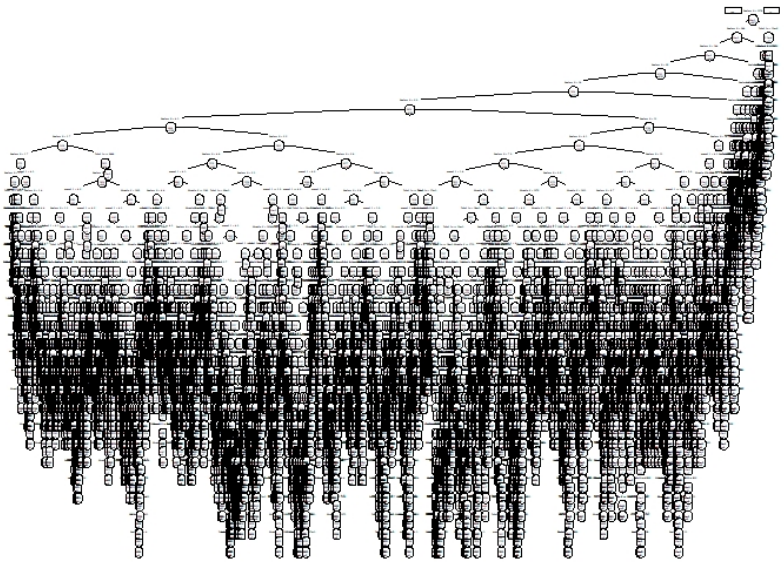


Appendix 12: Three Pruning Methods

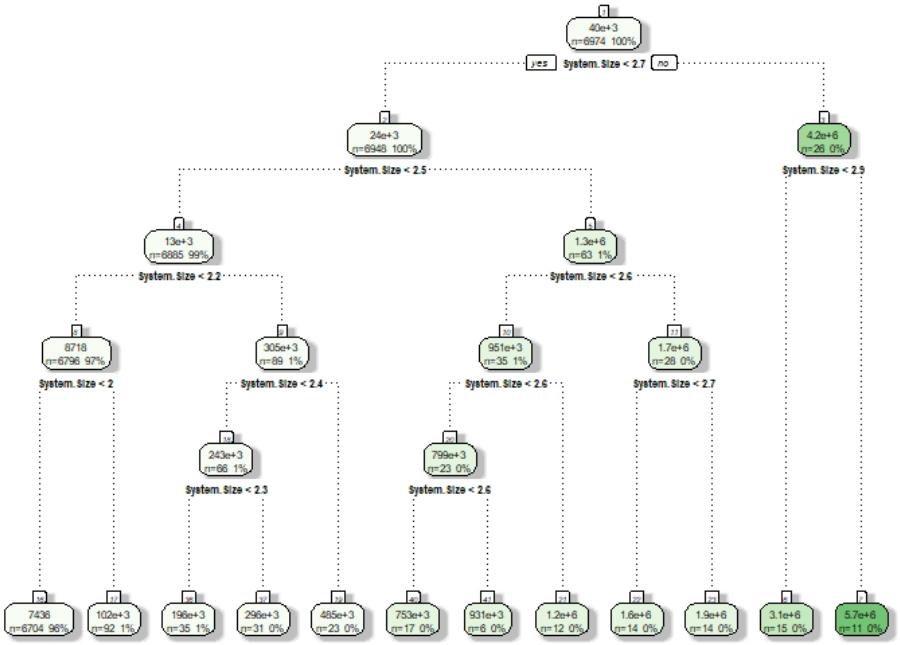
Default Tree



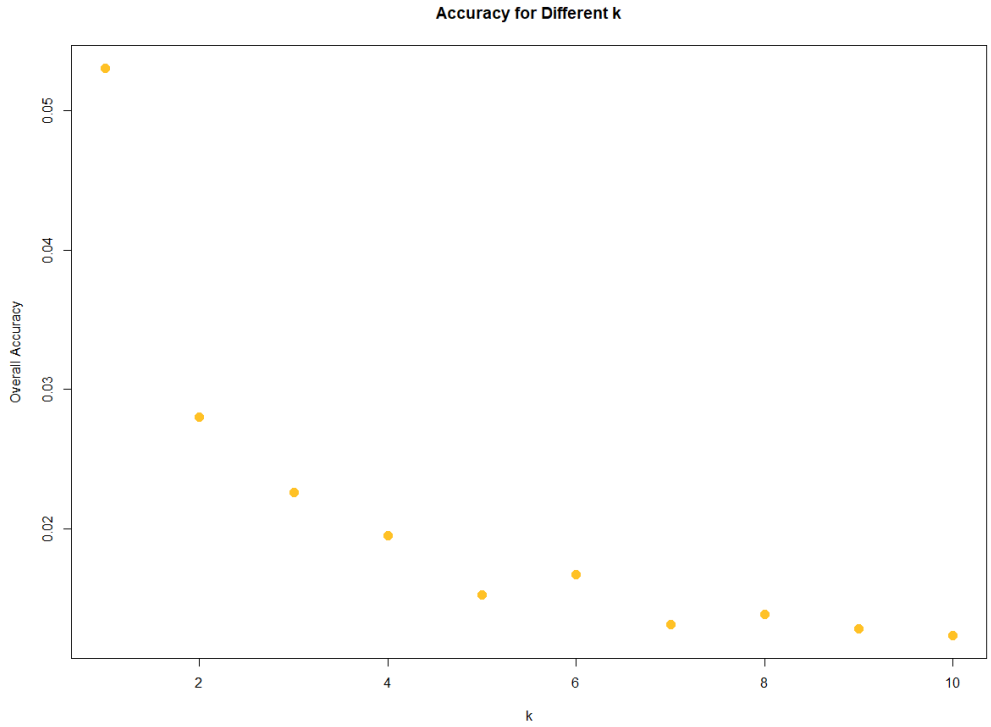
Unpruned Tree



Manually Pruned Tree



Appendix 13: Accuracy across Different k Values



k	1	2	3	4	5	6	7	8	9	10
Accuracy	**0.053**	0.028	0.023	0.02	0.015	0.017	0.013	0.014	0.013	0.012

Appendix 14: Accuracy Comparison

The following table summarizes the RMSE and MAPE values of the best model for each of the three predictive methods (Linear Regression, Regression Tree, k-NN Regression) and for each of the ten seeds. Recording of the ten seeds here is for easier comparison.

Seed	Linear Regression Model		Regression Tree		k-NN Regression		
	RMSE	MAPE	RMSE	MAPE	RMSE	MAPE	Best Model
666666	217888.4 Base (Module.Efficiency + System.Size + DC.Optimizer + Annual.Insolation + Total.Installed.Price + Ground.Mounted)	1785.463	23,731.36 Manual Pruned (System.Size)	71.94	11,345.74 k=5	1.93847	k-NN
12345	203946.5 Forward, Backward, Both (Module.Efficiency + Annual.Insolation + Total.Installed.Price + Ground.Mounted)	1410.289	24,618.92 Unpruned (System.Size)	17.609	7,586.465 k=9	0.1229383	k-NN
250	195685.4 Forward, Backward, Both (Module.Efficiency + System.Size + Annual.Insolation + Total.Installed.Price + Ground.Mounted)	1481.413	10,682.55 Unpruned (System.Size)	18.69	1,597.569 k=1	0.08478697	k-NN
1234	207546.7 Forward, Backward, Both (Module.Efficiency + Annual.Insolation +	1496.923	17,653.25 Unpruned (System.Size)	20.06	2,294.409 k=1	0.06558738	k-NN

