Supervised Machine Learning to Predict the Number of Solar Power Systems in a Tile

Abstract

Logistic regression, Random Forest, Classification tree, bagging, boosting and SVM models are built to predict the number of Solar Power Systems in a tile. The dataset is a subset of the Deep Solar database. The accuracy across each model is calculated through plots and confusion matrix. We will show how each model predicts the output variable and performance of 4 models are calculated to know the best suitable model for the process.

Contents

1. Introduction

Supervised machine learning is subbranch of Machine Learning. The main idea behind these models are they take certain part of the data as the input and train the model based on the subset of these data and these trained models will supervise to predict the correct label based on the inputs provided. The dataset consists of 20736 observations with 81 unique variables. The predicted label is the binary data with low representing less than or equal to 10 number of solar system and high represents greater 10 solar systems.

2. Methods

The methods involve two steps the first one is data analysing and cleaning, and the second one is the building various supervised models

2.1 Data Analysing and Cleaning

The proportion of high to low Solar system count is checked here in this dataset we can see that the proportion is close to each other. If there existed a lower of one quantity then proportioned had to be balanced to make the analysis unbiased.

The structure of the data is analysed, we see that there are 4 variables that are non-numeric. so we will go ahead by identify the missing and duplicate values. There was no duplicate variables out of 81 variables. There are no missing values in the data as well.

The correlation for numeric variables are checked, the variables convey the same information if the correlation is close to 1, and the correlation varies from -1 to 1. We see that when we check the correlation for the numeric data keeping the cut-of at .7 we get 27 different variable which has the correlation with other variable in the data. The list of 31 variables with their correlated variable has been listed below and out of these 31 variable 27 has high correlation and 4 has moderate correlation, 27 variables are eliminated from the original data and a new data set is created.

Column Number from the main raw data	Variable Name- Correlated with variable
1	Average House hold income has a high correlation with voting 2012 dem % and per capita income so we considering voting 2012 dem %
2	Employed has a high correlation with population and household_count
4	land area has a high correlation with total area
5	per capita income has a high correlation with average_household_income
6	population has a high correlation with employed
11	education less than highschool rate has a high correlation with poverty_family_below_poverty_level_rate 71% need to take into consideration
14	education bachelor rate has a high correlation with number_of_years_of_education
15	education master rate has a high correlation with number_of_years_of_education
28	heating fuel electricity rate has less correlation 67% hence will be taken into consideration
33	electricity price residential has a high correlation with electricity_price_overall and electricity_price_commercial
34	electricity price commercial has a high correlation with electricity_price_overall and electricity_price_residential
35	electricity price industrial has a high correlation with electricity_price_commercial and electricity_price_overall
37	electricity price overall has a high correlation with electricity_price_commercial and electricity_price_residential
38	electricty consume residential has a high correlation with electricity_consume_commercial
39	electricity consume commercial has a high correlation with electricity_consume_industrial
41	electricity consume total has a high correlation with electricity_consume_industrial
44	Housing Unit Count has a high correlation with household_count
45	Housing Unit Median value has a high correlation with average_household_income
47	heating design temperature has a high correlation with air_temperature and earth_temperature
50	frost days has a high correlation with heating_degree_days
51	air temperature has a high correlation with earth_temperature and cooling_degree_days and heating_design_temperature

52	relative humidity has a high correlation with atmospheric_pressure(Moderate)
53	daily solar radiation has a high correlation with earth_temperature
54	atmospheric pressure has a high correlation with relative_humidity(moderate)
56	earth temp has a high correlation with air_temperature and heating_design_temperature and cooling_degree_days
57	heating degree days has a high correlation with frost_days and air_temperature,
72	voting 2016 dem % has a high correlation with voting_2012_dem_percentage
73	voting 2016 gop % has a high correlation with voting_2012_gop_percentage
74	voting 2012 gop % has a high correlation with voting_2016_gop_percentage
76	Number of years of Education has a high correlation with education_bachelor_rate and education_master_rate

Once the new data is created the new data is scaled since there is a variation in the data from decimal places to thousands. The scaled data is bound again with the non-numeric data. The data is then partitioned randomly from 70% of the data into training and rest 30% in to the testing data. The data is taken randomly since the data is arranged in the order, if we consider directly dividing then the model will overfit for the training and the accuracy for the new prediction will be very low. The data is sufficiently large enough hence forth no iterations are done in the first section of the code, the second section involved comparing 5 different models over 50 iterations.

