QTM 2000 Final Project

Analysis of Solar PV Production Efficiency Through R

Team 1:
Raymond Huang, Silvia Huang, Xianle Jin, Shirley Ying

Table of Contents

01 Project Background

02 Dataset Overview

03 Predictive Modeling

04 Validation & Evaluation

05 Conclusion Summary

06 Lessons Learned

Project Background

Solar Energy

Government Incentive

Utility & Oil Industry

Dataset Overview

Original Dataset

Lawrence Berkeley National Laboratory

Variable Descriptions

6 Predictor Variables
1 Target Variable

Exploratory Analysis

Tableau:
Relationships between key predictors and the variable of interest

Preprocessing

- Merge based on 4 common keys
- Duplicates Elimination
- Variable Transformations

Original Dataset

Raw Dataset

- 1,020,813 rows
- 81 variables

The Open PV Project

Sourced primarily from state agencies and utilities that administer PV incentive programs

Public Data File

- 1,094,909 rows
- 52 variables

Vertically Merged

Includes only grid-connected residential and non-residential PV systems

Team Dataset

- 8,718 rows
- 7 variables

Cleaned

Contains only variables that are relevant to the project

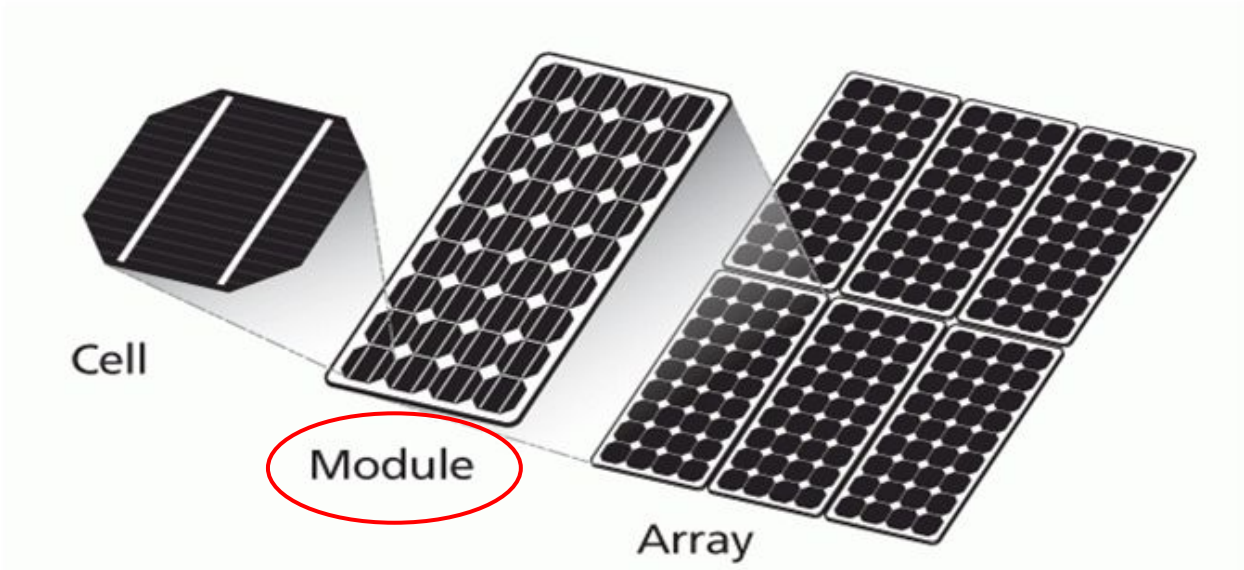
Data Dictionary

		Units	Descriptions
1	**Module Efficiency**	**%**	The rate of solar panels converting solar energy into electrical energy.
2	**System Size**	**kW**	The total rated direct-current (DC) output of the module at standard test conditions.
3	**DC Optimizer**	**-**	DC Optimizer is a converter technology developed to maximize the energy harvest from solar photovoltaic.
4	**Annual Insolation**	**kWh/m^2/ day**	The daily average insolation for the Earth is around 6 kWh/m^2. The output of a PV panel partly depends on the angle of the sun relative to the panel.
5	**Total Installed Price**	**$**	The total installed price for the system, prior to receipt of any incentives.
6	**Ground Mounted**	**-**	If the system is ground-mounted (1) or rooftop (0).
7	**Annual Energy Production**	**kWh/yr**	The annual energy production of the PV system. This measurement is represented as kWh per square meter of panel surface.

