Assessing the accuracy of a variety of classifiers on solar system coverage in the United States of America

This study looked at the classification accuracy of 4 different classifiers in predicting if a DeepSolar database observation has a low or high number of solar photovoltaic panels. Of the four methods studied (Binary Logistic Regression, Supper Vector Machines, Boosting, and Random Forest), the random forest classifier had the highest accuracy in classifying the validation dataset. The Random Forest classifier then produced a classification with an accuracy of 90.1%. The variables most important to the random forest classification were total area, population density, population, and average household income.

Introduction

The DeepSolar database is a database created from analysed satellite imagery that identifies solar photovoltaic (PV) panels in the United States (Yu et al. 2018). Each satellite image is analysed for the number of photovoltaic panels present, and a number of environmental and social factors are measured for the area the group of panels was found in. As climate change continues, governments will look to solar photovoltaic panels to meet energy demands without contributing to exhaust pollution (Ebinger, Vergara, and World 2011).

By understanding how geographical and social factors impact the use of solar panels, alternative forms of energy can be implemented in areas not geographically suited for solar panel use (Armaroli and Balzani 2011). If certain social features are found to impact the level of solar panel coverage, such as education level, programs could be launched to increase their usage by targeting these features in particular to maximise effectiveness. For this to take place however, it is crucial that there is an initial understanding in what factors effect solar panel use, and what is the best way to analyse this data.

The solar database used in this analysis is composed of 20736 observations on 81 variables. Each observation signifies a satellite image that DeepSolar has detected as having solar photovoltaic panels present. The response variable in this analysis is the number of solar panels present in each observation, a binary variable taking the two values 'Low' and 'High'. Low indicates that ≤ 10 solar panels were detected, High indicates that >10 solar panels were detected. The 80 predictor variables range from economic factors such as electricity price, to geographical such as humidity.

The aim of this study is firstly to determine which of 4 methods of classification provides the highest level of accuracy in predicting the level of solar panel coverage. The 4 methods analysed are:

- Binary Logistic Regression
- Supper Vector Machines
- Boosting

- Random Forest

In doing so, this study hopes to aid the accuracy and speed at which deeper analysis on the factors influencing solar panel use can be carried out. As well as providing the reader with the merits of the classification method discussed such as the speed taken to preform this classification

Secondly, this study aims to discover the variables within the DeepSolar dataset that are of the most importance to a observation being classified as having either a High or Low level of solar panel coverage. By answering this question, further studies can be carried out that look into the impact of the highlighted variables more deeply to assess what can be maximised and minimised to produce a high level of solar system coverage.

Exploratory Data Analysis

A number of data visualisation methods were carried out to assess potentially significant relationships between both the response variable and independent variables, as well as between the independent variables. It was determined that understanding how a variety of variables interact and respond to changes in solar power system coverage would provide useful insight during the data preprocessing and model creation steps.

Boxplot diagrams were created to show the mean, upper and lower quantiles of a variety of different independent variables, chosen by hand, and how they change based on solar power system coverage. Variables that showed a considerable difference in mean at low and high solar system coverage were considered to possess a high level of impact on the classification. Special attention was given to not remove these variables from the model creation unless evidence supporting their removal was discovered.

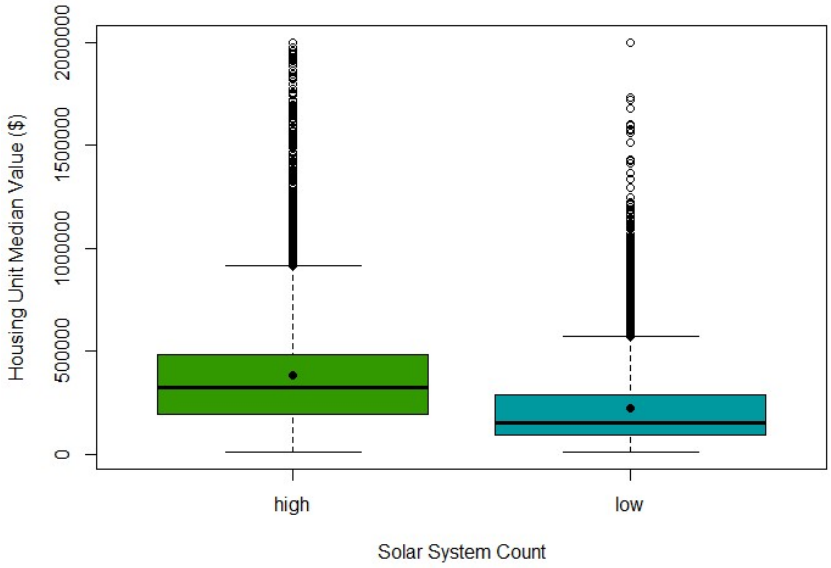

Figure 1. A boxplot showing the variation in Housing Unit median values ($) across high and low solar system count observations. Centre dot signifies the mean value