3. Models

3.1 Logistic Regression

The dependent variable in the given set is dichotomous and logistic regression model is very much suitable when the dependable variable is binary, basically here the log odds of the outcome are modelled as a linear combination of the predictor variables.

No additional library was used in building this model, and the data was used as is after the exploratory data analysis method. The training set consisted of 75% of randomly arranged data and testing set consisted of 25% of the remaining data.

Once the model is trained the summary of the fit gives us the details of the trained model.

```
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.8898  -0.3451  -0.0851   0.3463   3.5000
```

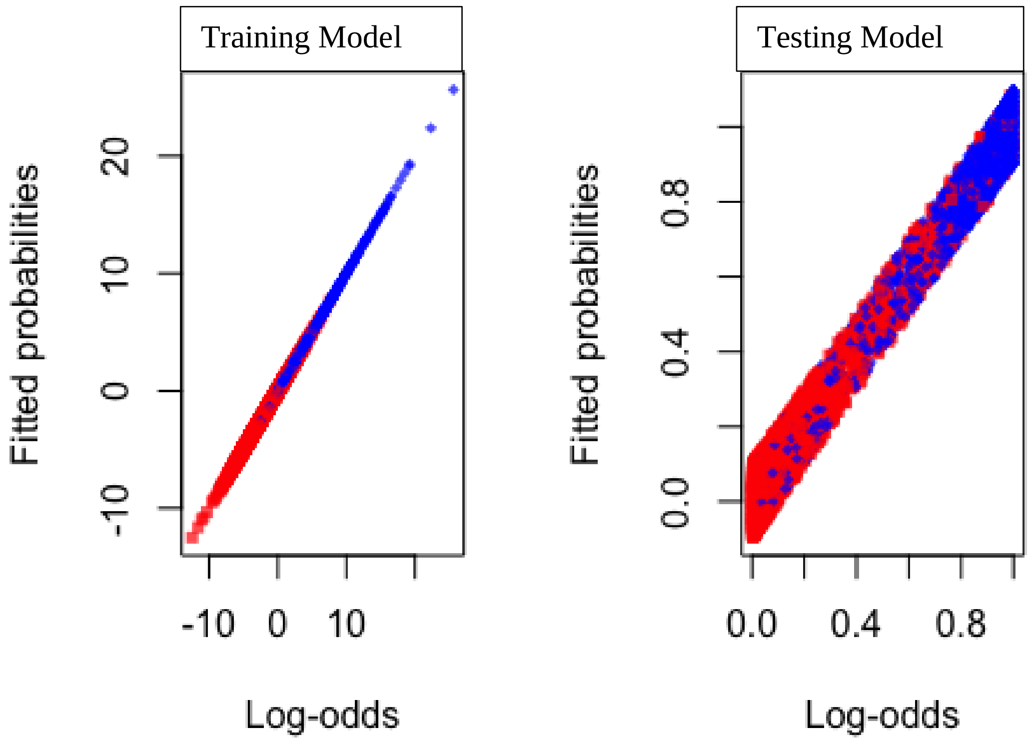
```
This part of the output shows the deviance residual for individual cases, the
minimum deviance is -3.8898, the average deviance is -0.0851 and the maximum
deviance could be 3.500. The range is from -3.8 to 3.5.
```

The next section of the output tells us for every unit change in the dependent variable, the log odds changes of the independent variables are noted. If its "+" then it increases and if it's "-" then it decreases by certain value. It also gives us the significant variables based on the z values from the summary we see that if the z value is less than 0.05 then is considered to be significant, it is denoted by *. We cannot come to a conclusion just based on this if we try to eliminate all the non-significant variables we might just end up overfitting the data for the training model. The list of variables with their significance level is given as follows.