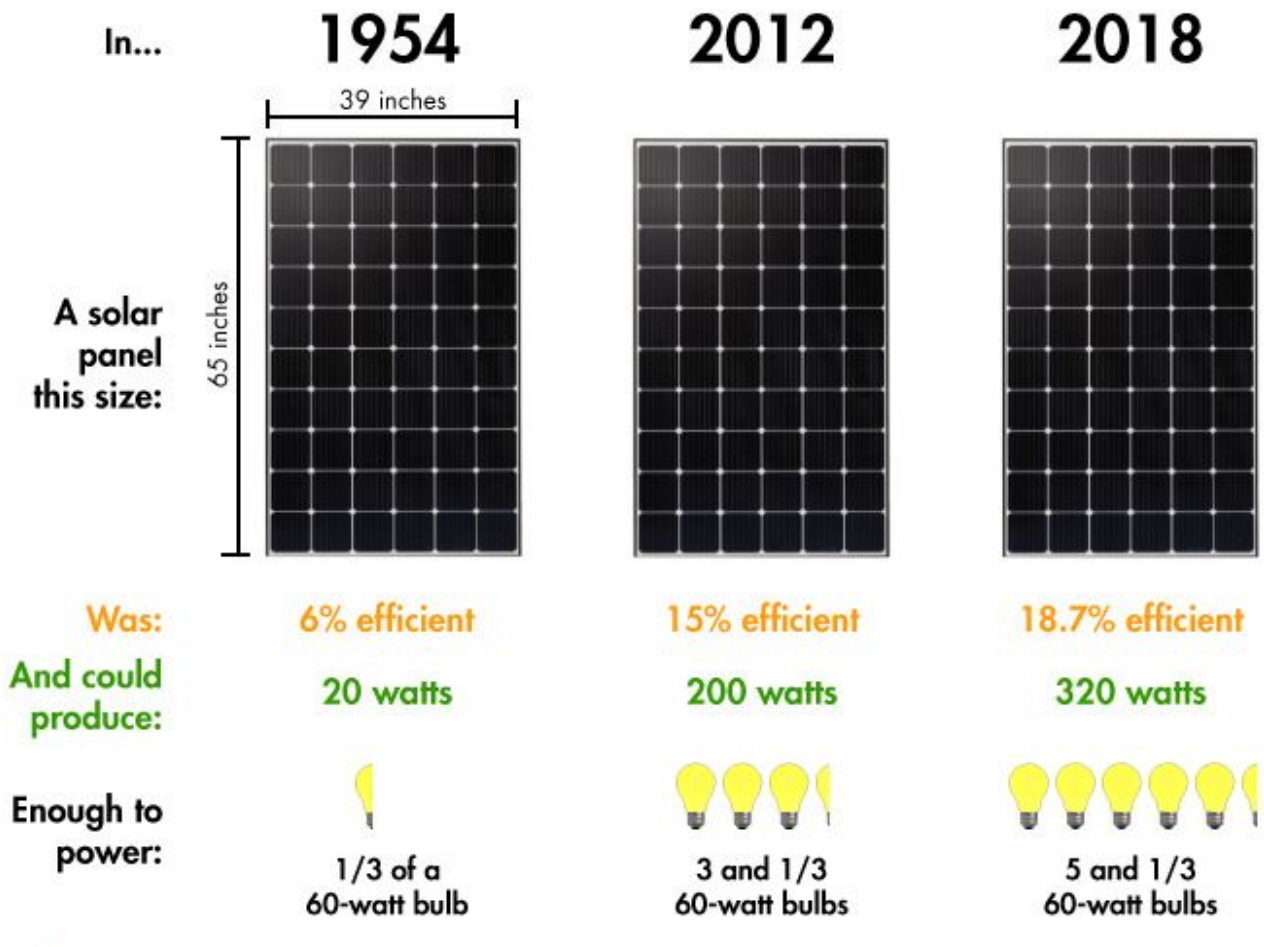
Solar Module

Solar Cell to Solar Farm

Many cells make a module → Many modules make an array → Many arrays make a solar farm