Figure 1 shows that a high solar system count level is associated with a higher housing unit value. The black line in the middle of the boxplots which signifies the median value can be seen to be higher on observations with a high solar system count than observations with a low solar system count .The same can be said for the mean. From this plot the belief that a high housing unit median value contributes to an observation being classified as having a high number of solar systems was established. The mean for both low and high classes was also observed to be above the median value, which implies a positively skewed distribution.

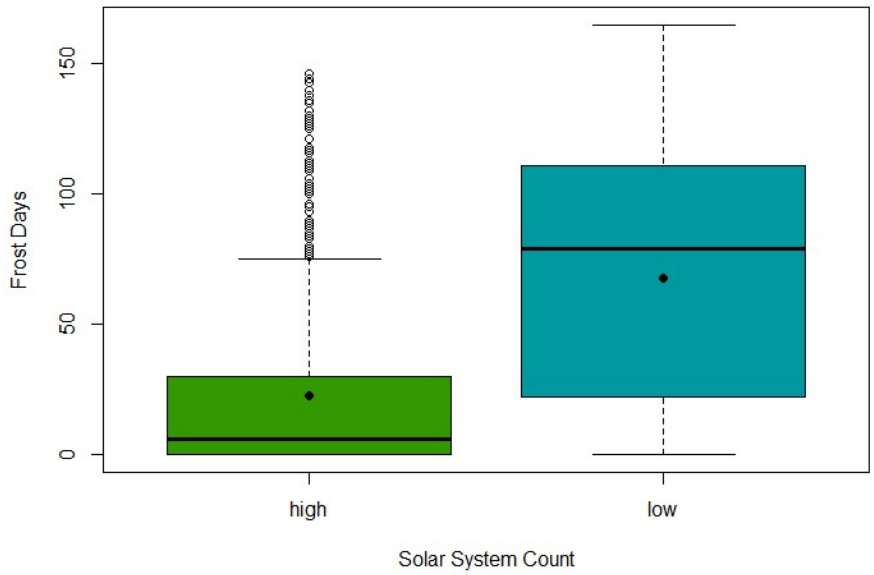

Figure 2. A boxplot showing the variation in Frost Days across high and low solar system count observations. Centre dot signifies the mean value

Figure 2 shows that a low solar system count level is associated with a higher housing unit value. This is the opposite of the relationship shown in figure 1. The median and median indicators for the number of frost days in high solar system count observations is much lower than observations with a low solar system count. From this plot it was believed that a high number of frost days in a year is associated with a low solar system count. Additionally, it was shown that the interquartile range of frost day values is much larger in low solar system count observations.

Correlations between predictor variables were assessed to look for multicollinearity issues (Figure 3). Multicollinearity is when two or more independent variables are highly correlated with one another, one variable can be linearly predicted from the others with a high degree of accuracy. A high correlation between the variables in the solar dataset would imply that they are predicting the same thing. If the classification methods are employed with multicollinearity among the independent variables, the coefficients received will be unstable and inaccurate. Overall, it would make the interpretation of the classifier, and the determining of which classifier is optimal, more difficult (Shen and Gao 2008).