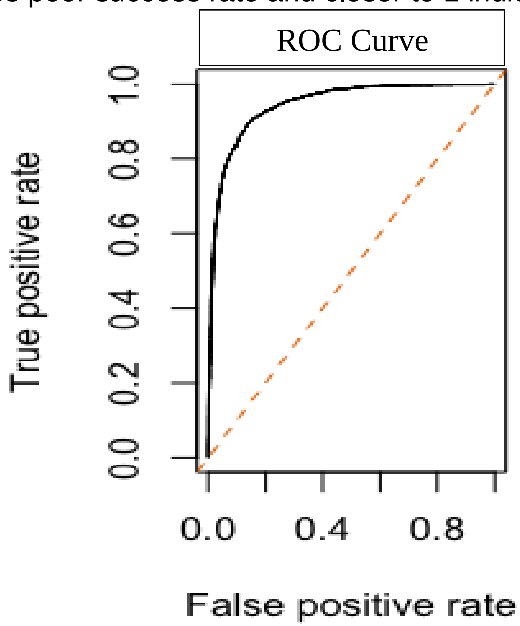
```
## gini_index
## population_density                          ***
## total_area
## unemployed                                  ***
## water_area
## education_less_than_high_school_rate        ***
## education_high_school_graduate_rate         .
## education_college_rate
## education_professional_school_rate
## education_doctoral_rate
## race_white_rate
## race_black_africa_rate
## race_indian_alaska_rate
## race_asian_rate
## race_islander_rate
## race_other_rate
## race_two_more_rate
## employ_rate                                 ***
## poverty_family_below_poverty_level_rate     ***
## heating_fuel_gas_rate
## heating_fuel_electricity_rate
## heating_fuel_fuel_oil_kerosene_rate
## heating_fuel_coal_coke_rate                 **
## heating_fuel_other_rate                     *
## heating_fuel_none_rate                      .
## electricity_price_transportation            ***
## electricity_consume_industrial              ***
## household_count                             ***
## average_household_size                      ***
## elevation
## cooling_design_temperature                  ***
## earth_temperature_amplitude                 ***
## relative_humidity                           ***
## atmospheric_pressure
## wind_speed
## cooling_degree_days                         ***
## occupation_construction_rate
## occupation_public_rate                      ***
## occupation_information_rate
## occupation_finance_rate                     **
## occupation_education_rate
## occupation_administrative_rate              **
## occupation_manufacturing_rate               **
## occupation_wholesale_rate
## occupation_retail_rate
## occupation_transportation_rate
## occupation_arts_rate                        *
## occupation_agriculture_rate                 **
## occupancy_vacant_rate                       ***
## voting_2012_gop_percentage
## stateca                                     ***
## stateil                                     ***
## statemi                                     ***
## statenj                                     ***
## stateny
## statetx
## voting_2016_dem_winTrue                     **
## voting_2012_dem_winTrue                     ***
```

Once the model is trained is check for the testing data to see how well the model is performed, a cross table matrix is create with the predicted outcome label and the original data from the testing set. Then the accuracy is calculated.

The plot of fitted probabilities vs log odd. From the graph we can see that the fitted probabilities of the testing model is greater than 90%.



One more way to check the accuracy of the plot is by using ROC curve, so the ROC curve gives the best cut-off value whether the predicted data of the new observation gives success or failureil. The range varies from 0 to 1. Closer to 0 indicates poor success rate and closer to 1 indicates better success rate.



From the ROC curve we can see that the success rate to predict a new input is closer to 1. The area under the curve gives the information about the accuracy of the model. The accuracy is 94.60%.