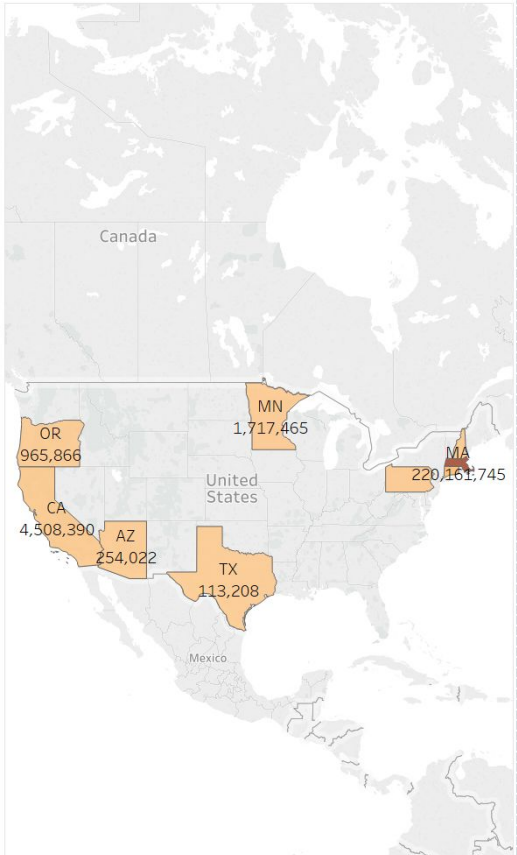


Module Efficiency

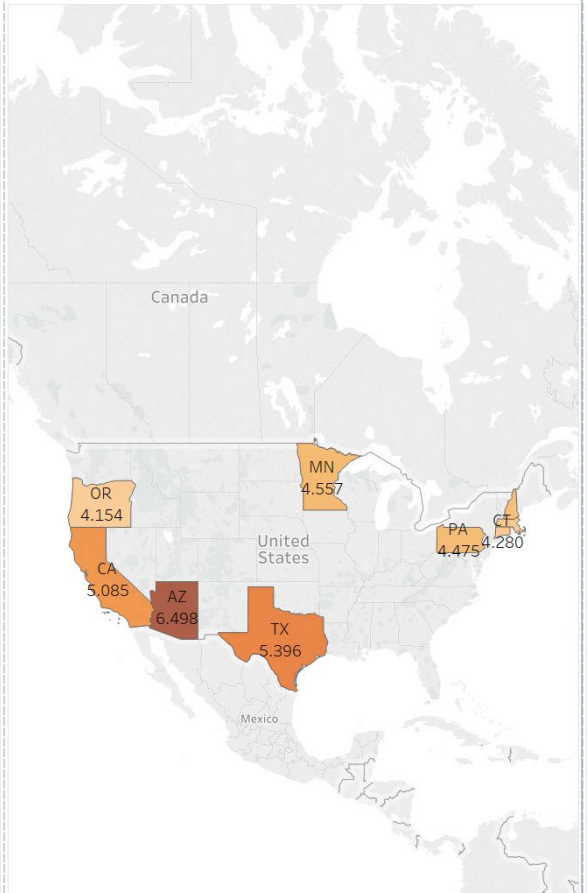


Exploratory Analysis Tableau

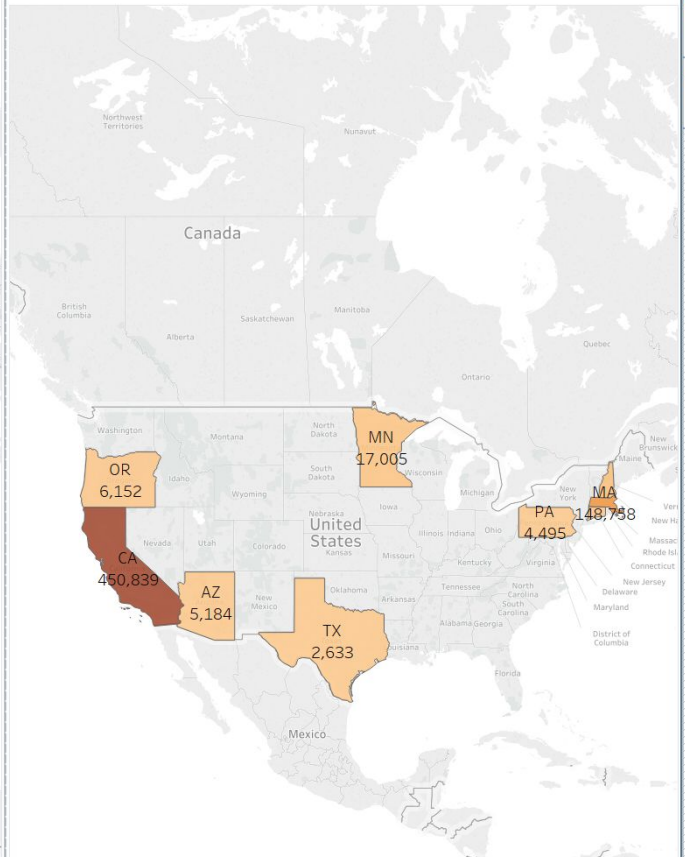
Sum of Reported Annual Energy Produciton Over States