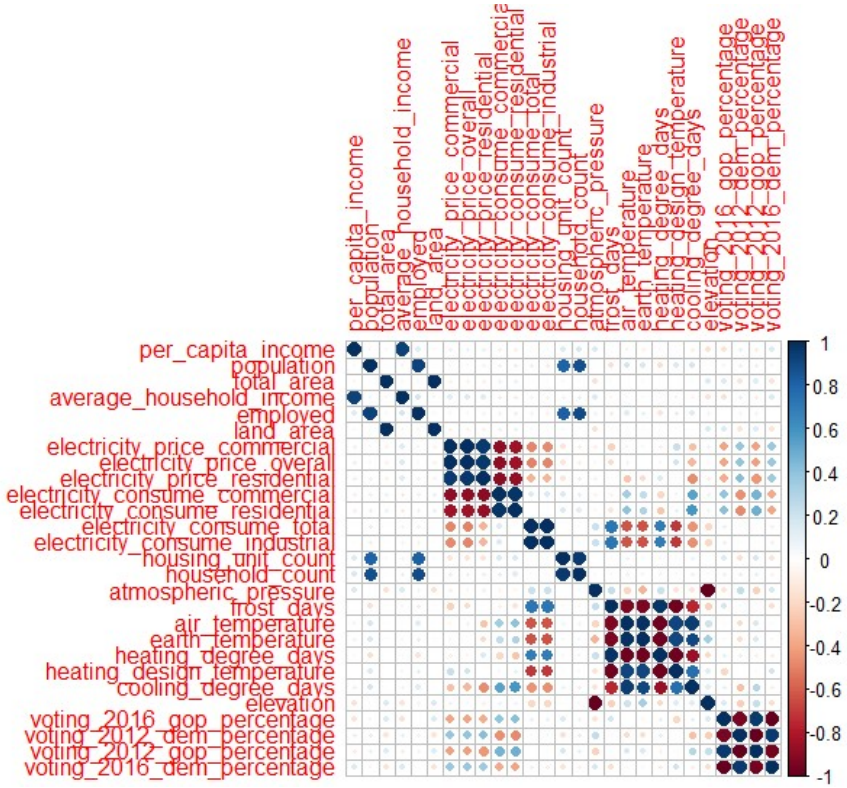

Figure 3. A graph of the highly correlated variables and their relationships. The darkness of the colour indicates the strength of the correlation. Note each variable above is strongly correlated with at least one other variable.

Data Pre-processing

From the exploratory data analysis carried out, it was discovered that a number predictor variables were highly correlated with one another Figure 3.. To prevent multicollinearity issues, when a response variable had a correlation greater than 0.9 or less than -0.9 with another variable, the variable that had the weakest correlation to the dependent variable – solar_system_count, was removed. The thinking behind this decision was that one of the two highly correlated variables must be removed, and rather than arbitrarily choosing one of them to remove, the one that had the weakest relationship with the response variable should be removed as it provides the least amount of information to the classifier. This method was chosen over dropping the variable with the assumed lesser impact due to theory and prior knowledge (Graham 2003) as there was limited prior knowledge on the theoretical and past impact of each variable.

Although not highly correlated on a 1 to 1 basis, certain variables were shown to have singularity issues when considered together in the classifier. These were the variables relating to electricity consumption and electricity price. Broken up into commercial, residential and industrial, they still add up to the total and overall values. These singularities were removed and instead only the total consumption and overall price variables were used in the model rather than those broken down by sector. Although this caused a loss of information, it was deemed the better choice than using each individual sectors value rather than the total, as certain sector variables had already been removed because of high correlation, so including all 3 sectors was no longer possible.

A list of all removed variables is available in the appendix.

Additionally, all numerical variables were standardised so that each had a mean value of 0 and a standard deviation of 1, this prevents large value variables from dominating the classification.

The data set was randomly split into training, validation, and test sets with a ratio of 60-20-20, meaning that the training set had 12442 observations, and the validation and test sets both had 4147 observations. The motivation behind creating these sets is so that each model is created using the same data so that fair comparisons can be made, and additionally they are all assessed using the same validation data that is distinct from the data used to create the model as this provides an unbiased evaluation of the classification.

Classifier creation and performance on validation set

Binomial Logistic regression

Binomial logistic regression is a linear model that predicts the probability of a certain observation falling into a alternative class over the default (In this papers case, the probability of being a low solar system class rather than the default high solar system class). It is assigned to this alternative class when the value obtained is greater than tau, which is a fixed value between 0 and 1. (Couronné, Probst, and Boulesteix 2018)

In this study the binomial logistic regression was created using the training dataset and all standardised variables except those removed due to correlations and singularities (as discussed in section 4.2).

As the two classes, high solar system count and low solar system count were relatively balanced, composing 10,900 and 9,836 observations respectively, a tau value of 0.5 was selected. If one of the two classes was rarer than the other, for example only 10% of the observations having a low solar system count, a tau value other than 0.5 would have been gained through the use of a Receiver operating characteristic (ROC) curve.

Runtime was measured by getting the system time at before and after the classifier was run.