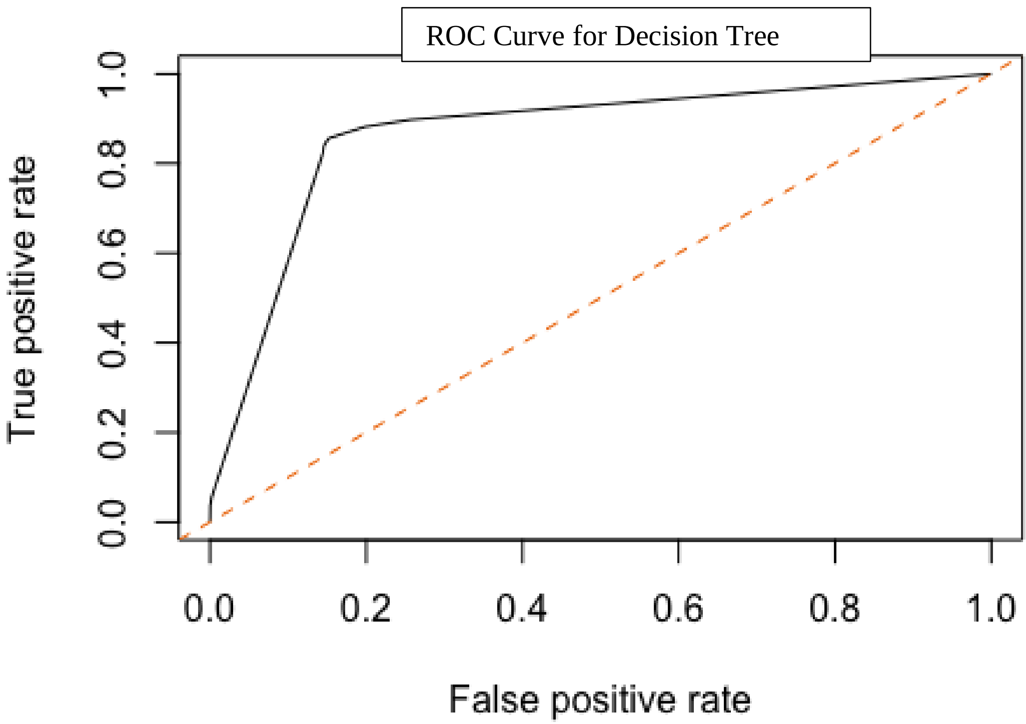
3.2 Classification Tree

One of the versatile machine learning algorithm that can perform both regression and classification task. It is the basic component of random forest. Two libraries packages are used for this model 1. rpart and 2. partykit. Two things need to be taken care of while building the model the response variable needs to factored or type="class" need to be mentioned while building the model. We take the data is available after the EDA process for building the data since the measures are already taken care during the EDA. Once the model is ready we take the summary of the model to see the significant variables among the ones already included and also the number of observation under each node.

```
## Variable importance
##                                state                 relative_humidity
##                                   20                                19
##       electricity_consume_industrial    electricity_price_transportation
##                                   18                                11
##                 atmospheric_pressure                         elevation
##                                   11                                11
##                           total_area                population_density
##                                    4                                 3
## heating_fuel_fuel_oil_kerosene_rate           voting_2012_gop_percentage
##                                    2                                 1
```

Then we predict for the new set of testing data. Once the outcome variables are predicted for the testing data the accuracy across the predicted variables are checked using the cross matrix. The accuracy from the cross matrix is 85.18%.

To have a better result we go ahead with the ROC curve, again here the range is from 0 to 1 and closer 0 indicates poor success rate and closer 1 indicates better success rate for the prediction of the new outcome variable.



From the ROC curve we see that the success rate is between 85 to 90%. The area under the curve is 86.62%

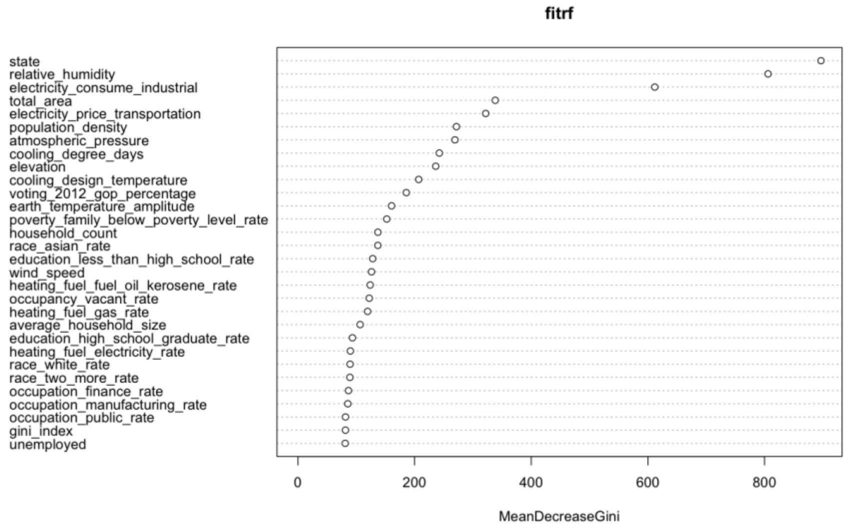
3.3 Random Forest

The basic component of Random Forest is decision trees, here we train a group of decision tree classifiers each on different random set of the train data, aggregate the result of multiple predictors to make the prediction. Here we make use of the randomForest library package. Again the training data set is used from the training data made after the EDA. Here we can pass the argument based on the number of trees required. Once the model is trained, the fitted model gives us the following details the number of trees and how often the tree is split. The OOB estimate of error rate is 10.14% and classification error for high is 0.09 and for low is 0.1077.