	Total.Installed.Price + Ground.Mounted)						
009	93346.25 Forward, Backward, Both (Module.Efficiency + Annual.Insolation + Total.Installed.Price + Ground.Mounted)	1458.023	69,872.8 Default (System.Size)	89.68	8,664.882 k=5	0.1021863	k-NN
169	206363.4 Forward, Backward, Both (Module.Efficiency + System.Size + Annual.Insolation + Total.Installed.Price + Ground.Mounted)	1508.562	86,229.82 Unpruned (System.Size + Total.Installed.Price)	87.62	9,168.279 k=9	0.1151689	k-NN
1609	288548.7 Base (Module.Efficiency + System.Size + DC.Optimizer + Annual.Insolation + Total.Installed.Price + Ground.Mounted)	2187.7	102,714 Default (System.Size)	96.96	10,821.94 k=3	0.190474	k-NN
998	365933.1 Base (Module.Efficiency + System.Size + DC.Optimizer + Annual.Insolation + Total.Installed.Price + Ground.Mounted)	1079.086	91,693.8 Default (System.Size)	113.12	14,555.94 k=1	0.05899157	k-NN
98	271191.2 Base (Module.Efficiency +	1546.82	17,187.03 Unpruned (System.Size)	73.32	11,171.01 k=1	0.06127158	k-NN

	System.Size + DC.Optimizer + Annual.Insolation + Total.Installed.Price + Ground.Mounted)						
78	840,783 Forward, Backward, Both (Module.Efficiency + Annual.Insolation + Total.Installed.Price + Ground.Mounted)	1533.812	22270.16 Unpruned (System.Size)	73.13	19,009.99 k=1	0.07368463	k-NN

Appendix 15: One example of the comparison between the original k-NN model and the rescaled model.

k	Original		Rescaled	
	RMSE	MAPE	RMSE	MAPE
1	13,659.76	0.07341897	0.09913569	5.500836
3	15,079.07	0.09092837	0.1223744	6.540484
5	8,664.882	0.1021863	0.1251667	7.022467
9	21,336.58	0.1495585	0.1513356	7.370016
11	30,902.95	0.1764785	0.1601428	7.353613
25	70,507.45	0.2856143	0.252024	8.336807