Support Vector Machines

Support Vector Machines is a supervised classification method based on determining the best plane of separation of the two classes and can use non-linear patterns.

In this study the Support Vector Machine was created using the training dataset and all standardised variables except those removed due to correlations and singularities (as discussed above). A Radial Basis kernel "Gaussian" was used.

The classifiers performance was measured by summing the correctly classified validation set observations and dividing it by the total number of observations to receive a percentage score.

Runtime was measured by getting the system time at before and after the classifier was run.

Boosting

Boosting is a supervised classification method which uses a collection of weak classifiers to produce a strong classifier. A first model is built, then a second model based on correcting the errors from the first model. This process is repeated for the instructed number of times.

In this study the Boosting classifier was created using the training dataset and all standardised variables except those removed due to correlations and singularities (as discussed above). The Breiman coefficient learning was used and no bootstrapping was applied. This model was repeated for 100 iterations.

The classifiers performance was measured by summing the correctly classified validation set observations and dividing it by the total number of observations to receive a percentage score.

Runtime was measured by getting the system time at before and after the classifier was run.

Random Forest

Random forest is a supervised classification method which consists of aggregating a large number of decision trees(Couronné, Probst, and Boulesteix 2018). Only a random subset of the predictor variables are used at each split of the classification tree fitting step, which allows for assessment over which variables are most important in terms of classification accuracy.

In this study the random forest was created using the training dataset and all standardised variables except those removed due to correlations and singularities (as discussed above).

Classifier performance was measured by summing the correctly classified validation set observations and dividing it by the total number of observations to receive a percentage score.

Runtime was measured by getting the system time at before and after the classifier was run.

Final Classifier

The final best classifier was determined by examining which classifier received the highest correct classification rate on the validation set. The best preforming model was then used to predict the observations in the test dataset, and the results of this classification was analysed as the performance of the final model.

Results and Discussion

Classifier Performance on validation data

Binomial Logistic regression

Table 1. The correctly and incorrectly classified observations by the binomial logistic classifier.

Observed	**Predicted**	
	High Solar System Count	**Low Solar System Count**
High Solar System Count	1996	155
Low Solar System Count	361	1635

From Table 1 it was observed that the binomial logistic classifier correctly classified 92.8% of the observations that had a High Solar System Count, and 81.9% of the observations that had a Low Solar System Count.
Overall, the model had a classification accuracy of **87.6%** on the validation dataset. This is a good accuracy score, particularly when you consider the efficient processor power requirements for creating the model, and the easy to interpret coefficients that are output from such a classifier, two of the primary merits of a binomial logistic regression. The classifier took 0.52 seconds to run.

It is clear that the model had particular difficulty in the classification of low solar system count observations, mistaking them for high solar system count observations more frequently than it mistook high solar system count observations for low system counts.

Support Vector Machines

Table 2. The correctly and incorrectly classified observations by the Support Vector Machines Classifier.

Observed	**Predicted**	
	High Solar System Count	**Low Solar System Count**
High Solar System Count	1942	209
Low Solar System Count	227	1769

From Table 2 it was observed that the support vector machines classifier correctly classified 90.3% of the observations that had a High Solar System Count, and 88.6% of the observations that had a Low Solar System Count.
Overall, the model had a classification accuracy of **89.4%** on the validation dataset. Although slightly weaker at high solar system count classification than the binomial regression classifier (Table 1), the support vector machines classifier was superior at low solar system count classification, and overall had a less unbalanced successful classification rate between low and high counts, leading to a higher accuracy.

Although the support vector machine classifier had a superior classification rate, it is worth noting that the processing requirement is much higher than binomial classifier, taking 14.57 seconds to run.

Boosting

Table 3. The correctly and incorrectly classified observations by the Boosting Classifier.

Observed	**Predicted**	
	High Solar System Count	**Low Solar System Count**
High Solar System Count	1954	197
Low Solar System Count	235	1761

From Table 3 it was observed that the support vector machines classifier correctly classified 90.8% of the observations that had a High Solar System Count, and 88.2% of the observations that had a Low Solar System Count.
Overall, the model had a classification accuracy of **89.6%** on the validation dataset. This classifier achieved values roughly equal to the performance of the Support Vector Machine classifier (Table 2), but the boosting classifier had a significantly longer runtime to complete, at 137.1 seconds. It can be seen that boosting without bootstrapping is an effective method of classification, but has a strong processor requirement to run, achieving results comparable to support vector machine classifiers, which has a much lower processor requirement.