Average Annual Insolation Rate Over States



Average Reported Annual Energy Produciton Over States

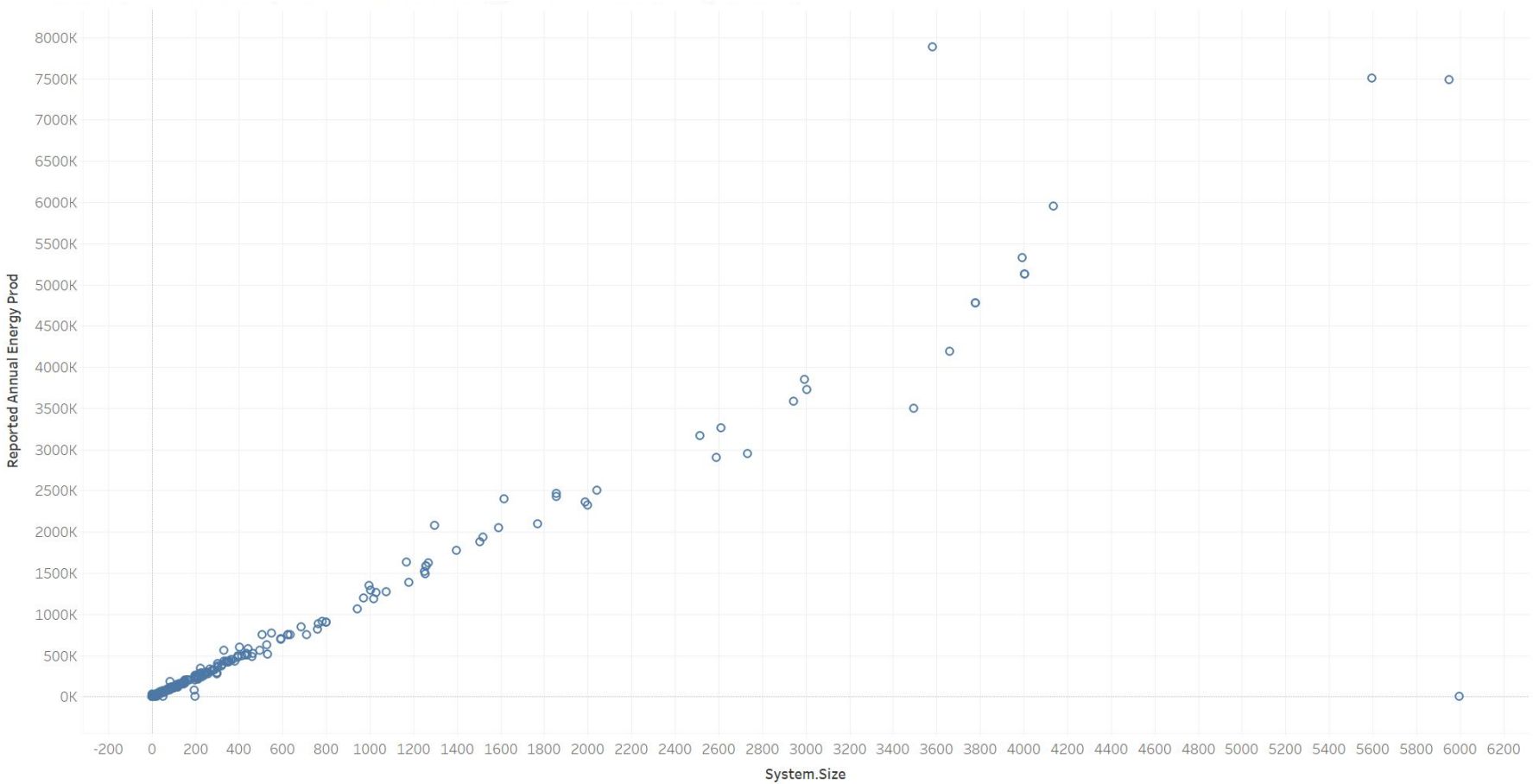


Avg. Annual Inso..
4.154 6.498