Appendix 16: Complete R Code

```
###DATA CLEANING###
full=read.csv("openpv_all.csv") #read full dataset
#read cleaned datasets
clean1=read.csv("TTSX_LBNL_OpenPV_public_file_p1.csv")
clean2=read.csv("TTSX_LBNL_OpenPV_public_file_p2.csv")
#Vertically merge cleaned datasets
library("dplyr")
clean=bind_rows(clean1,clean2)
#Check data types
str(full)
str(clean)
#Change data types
full$date_installed=as.character(full$date_installed)
clean$Installation.Date=as.character(clean$Installation.Date)
full$zipcode=as.factor(full$zipcode)
clean$Zip.Code=as.factor(clean$Zip.Code)
full$installer=as.character(full$installer)
clean$Installer.Name=as.character(clean$Installer.Name)
#Merge full dataset with clean dataset based on 4 primary keys
innerjoin=inner_join(clean,full,by=c("Installation.Date"="date_installed","System.Size"="size_k
w","Zip.Code"="zipcode","Installer.Name"="installer"))
#Check duplicate rows
anyDuplicated(innerjoin)
#Keep only unique rows
innerjoin=distinct(innerjoin)
#Create new dataset only containing useful variables
new_innerjoin=select(innerjoin,Module.Efficiency..1,System.Size,DC.Optimizer,annual_insolati
on,reported_annual_energy_prod,Total.Installed.Price,Ground.Mounted)
str(new_innerjoin)
#Change variable types to factor
new_innerjoin$DC.Optimizer=as.factor(new_innerjoin$DC.Optimizer)
new_innerjoin$Ground.Mounted=as.factor(new_innerjoin$Ground.Mounted)
#Replace -9999 with NA in all columns
new_innerjoin$Module.Efficiency..1[new_innerjoin$Module.Efficiency..1== -9999] <- NA
new_innerjoin$System.Size[new_innerjoin$System.Size== -9999] <- NA
new_innerjoin$DC.Optimizer[new_innerjoin$DC.Optimizer== -9999] <- NA
new_innerjoin$annual_insolation[new_innerjoin$annual_insolation== -9999] <- NA
new_innerjoin$reported_annual_energy_prod[new_innerjoin$reported_annual_energy_prod== -
9999] <- NA
```

```
new_innerjoin$Total.Installed.Price[new_innerjoin$Total.Installed.Price== -9999] <- NA
new_innerjoin$Ground.Mounted[new_innerjoin$Ground.Mounted== -9999] <- NA
apply(new_innerjoin,2,anyNA)
#Remove rows based on NAs in DC Optimizer, Ground Mounted &
Reported_annual_energy_prod
new_innerjoin=new_innerjoin[-which(is.na(new_innerjoin$reported_annual_energy_prod)),]
new_innerjoin=new_innerjoin[-which(is.na(new_innerjoin$DC.Optimizer)),]
new_innerjoin=new_innerjoin[-which(is.na(new_innerjoin$Ground.Mounted)),]
#Replace NAs with means of the column for numerical variables
new_innerjoin$Module.Efficiency..1[is.na(new_innerjoin$Module.Efficiency..1)]=mean(new_in
nerjoin$Module.Efficiency..1,na.rm=TRUE)
new_innerjoin$annual_insolation[is.na(new_innerjoin$annual_insolation)]=mean(new_innerjoin
$annual_insolation,na.rm=TRUE)
new_innerjoin$Total.Installed.Price[is.na(new_innerjoin$Total.Installed.Price)]=mean(new_inne
rjoin$Total.Installed.Price,na.rm=TRUE)
str(new_innerjoin)
#Reorder columns
new_innerjoin=new_innerjoin[c(1,2,3,4,6,7,5)] #Cleaned dataset