Random Forest

Table 4. The correctly and incorrectly classified observations by the Random Forest Classifier.

Observed	**Predicted**	
	High Solar System Count	**Low Solar System Count**
High Solar System Count	1976	175
Low Solar System Count	225	1771

From Table 4 it was observed that the random forest classifier correctly classified 91.9% of the observations that had a High Solar System Count, and 88.7% of the observations that had a Low Solar System Count.
Overall, the model had a classification accuracy of **90.3%** on the validation dataset. This is the highest overall accuracy from any of the 4 classifiers examined, and was achieved in a runtime of 70.8seconds. The random forest classifier also provides a list of the variables that were the most important to the classification, a useful feature for interpretation

Summary

Table 5. Overall summary of the classifier performance on the validation set. Best values are highlighted in green.

	Binomial Logistic	Support Vector Machine	Boosting	Random Forest
% High Correct	92.8	90.3	90.8	91.9
% Low Correct	81.9	88.6	88.2	88.7
% Overall Correct	87.6	89.4	89.6	90.3
Runtime (seconds)	0.52	14.57	137.1	70.8

The Random Forest classifier was chosen as the best due to possessing the highest overall classification rate. Interestingly, it was not the best at classifying the high solar system cover correctly. It was the simplest mode, the binomial logistic classifier that had the best rate, at 92.8% (Table 5). The overall tradeoff between classifier accuracy and processor requirements is visible in Table 5, with the less accurate classifiers running faster than the more accurate classifiers. The exception is the boosting classifier, which had the longest and most intensive processor requirements by a large margin, yet obtained results comparable to those received by the support vector machine classifier.

Best Classifier

The random forest classifier, selected on the basis of its accuracy on the validation set, was now used to classify the values of the test dataset. The following values were received:

Table 6. Performance of the random forest classifier on the test dataset

Observed	**Predicted**	
	High Solar System Count	**Low Solar System Count**
High Solar System Count	2115	75
Low Solar System Count	93	1864

The random forest classifier correctly classified 96.5% of the High Solar System observations in the test dataset, and 95% of the Low Solar System observations in the test dataset. It had an overall accuracy of 95.9%.

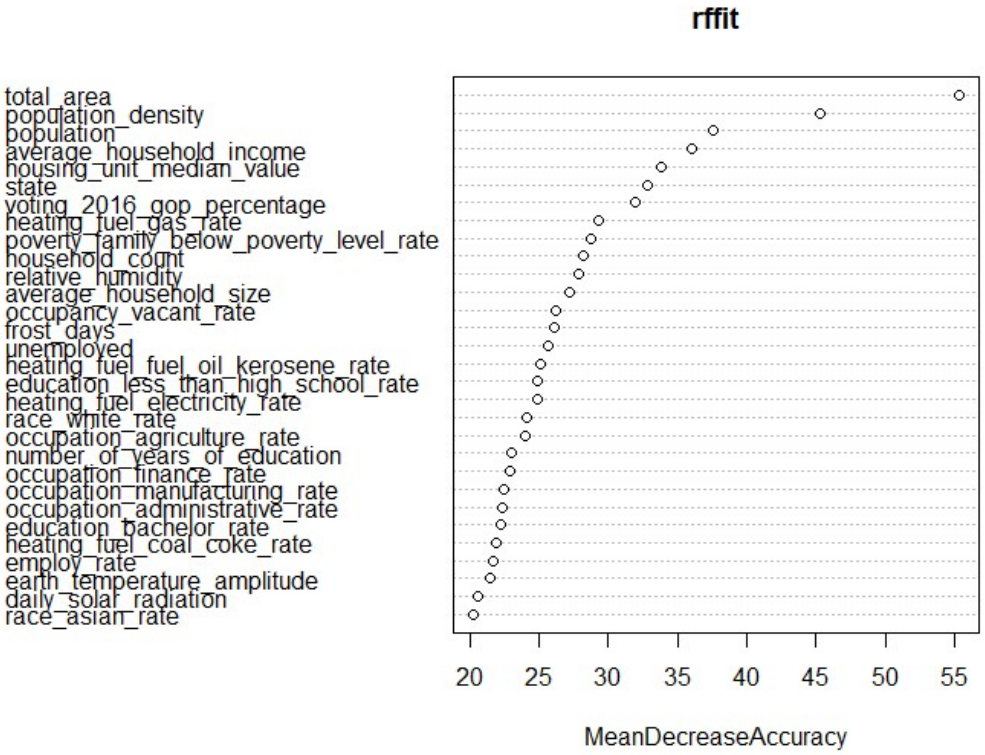

Figure 4. Plot of the impact on accuracy the removal of each variable has on the random forest classifier. Variables with an impact weaker than race_asian_rate not included in the plot.