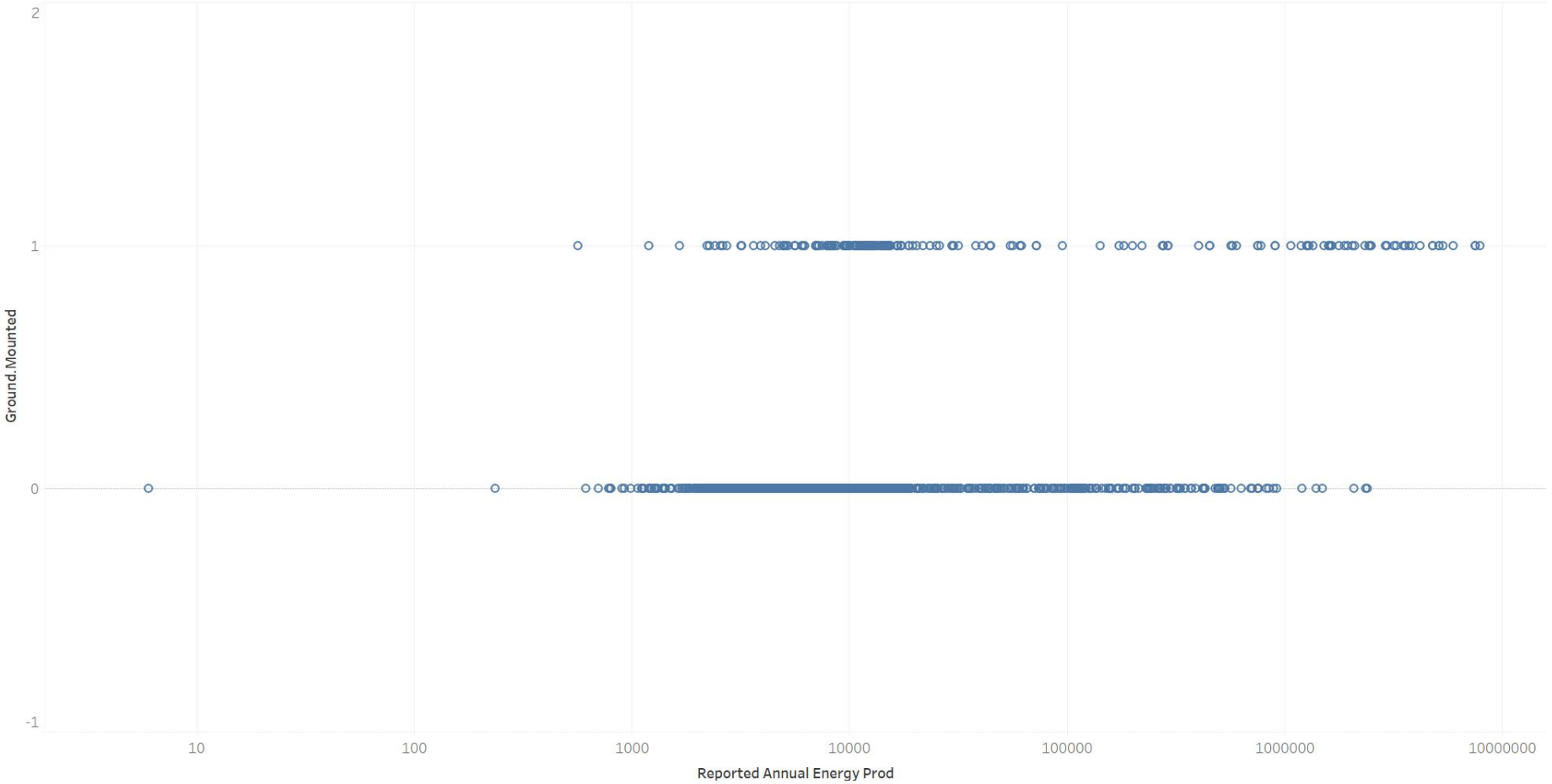
Avg. Reported A..
2,633 450,839

Reported Annua..
113,208 220M

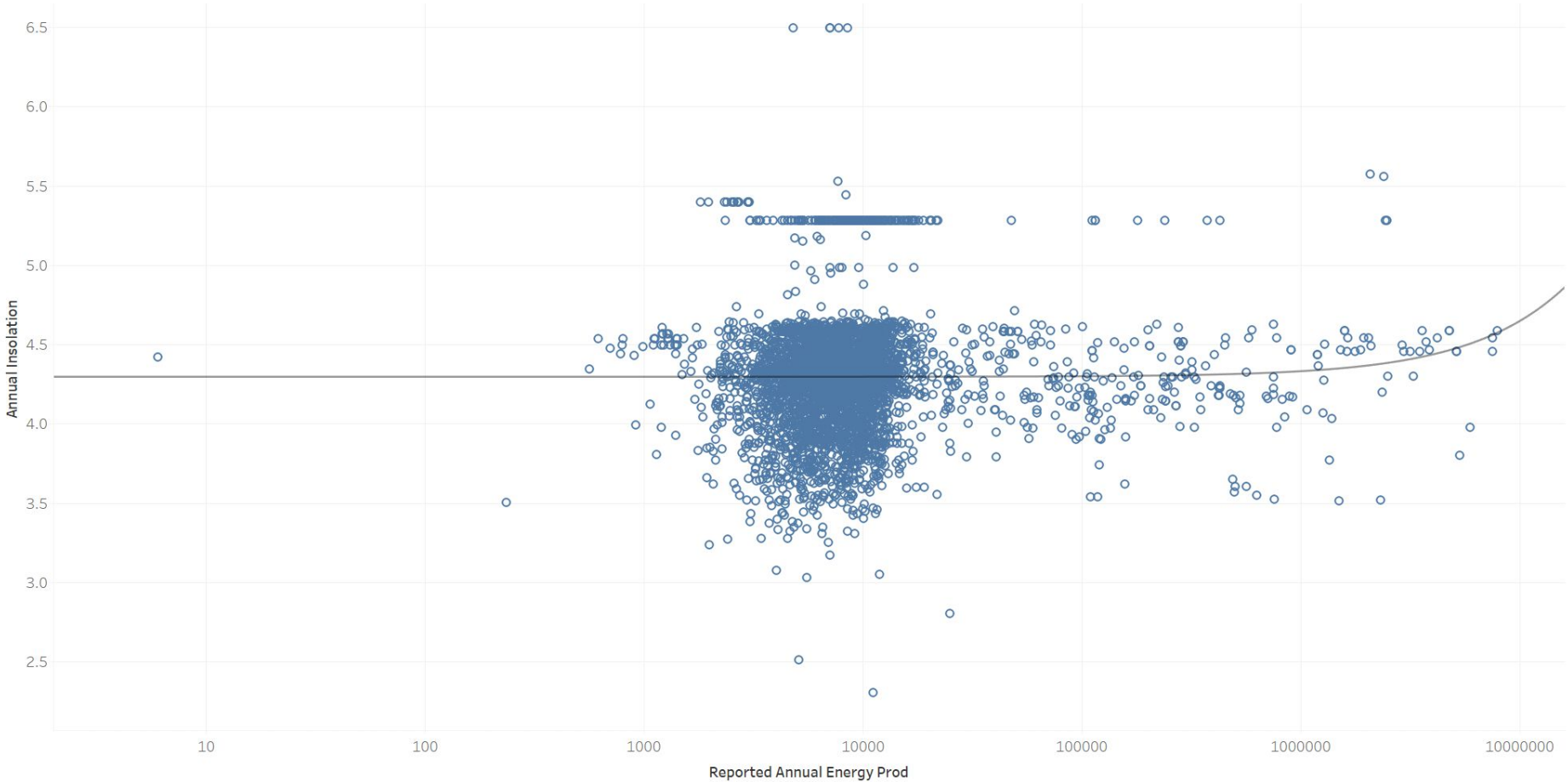
Annual Energy Production vs. System Size