#Descriptive Statistics
summary(new_innerjoin)
#Check Outliers
boxplot(new_innerjoin$Module.Efficiency..1,ylab="Module Efficiency",main="Boxplot for
Module Efficiency")
boxplot(new_innerjoin$System.Size,ylab="System Size",main="Boxplot for System Size")
boxplot(new_innerjoin$annual_insolation,ylab="Annual Insolation",main="Boxplot for Annual
Insolation")
boxplot(new_innerjoin$Total.Installed.Price,ylab="Total Installed Price",main="Boxplot for
Total Installed Price")
boxplot(new_innerjoin$reported_annual_energy_prod,ylab="Annual Energy
Production",main="Boxplot for Annual Energy Production")
#Find and remove outliers in "Total.Installed.Price" and "reported_annual_energy_prod"
which.max(new_innerjoin$reported_annual_energy_prod)
which.max(new_innerjoin$Total.Installed.Price)
new_innerjoin=new_innerjoin[-c(3969,4477),]
which.max(new_innerjoin$Total.Installed.Price)
new_innerjoin=new_innerjoin[-4476,]
#Removed 3 rows
summary(new_innerjoin)
#Explanatory Analysis
reg_exp=lm(reported_annual_energy_prod~.,new_innerjoin)
```

```
summary(reg_exp)
#Evaluate each variable
#Histogram to check skewness
hist(new_innerjoin$Module.Efficiency..1)
hist(new_innerjoin$System.Size) #Highly skewed right
hist(new_innerjoin$annual_insolation)
hist(new_innerjoin$Total.Installed.Price) #Highly skewed right
#Test skewness
library(e1071)
skewness(new_innerjoin$Module.Efficiency..1)
skewness(new_innerjoin$System.Size)
skewness(new_innerjoin$annual_insolation)
skewness(new_innerjoin$Total.Installed.Price)
#System.Size and Total.Installed.Price are highly skewed to the right that need to be
transformed
#Choose to use log transformation for System.Size and Total.Installed.Price
new_innerjoin$System.Size=log(new_innerjoin$System.Size)
new_innerjoin$Total.Installed.Price=log(new_innerjoin$Total.Installed.Price)
#Check skewness again
hist(new_innerjoin$System.Size)
skewness(new_innerjoin$System.Size)
hist(new_innerjoin$Total.Installed.Price)
skewness(new_innerjoin$Total.Installed.Price) #Still highly skewed
#Second round of transformation
#Choose to use square root transformation for System.Size
new_innerjoin$System.Size=(new_innerjoin$System.Size)^(1/2)
anyNA(new_innerjoin$System.Size) #Found NAs in transformed results
#Replace NAs with means
new_innerjoin$System.Size[is.na(new_innerjoin$System.Size)]=mean(new_innerjoin$System.Si
ze,na.rm=TRUE)
#Check skewness again
hist(new_innerjoin$System.Size)
skewness(new_innerjoin$System.Size) #Ready
#Choose to use log transformation for Total.Installed.Price
new_innerjoin$Total.Installed.Price=log(new_innerjoin$Total.Installed.Price)
#Check skewness again
hist(new_innerjoin$Total.Installed.Price)
skewness(new_innerjoin$Total.Installed.Price)
#Still highly skewed: We decided to stop transformation here since it is already
significantly better than the original data and more transformation will not significantly
```