From plot 4, we can see that the most important variable for the correct classification of the number of solar panels is total area. Removing the total are variable results in a mean decrease in accuracy of over 50%. Population density is also highly important for classification accurary. From there, 5 more variables are of high importance as well : Population, average household income, housing unit medium value, state, and voting 2016 gop percentage. From there the remaining variables are of declining importance.

Conclusion

It was discovered that for the solar system dataset, the best method of classification was random forests, which was able to classify with an accuracy of 96.5%. It was discovered that the 4 most important variables for predicting the solar system cover being high or low, was Total area, followed by Population Density, Population, and Average Household Income.

Binomial logistic regression was shown to provide a strong level of classification accuracy for the level of processor power provided, whereas the accuracy of the boosting classifier was comparable to that of the support vector machine classifier – which was able to achieve this accuracy with much lower processor power.

It is suggested that to model Solar system data, that binomial logistic regression be used as the classifier initially while data cleaning and selection is still being carried out, to quickly check the effect of variable cleaning/removal. Then, once the dataset has been finalised, that a random forest classifier be used, as this provides a higher level of accuracy, and also allows the importance of each variable to be analysed.

Future studies could look at the effect of removing insignificant variables from the model classification, as in this study all variables that did not provide multicollinearity or singularity issues, were maintained in the model. It may be discovered that removing variables with a low importance could remove noise from the classification and produce an even better classifier.

7. References

Armaroli, Nicola, and Vincenzo Balzani. 2011. *Energy for a sustainable world: from the oil age to a sun-powered future*. Weinheim, German: Wiley-VCH Verlag GmbH.

Couronné, Raphael, Philipp Probst, and Anne-Laure Boulesteix. 2018. "Random forest versus logistic regression: a large-scale benchmark experiment." *BMC Bioinformatics* 19 (1):1-14. doi: 10.1186/s12859-018-2264-5.

Ebinger, Jane O., Walter Vergara, and Bank World. 2011. *Climate impacts on energy systems: key issues for energy sector adaptation*. Washington, D.C: World Bank.

Graham, Michael H. 2003. "Confronting Multicollinearity in Ecological Multiple Regression." *Ecology* 84 (11):2809-2815. doi: 10.1890/02-3114.

Shen, Jianzhao, and Sujuan Gao. 2008. "A Solution to Separation and Multicollinearity in Multiple Logistic Regression." *Journal of data science : JDS* 6 (4):515-531.

Yu, J., Wang, Z., Majumdar, A. and Rajagopal, R., 2018. DeepSolar: A machine learning framework to efficiently construct a solar deployment database in the United States. *Joule*, 2(12), pp.2605-2617.

Appendix

Additional R packages used

The package “corrplot” was used to visualise the correlations between variables:

Taiyun Wei and Viliam Simko (2017). R package "corrplot": Visualization of a Correlation Matrix (Version 0.84). Available from https://github.com/taiyun/corrplot

List of removed variables

per_capita_income employed land_area
voting_2012_gop_percentage
voting_2012_dem_percentage
voting_2016_dem_percentage heating_degree_days
air_temperature heating_design_temperature
earth_temperature elevation housing_unit_count
electricity_consume_commercial
electricity_price_industrial electricity_price_transportation
electricity_consume_residentialelectricity_price_residential
electricity_consume_industrial
electricity_price_commercial electricity_consume_total
electricity_price_overall