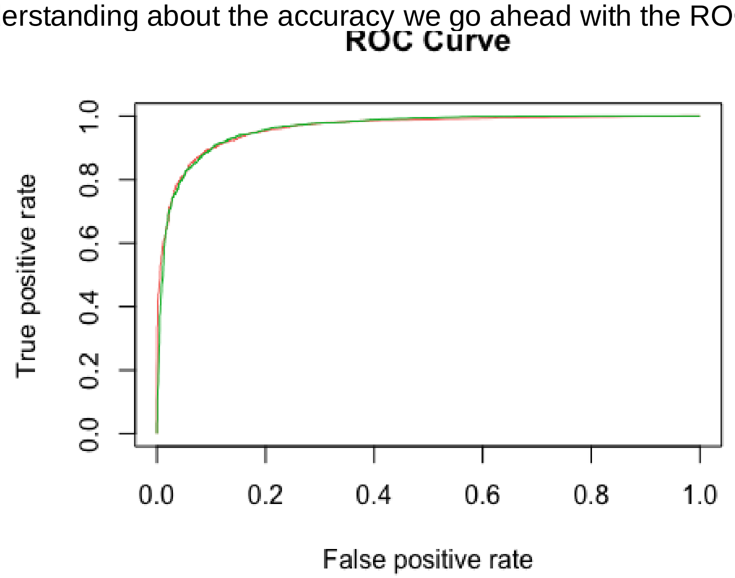
```
## Variable importance
```
Number of trees: 500
No. of variables tried at each split: 7
OOB estimate of error rate: 10.14%
Confusion matrix:

	high	low	class.error
high	7393	782	0.09565749
low	795	6582	0.10776739

The important variables are also plotted from the trained model. As the mean Decrease gini increases the importance of the variables also increases



Once the model is ready, it is tested by predicting the response variable for testing data. The accuracy is calculated through cross table matrix. The accuracy is 90.52%.
To get a clear understanding about the accuracy we go ahead with the ROC curve.



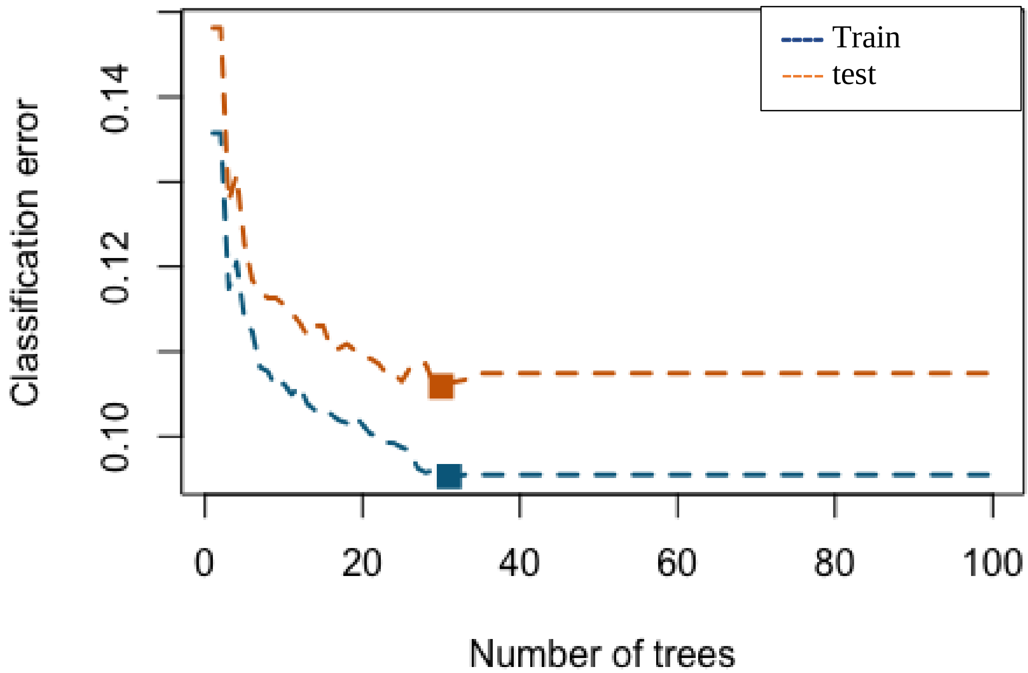
From the plot we can see two different colour one indicating the high and the other indicate the low success/failure rate. We can see that the accuracy is closer to 1 indicating high success rate in the prediction of the new outcome label based on the inputs. The area under the curve gives the accuracy of the success rate . We observe that 96.19% of true positive is present for this model.