Ground Mounted vs. Annual Energy Production



Annual Insolation vs. Annual Energy Production



Data Pre-processing

Primary Key Selection

- Installation Date
- System Size
- Zip Code
- Installer Name

Aggregation

dplyr: inner_join

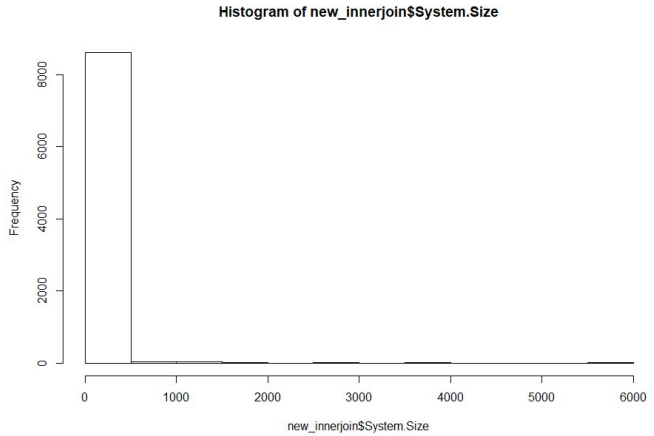
80,000 rows
&
129 variables

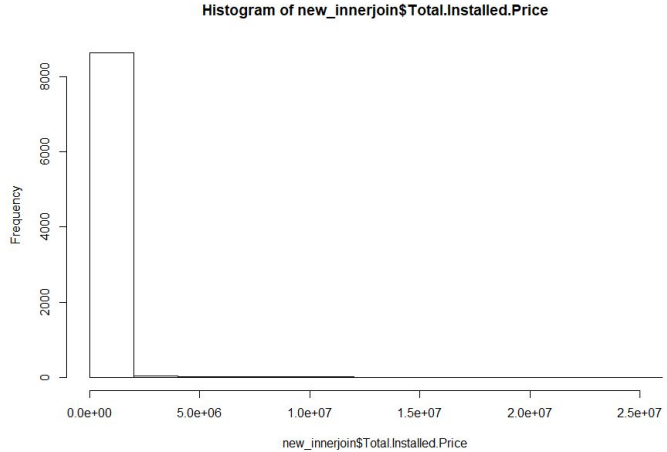
Elimination

Duplicates Exclusion
+
N/A Exclusion

Numeric Categorical

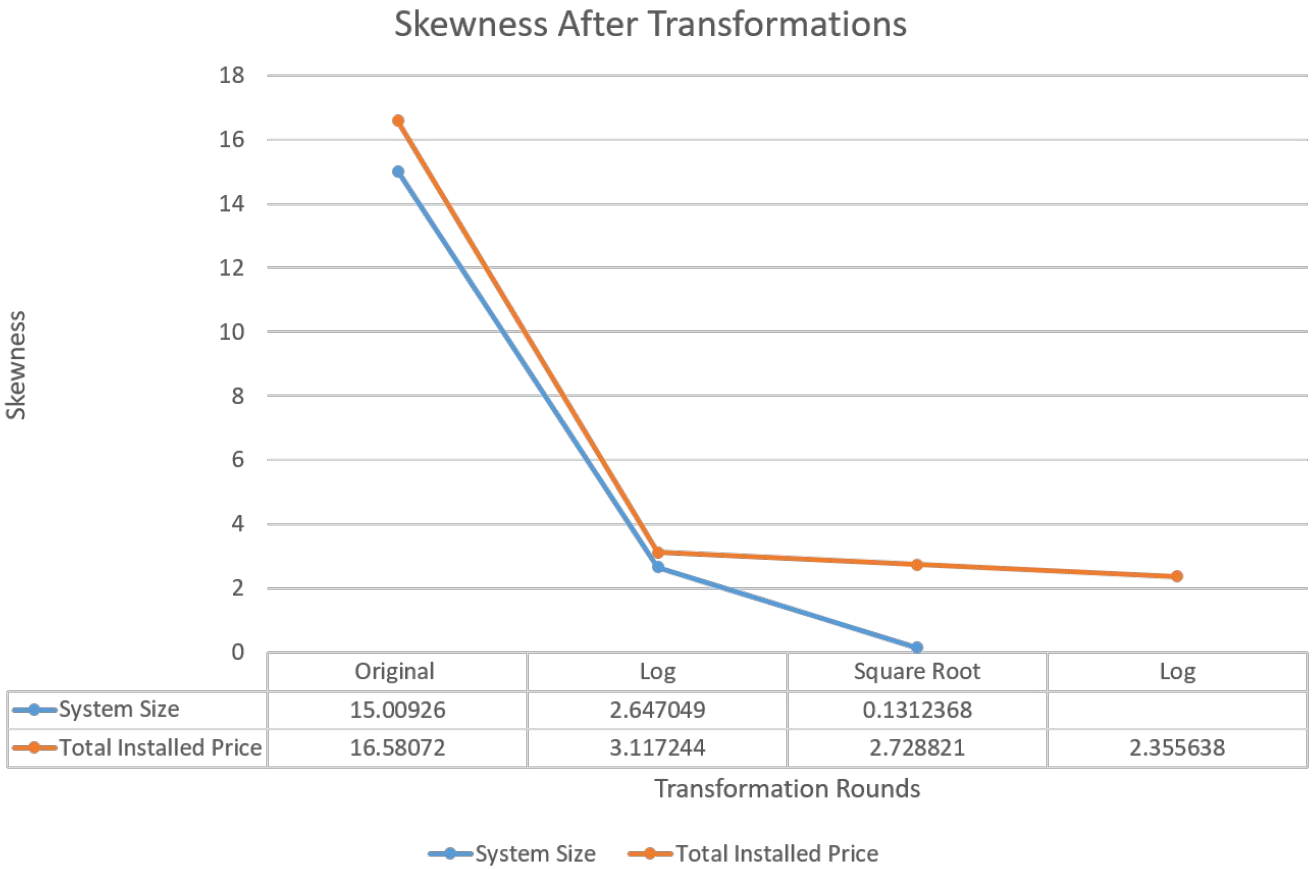
Data Transformations





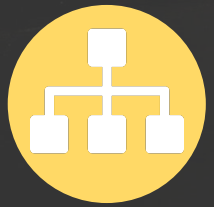
Transformations	Skewness	
	System Size	Total Installed Price
Original Data	15.01	16.58
Cube Root	6.63	6.86
Log	**2.65**	**3.12**
Square Root	8.72	9.05

System Size & Total Installed Price



	Original	Log	Square Root	Log
System Size	15.00926	2.647049	0.1312368	
Total Installed Price	16.58072	3.117244	2.728821	2.355638

Predictive Modeling



Linear Regression

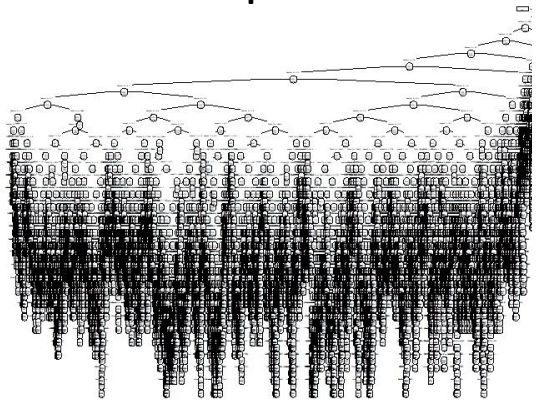
Seed	RMSE (Base)		RMSE (Forward)	RMSE (Backward)	RMSE (Stepwise)
009	262,965.6	>	262,777.6	262,777.6	262,777.6

Annual.Energy.Prod
=
59,562*Annual.Insolation + 1,978,410*Total.Installed.Price + 534,093*Ground.Mounted - 4,862,615

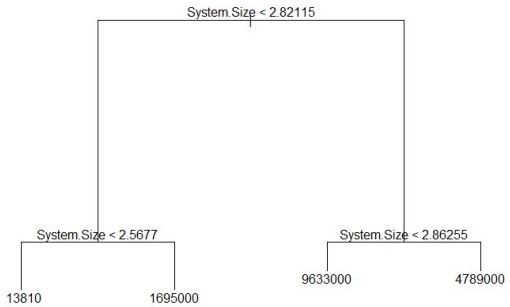
Regression Tree

Packages: MASS, tree

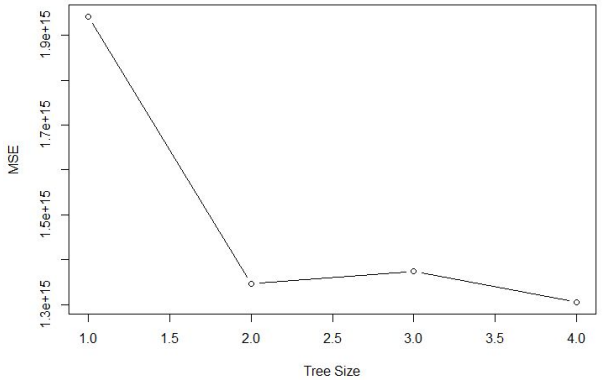
Unpruned



Pruned



Choose Optimal Number of Splits



Prune.tree (model, best = 4)