Code used

```
#Import dataset
solar = read.csv("data_project_deepsolar.csv")

#Making solar system count a factor
solar$solar_system_count = factor(solar$solar_system_count)
#Looking at the number of observations in each category table(solar$solar_system_count)



################################################
#Removing highly correlated predictor variables as they are redundant and effect multicollinearity
################################################

solarnumeric = solar[ , purrr::map_lgl(solar, is.numeric)]

library(corrplot)
 corrplot(cor(solarnumeric[,c(unique(row.names((which(abs(cor(solarnumeric)) > 0.9 &
cor(solarnumeric) < 1, arr.ind=TRUE))))])) #Looking at the correlations of variables above the
absoulte value of 0.9



#Examining each pair of variables highly correlated and removing the one with the weakest correlation
to the response variable

which(abs(cor(solarnumeric)) > 0.9 & cor(solarnumeric) < 1, arr.ind=TRUE)

cor(solar$average_household_income, as.numeric(solar$solar_system_count)) cor(solar$per_capita_income,
as.numeric(solar$solar_system_count))
solarsub = subset(solar, select=- c(per_capita_income)) #The one with the weaker correlation to solar
system count is removed
cor(as.numeric(solar$solar_system_count), solar$employed) solarsub
= subset(solarsub, select=-c(employed))
  cor(as.numeric(solar$solar_system_count),
solar$total_area) cor(as.numeric(solar$solar_system_count),
solar$land_area) solarsub = subset(solarsub, select=-
c(land_area))
 cor(as.numeric(solar$solar_system_count), solar$electricity_price_overall)
cor(as.numeric(solar$solar_system_count), solar$electricity_price_commercial)
#Only the total and overall values were used in electricity
solarsub = subset(solarsub, select=-c(electricity_consume_residential, electricity_price_residential))
solarsub = subset(solarsub, select=-c(electricity_consume_industrial, electricity_price_commercial))
solarsub = subset(solarsub, select=-c(electricity_consume_commercial,electricity_price_industrial, ele
ctricity_price_transportation))
solarsub = subset(solarsub, select=-c(electricity_consume_residential, electricity_price_residential))
solarsub = subset(solarsub, select=-c(electricity_consume_industrial, electricity_price_commercial))
 cor(as.numeric(solar$solar_system_count),
solar$voting_2012_gop_percentage) cor(as.numeric(solar$solar_system_count),
solar$voting_2012_dem_percentage) cor(as.numeric(solar$solar_system_count),
solar$voting_2016_dem_percentage) cor(as.numeric(solar$solar_system_count),
solar$voting_2016_gop_percentage)
#2016 gop percentage has strongest negative correlation
solarsub = subset(solarsub, select=-c(voting_2012_gop_percentage, voting_2012_dem_percentage, voting_2
016_dem_percentage))

cor(as.numeric(solar$solar_system_count), solar$frost_days) cor(as.numeric(solar$solar_system_count),
solar$earth_temperature) cor(as.numeric(solar$solar_system_count), solar$heating_degree_days)
cor(as.numeric(solar$solar_system_count), solar$air_temperature)
cor(as.numeric(solar$solar_system_count), solar$heating_design_temperature)
 solarsub = subset(solarsub, select=-
c(heating_degree_days,air_temperature,heating_design_temperature,e arth_temperature))
```

```r
cor(as.numeric(solar$solar_system_count),solar$atmospheric_pressure)
cor(as.numeric(solar$solar_system_count),solar$elevation) solarsub =
subset(solarsub, select=-c(elevation))

cor(solar$household_count,solar$housing_unit_count)
cor(as.numeric(solar$solar_system_count),solar$household_count)
cor(as.numeric(solar$solar_system_count),solar$housing_unit_count)

solarsub = subset(solarsub, select=-c(housing_unit_count))



#No predictor variables have a correlation greather than 0.9 now. solarsubnumeric
= solarsub[ , purrr::map_lgl(solarsub, is.numeric)]
which(abs(cor(solarsubnumeric)) > 0.9 & cor(solarsubnumeric) < 1, arr.ind=TRUE)



#But still issues with singularities. All educations add up to 1 so remove 1, same with races and elec
trcity
solarsub = subset(solarsub, select=-c(race_two_more_rate))
solarsub = subset(solarsub, select=-c(education_bachelor_rate,education_high_school_graduate_rate,
education_less_than_high_school_rate,education_master_rate, educ ation_doctoral_rate,