3.4 Bagging

This model is improve the accuracy and stability of the machine learning algorithms used in statistical classification and regression. It reduces variance and avoid overfitting of the data. This is special case of model approaching method. The library used is adabag, the data is already pre-processed hence no changes required. The model is built on the training data and is tested on the testing data. Using the cross table confusion matrix the accuracy is checked across the trained model. The accuracy if found to be 84.33%

3.5 Boosting

The first algorithm is trained on the entire dataset and subsequent algorithms are built by fitting the residual of the first algorithm, thus giving higher weight to those observation those were poorly predicted. This model basically boost the performance of the models. The library used in adabag. No changes to the training and testing data were done. The model is built using the training data.



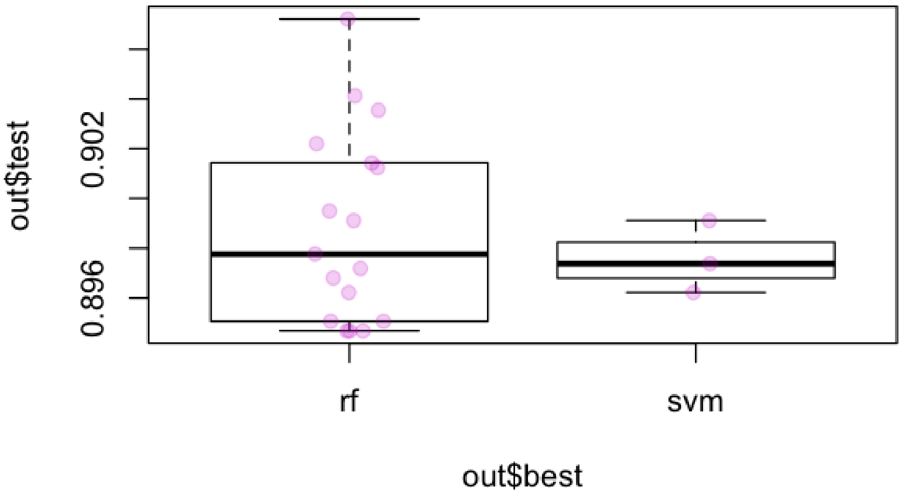
The classification error vs fitted tree plot shows how the error decreases for the predicted model and training model. We see that classification error for testing data is comparatively high for testing data then that of the training data.

3.6 Performance testing of 5 models together

5 models are iterated over 20 times, the data is divided into training, validation and testing. Initially a model is built based on training data, then training data is further divided into validation, on which the model is tested for accuracy. The first comparison between the 5 models is checked then again the model is tested with testing data and checked for accuracy. Whichever model has the best test accuracy is noted. The 5 models used are logistic regression, Decision tree, Random forest, bagging and SVM. The libraries for the first models are same as used in the earlier sections. The library package used to build the SVM model is kernlab.

SVM is a data classification method that separates data based on hyperplanes. It is very much useful to separate the data with labels, basically divides the space multiple times into segments.

Once the iterations are complete for all 5 models, the maximum repeated model is checked on every iteration. From the data we can see that Random Forest has repeated 85% where as SVM has repeated 15% the entire number of iterations.



The box plot is plotted for the best repeated data, we can see that for random forest the accuracy varies from 89% to 91% and for SVM we can see that the test accuracy ranges from 89% to 90%.