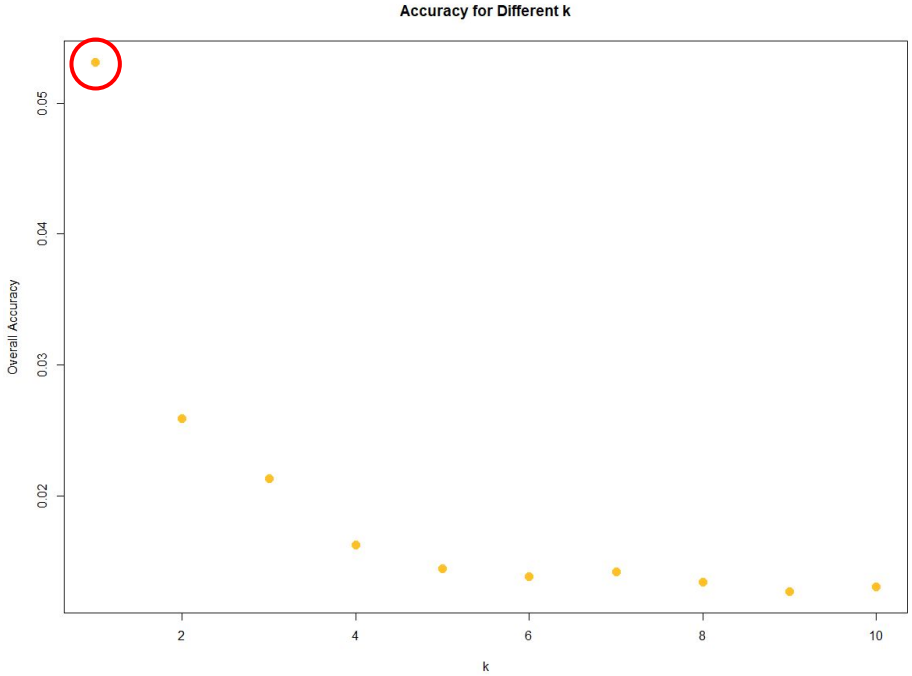
which.min(cv_tree$dev)

Seed	ME	RMSE	MAE	MPE	MAPE
009	-1,700.99	225,818.30	27,813.87	-190.94	200.40

k-NN Regression

class: knn.cv

Find k with highest accuracy



FNN: knn.reg
forecast: accuracy

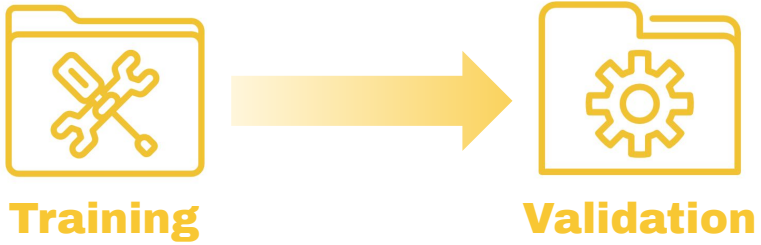
k Values	RMSE
1	11,404.04
3	13,381.71
5	9,097.48
9	29,358.93
11	41,843.69
25	80,899.74

Validation & Evaluation

Model	RMSE	%
Linear (forward)	262,777.6	0
k-NN (k=5)	9,097.48	100
CART (unpruned)	225,818.30	0

Validation & Evaluation

caret: preProcess



	k	1	3	5	9	11	25
RMSE	**Original**	11,404.04	13,381.71	9,097.48	29,358.93	41,843.69	80,899.74
	Rescaled	0.057	0.083	0.082	0.119	0.120	0.184
MAPE	**Original**	0.128	0.164	0.225	0.386	0.478	0.811
	Rescaled	5.041	6.697	6.847	7.518	7.335	8.145

Conclusion Summary

Assist Installation Decision-Making

Customers can use solar energy annual production output and compare with its current utility usage

Modify Specific Installation Settings

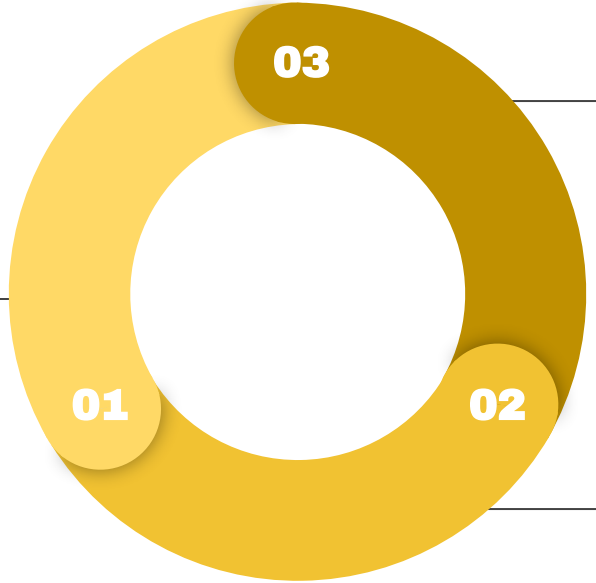
Variables such as system size and module efficiency have the biggest influence on annual PV production

Solar Panel Research Development

Need more policy assistance on solar energy research to unify data collection practice

Lessons Learned

Creative Problem-Solving Approach



Teamwork

Perseverance

Lessons Learned

“If you torture the data long enough, it will confess.”