                                      education_professional_school_rate, education_college_rate))

################# #EDA############
#################

#Density plots to look at distribution
plot(density(solar$average_household_income)) plot(density(solar$employed))

#Creating boxplots

boxplot(solar$housing_unit_median_value ~solar$solar_system_count, ylab = "Housing Unit Median Value (
$)", xlab = "Solar System Count", col=c("#339900", "#00999F"))
points(x = 1:2,y=tapply(solar$housing_unit_median_value,solar$solar_system_count,mean),pch=19)

boxplot(solar$cooling_degree_days ~ solar$solar_system_count, ylab = "Cooling degree days", xlab = "So
lar System Count", col=c("#339900", "#00999F"))

boxplot(solar$earth_temperature_amplitude ~ solar$solar_system_count, ylab = "Earth Temperature", xlab
= "Solar System Count", col=c("#339900", "#00999F"))

boxplot(solar$daily_solar_radiation ~ solar$solar_system_count, ylab = "Daily solar radiation", xlab =
"Solar System Count", col=c("#339900", "#00999F"))
  boxplot(solar$frost_days ~ solar$solar_system_count, ylab = "Frost Days", xlab = "Solar System
Count", col=c("#339900", "#00999F"))
points(x = 1:2,y=tapply(solar$frost_days,solar$solar_system_count,mean),pch=19)



##############################

##############################

#Data is broken into training ,test and validation sets of side 60-20-20 ind
= sapply(solarsub, is.numeric)
solarsub[ind] = lapply(solarsub[ind], scale) #Scale
set.seed(1996) D = nrow(solarsub) keep =
sample(1:D, 16589) test = setdiff(1:D, keep) dat =
solarsub[keep,] dat_test = solarsub[test,]
#Validation D
= nrow(dat)
keep = sample(1:D, 12442)
val = setdiff(1:D, keep)
dat_val = dat[val,] dat =
dat[keep,]
```

```r
#binomial modelling
reg = glm(solar_system_count ~ .,data=dat, family = "binomial") summary(reg)

predict2 = predict(reg, newdata= dat_val) #Fitting model to validation data

pred = ifelse(predict2 > 0.5, 1,0)    #Setting tau value to 0.5. A value over 0.5 is classified as low
solar system count
table(dat_val$solar_system_count, pred)
sum(diag(table(dat_val$solar_system_count, pred)))/ sum(table(dat_val$solar_system_count, pred)) #Scor
e of 0.876

#Timing classifier
{
start = Sys.time()
reg = glm(solar_system_count ~ .,data=dat, family = "binomial")
end = Sys.time() end - start
}
 #SVM
library(kernlab)
ksvfit = ksvm(solar_system_count ~ .,data=dat)
ksvfit
predksv = predict(ksvfit, type="response", newdata=dat_val) #Fitting model to validation data
table(dat_val$solar_system_count, predksv)
sum(diag(table(dat_val$solar_system_count, predksv)))/ sum(table(dat_val$solar_system_count, predksv))
#0.89



#Timing classifier
{
  start = Sys.time()
  ksvfit = ksvm(solar_system_count ~ .,data=dat)
end = Sys.time()   end - start
}



#Boosting  library(adabag)
boostfit = boosting(solar_system_count ~ ., data= dat, coeflearn="Breiman",boos=FALSE) boostfit

predboost = predict(boostfit, type="response", newdata=dat_val) #Fitting model to validation data
table(dat_val$solar_system_count, predboost$class)
sum(diag(table(dat_val$solar_system_count, predboost$class)))/ sum(table(dat_val$solar_system_count, p
redboost$class)) #0.895

#Timing classifier
{
  start = Sys.time()
  boostfit = boosting(solar_system_count ~ ., data= dat, coeflearn="Breiman",boos=FALSE)
end = Sys.time()   end - start
}



##Random Forest library(randomForest)
rffit = randomForest(solar_system_count ~ .,data=dat) rffit$importance
 predrf = predict(rffit, type="response", newdata=dat_val) #Fitting model to validation
data table(dat_val$solar_system_count, predrf)
sum(diag(table(dat_val$solar_system_count, predrf)))/ sum(table(dat_val$solar_system_count, predrf)) #
Score of 0.904

#Timing classifier
{
  start = Sys.time()
```

```
  rffit = randomForest(solar_system_count ~ .,data=dat, importance= TRUE)
end = Sys.time()   end - start
}
```

```
#Best Model on test set

predrf2 = predict(rffit, type="response", newdata=dat_test)  #Best model is used on test dataset
table(dat_test$solar_system_count, predrf2)
sum(diag(table(dat_test$solar_system_count, predrf2)))/ sum(table(dat_test$solar_system_count, predrf2
))
importance(rffit, type=1)
varImpPlot(rffit, type=1)    #Looking at the importance of each variable in the classification of sola
r system count rffit
```