improve its normality

```
#Change Variable Names
colnames(new_innerjoin)[1]="Module.Efficiency"
colnames(new_innerjoin)[4]="Annual.Insolation"
colnames(new_innerjoin)[7]="Annual.Energy.Prod"

###Linear Regression Code###
row=nrow(new_innerjoin)
set.seed(666666)
train_index=sample(row,0.8*row,replace=FALSE)
training=new_innerjoin[train_index,]
validation=new_innerjoin[-train_index,]

reg_base=lm(Annual.Energy.Prod~.,training)
summary(reg_base)
PredBase=predict(reg_base,validation)
reg_null=lm(Annual.Energy.Prod~1,training)
reg_forward=step(reg_null,scope =list(upper=reg_base),direction='forward')
PredForward=predict(reg_forward,validation)

reg_backward=step(reg_base,direction='backward')
PredBackward=predict(reg_backward,validation)
reg_both=step(reg_base, direction='both')
PredBoth=predict(reg_both,validation)

library('forecast')
accuracy(PredBase,validation$Annual.Energy.Prod)
accuracy(PredForward,validation$Annual.Energy.Prod)
accuracy(PredBackward,validation$Annual.Energy.Prod)
accuracy(PredBoth,validation$Annual.Energy.Prod)

###Regression Tree Model###
library(MASS) #to make CART
library(tree)
library(forecast)
#Check structure to make sure all categorical variables are read as factor
str(new_innerjoin)
head(new_innerjoin)
model=tree(Annual.Energy.Prod~.,training)
plot(model)
```

```
text(model,pretty=0)
#Check how the model is doing using validation dataset
model_pred=predict(model,validation)
accuracy(model_pred,validation$Annual.Energy.Prod)
#Cross Validation for Pruning the Tree
cv_tree= cv.tree(model)
plot(cv_tree$size,
    cv_tree$dev,
    type = "b",
    xlab= "Tree Size",
    ylab= "MSE")
which.min(cv_tree$dev)
cv_tree$size[1] #Returns the value 6
#prune the tree to size 6
prune_model = prune.tree(model, best = 6)
plot(prune_model)
text(prune_model)
prune_predict = predict(prune_model,validation)
accuracy(prune_predict,validation$Annual.Energy.Prod)

###k-NN Regression###
#Change 2 binary categorical variables to numeric
new_innerjoin$DC.Optimizer=as.character(new_innerjoin$DC.Optimizer)
new_innerjoin$Ground.Mounted=as.character(new_innerjoin$Ground.Mounted)
new_innerjoin$DC.Optimizer=as.numeric(new_innerjoin$DC.Optimizer)
new_innerjoin$Ground.Mounted=as.numeric(new_innerjoin$Ground.Mounted)

#Re-split dataset
row=nrow(new_innerjoin)
set.seed(666666) #set seed
trainIndex=sample(row,0.8*row) #80% of data are training
training=new_innerjoin[trainIndex,]
validation=new_innerjoin[-trainIndex,]


#Find best k
library("class")
accr_results=array(dim=c(1,10)) #Create an array to save the generated list

for(k in 1:10) {
```

```
  pred <- knn.cv(scale(new_innerjoin[,c(1:6)]),new_innerjoin[,7],k)
  accr_results[1,k] <- sum(new_innerjoin[,7]==pred)/nrow(new_innerjoin)
}

round(accr_results,3) #Rounded to 3 decimals
plot(accr_results[1,],xlab="k",ylab="Overall Accuracy",main="Accuracy for Different
k",col="goldenrod1",pch=19,cex=1.5)
#k=1 is the most accurate
#Starting with k=1, also testing other k values

#kNN Regression Models with different k's
Pred_knn_1=FNN::knn.reg(train=training,test=validation,y=training$Annual.Energy.Prod,k=1)
Pred_knn_3=FNN::knn.reg(train=training,test=validation,y=training$Annual.Energy.Prod,k=3)
Pred_knn_5=FNN::knn.reg(train=training,test=validation,y=training$Annual.Energy.Prod,k=5)
Pred_knn_9=FNN::knn.reg(train=training,test=validation,y=training$Annual.Energy.Prod,k=9)
Pred_knn_11=FNN::knn.reg(train=training,test=validation,y=training$Annual.Energy.Prod,k=11)
Pred_knn_25=FNN::knn.reg(train=training,test=validation,y=training$Annual.Energy.Prod,k=25)
#Check accuracy
library("forecast")
accuracy(Pred_knn_1$pred,validation$Annual.Energy.Prod)
accuracy(Pred_knn_3$pred,validation$Annual.Energy.Prod)
accuracy(Pred_knn_5$pred,validation$Annual.Energy.Prod)
accuracy(Pred_knn_9$pred,validation$Annual.Energy.Prod)
accuracy(Pred_knn_11$pred,validation$Annual.Energy.Prod)
accuracy(Pred_knn_25$pred,validation$Annual.Energy.Prod)
#k=1 is the most accurate; however, we also use models with other k values for comparison
with Linear regression and tree models

###k-NN Regression with Rescaling###
#Rescale training and validation separately
library("caret")
normParam=preProcess(training)
norm.training=predict(normParam,training)
norm.validation=predict(normParam,validation)

#kNN Regression on rescaled datasets
Pred_knn_1.=FNN::knn.reg(train=norm.training,test=norm.validation,y=norm.training$Annual.
Energy.Prod,k=1)
Pred_knn_3.=FNN::knn.reg(train=norm.training,test=norm.validation,y=norm.training$Annual.
Energy.Prod,k=3)
```

```
Pred_knn_5.=FNN::knn.reg(train=norm.training,test=norm.validation,y=norm.training$Annual.
Energy.Prod,k=5)
Pred_knn_9.=FNN::knn.reg(train=norm.training,test=norm.validation,y=norm.training$Annual.
Energy.Prod,k=9)
Pred_knn_11.=FNN::knn.reg(train=norm.training,test=norm.validation,y=norm.training$Annual.
Energy.Prod,k=11)
Pred_knn_25.=FNN::knn.reg(train=norm.training,test=norm.validation,y=norm.training$Annual.
Energy.Prod,k=25)
#Check accuracy for rescaled model
accuracy(Pred_knn_1.$pred,norm.validation$Annual.Energy.Prod)
accuracy(Pred_knn_3.$pred,norm.validation$Annual.Energy.Prod)
accuracy(Pred_knn_5.$pred,norm.validation$Annual.Energy.Prod)
accuracy(Pred_knn_9.$pred,norm.validation$Annual.Energy.Prod)
accuracy(Pred_knn_11.$pred,norm.validation$Annual.Energy.Prod)
accuracy(Pred_knn_25.$pred,norm.validation$Annual.Energy.Prod)
```