--- Ronald H. Coase


THANK
YOU
Questions?

APPENDIX

Appendix A Data Cleaning Code 1

```
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_kw","Zip.Code"="zipcode","Installer.Name
"="installer"))
```

Appendix A Data Cleaning Code 2

```
#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_insolation,reported_annual_energy_prod,Total.I
nstalled.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)
```

Appendix A Data Cleaning Code 3

```
#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$Module.Efficiency..1)]=mean(new_innerjoin$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$Total.Installed.Price)]=mean(new_innerjoin$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
#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)
```

Appendix A Data Cleaning Code 4

```
#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.Size,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)
```

Appendix A Data Cleaning Code 5

#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"
```

Appendix B 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)
```

Appendix C Regression Tree Code

```
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)
```

Appendix D k-NN Regression Code 1

```
#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)
#Split dataset
row=nrow(new_innerjoin)
set.seed(666666) #set seed of 666666
trainIndex=sample(row,0.8*row) #80% of data are training
training=new_innerjoin[trainIndex,]
validation=new_innerjoin[-trainIndex,]
#Did not scale data first
#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)
```

Appendix D k-NN Regression Code 2

Choose k=1

#But since k=1 usually overfits, we decided to test different k's

#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 has the lowest error (RMSE); however, since it is usually overfitting, we also use models with other k's for comparison with Linear regression and Tree models

Appendix E Results on Validation 1

Seed	RMSE (Linear Regression)	RMSE (Prediction Tree)	RMSE (kNN)	Best Model
666666	836937.5 Forward, Backward, Both Variable: Module.Efficiency + Annual.Insolation + Total.Installed.Price + Ground.Mounted	738897.8 Unpruned Variables: System.Size + Annual.Insolation	649398 k=1 655478.4 k=3	kNN
12345	834820.1 Forward, Backward, Both Variable: Module.Efficiency + Annual.Insolation + Total.Installed.Price + Ground.Mounted	782594.2 Unpruned Variable: System.Size	649307.4 k=1 655321.4 k=3	kNN
250	216946.0 Forward, Backward, Both Variables: Annual.Insolation + Total.Installed.Price + Ground.Mounted	69449.4 Unpruned Variable: System.Size	13580.62 k=5	kNN
78	840783 Forward, Backward, Both Variables:Module.Efficiency + System.Size + DC.Optimizer + Annual.Insolation + Total.Installed.Price + Ground.Mounted	778329.3 Unpruned Variables: System.Size	649457.2 k=1 655550 k=3	kNN

Appendix E Results on Validation 2

Seed	RMSE (Linear Regression)	RMSE (Prediction Tree)	RMSE (kNN)	Best Model
1609	219722.4 Base Variables: Module.Efficiency + System.Size + DC.Optimizer + Annual.Insolation + Total.Installed.Price + Ground.Mounted	160425.6 Pruned Variables: System.Size	9611.701 k=5	kNN
998	220595.5 Forward, Backward, Both Variables: Annual.Insolation + Total.Installed.Price + Ground.Mounted	145943.6 Unpruned Variables: System.Size	1379.926 k=1 5558.41 k=5	kNN
98	198103.6 Base Variables: Module.Efficiency + System.Size + DC.Optimizer + Annual.Insolation + Total.Installed.Price + Ground.Mounted	81700.1 Unpruned Variables: System.Size	3983.245 k=1 4350.727 k=3	kNN

Appendix E Results on Validation 3

Seed	RMSE (Linear Regression)	RMSE (Prediction Tree)	RMSE (kNN)	Best Model
1234	251231.7 Forward, Backward, Both Variables: Annual.Insolation + Total.Installed.Price + Ground.Mounted	128319.5 Unpruned Variables: System.Size	9797.461 k=1 11100.49 k=3	kNN
009	262777.6 Forward, Backward, Both Variables: Annual.Insolation + Total.Installed.Price + Ground.Mounted	225818.3 Unpruned Variable: System.Size	9097.478 k=5	kNN
169	231027.3 Forward, Backward, Both Variables: Annual.Insolation + Total.Installed.Price + Ground.Mounted	288876.5 Unpruned Variables: System.Size + Total.Installed.Price	642.5359 k=1 7019.865 k=3	kNN

Appendix F k-NN Rescaling Code

```
#Scale training and validation
library("caret")
normParam=preProcess(training)
norm.training=predict(normParam,training)
norm.validation=predict(normParam,validation)

#kNN Regression on rescaled dataset
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)
```

Appendix G Descriptive Statistics

```
> summary(new_innerjoin)
 Module.Efficiency  System.Size      DC.Optimizer Annual.Insolation Total.Installed.Price Ground.Mounted Annual.Energy.Prod
 Min.   :0.0620    Min.   :0.3087   -9999:   0   Min.   :1.335     Min.   :2.032         -9999:   0     Min.   :       6
 1st Qu.:0.1607    1st Qu.:1.2401   0    :5802   1st Qu.:4.112     1st Qu.:2.302         0    :8424     1st Qu.:    4753
 Median :0.1614    Median :1.3681   1    :2916   Median :4.300     Median :2.336         1    : 294     Median :    6712
 Mean   :0.1614    Mean   :1.3519               Mean   :4.268     Mean   :2.342                       Mean   :   43079
 3rd Qu.:0.1614    3rd Qu.:1.4847               3rd Qu.:4.474     3rd Qu.:2.366                       3rd Qu.:    9506
 Max.   :0.2116    Max.   :2.9494               Max.   :6.498     Max.   :2.835                       Max.   :35000000
```