4. Results

Without iterations the results are as follows:

a. Logistic Regression has an accuracy of : `0.8868313`
   The area under the ROC curve is : `0.9460358`

b. Decision Tree has an accuracy of: `0.8518519`
   The area under the ROC curve : `0.8662943`

c. Random Forest has an accuracy of: 0.9052855
   The area under the ROC curve: 0.9675643

d. Bagging has an accuracy of: `0.8433642`

e. Boosting has an accuracy of: 0.8939043

```
## $error rate
## [1] 0.107446
```

Over the iterations performance and the best accuracy check

We see that Random Forest has repeated 85% of the times and SVM has repeated 15% of the time the models were checked for the accuracy. Random forest has a minimum accuracy of 89.47 and maximum accuracy of 90.72, with the median accuracy of 89.89. SVM has a minimum accuracy of 89.62 and maximum accuracy of 89.91 and the median accuracy is 89.74. The rest of the models varies as follows, Decision tree has maximum accuracy of 86.55%, bagging has a maximum accuracy of 86.44%.

$random_forest

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
0.8947	0.8951	0.8978	0.8989	0.9014	0.9072

$svm
Min. 1st Qu. Median Mean 3rd Qu. Max.
0.8962 0.8968 0.8974 0.8976 0.8982 0.8991

$Decison Tree
Min. 1st Qu. Median Mean 3rd Qu. Max.
0.8262 0.8443 0.8501 0.8491 0.8549 0.8655

$Logistic Regression
Min. 1st Qu. Median Mean 3rd Qu. Max.
0.8764 0.8798 0.8818 0.8821 0.8833 0.8900

$Bagging
Min. 1st Qu. Median Mean 3rd Qu. Max.
0.8495 0.8537 0.8568 0.8572 0.8605 0.8644

The % of models repeated as the best model from the iterations
rf svm
0.85 .15

5. Conclusions

From the results it is very much evident that random forest gives the best prediction for the current dataset with the selected variables, followed by SVM then Logistic regression and Decision trees. Further analysis can be also done by changing the variables during EDA process.

6. References

1. New library package “caret” was used to create data portioning in the process. https://www.rdocumentation.org/packages/caret/versions/6.0-86/topics/createDataPartition

7. Code

Load the dataset and information data

```
 solarsystem<-read.csv("//Users//apple//Documents//UCD//sem2//machine learning//Projec
t//data_project_deepsolar.csv",na.strings = "na")

info<-read.csv("//Users//apple//Documents//UCD//sem2//machine learning//Project//data_
project_deepsolar_info.csv")
```

Data Cleaning

```
#Load the library

library(caret)

## Loading required package: lattice

## Loading required package: ggplot2



#the proportion
prop.table(table(solarsystem$solar_system_count))

## 
##      high       low 
## 0.5256559 0.4743441

#non-numeric columns are made into factor with respective levels
data<-solarsystem
data$solar_system_count <-as.factor(data$solar_system_count)
data$state<-as.factor(data$state)
data$voting_2016_dem_win<-as.factor(data$voting_2016_dem_win)
data$voting_2012_dem_win<-as.factor(data$voting_2012_dem_win)


#Checking for Duplicate variables
dup_data = data[duplicated(data), ]

#Checking for missing values
colSums(is.na(data))

##                        solar_system_count                                     state 
##                                         0                                         0 
##                  average_household_income                                  employed 
##                                         0                                         0 
##                                gini_index                                 land_area 
##                                         0                                         0 
##                         per_capita_income                                population 
##                                         0                                         0 
##                        population_density                                total_area 
##                                         0                                         0 
##                                unemployed                                water_area 
##                                         0                                         0 
##      education_less_than_high_school_rate     education_high_school_graduate_rate 
##                                         0                                         0 
##                    education_college_rate                   education_bachelor_rate 
##                                         0                                         0 
##                     education_master_rate     education_professional_school_rate 
##                                         0                                         0 
##                   education_doctoral_rate                           race_white_rate 
```

```
##                                        0                                        0
##                     race_black_africa_rate                    race_indian_alaska_rate
##                                        0                                        0
##                          race_asian_rate                      race_islander_rate
##                                        0                                        0
##                          race_other_rate                      race_two_more_rate
##                                        0                                        0
##                              employ_rate poverty_family_below_poverty_level_rate
##                                        0                                        0
##                    heating_fuel_gas_rate            heating_fuel_electricity_rate
##                                        0                                        0
##      heating_fuel_fuel_oil_kerosene_rate              heating_fuel_coal_coke_rate
##                                        0                                        0
##                  heating_fuel_other_rate                   heating_fuel_none_rate
##                                        0                                        0
##             electricity_price_residential             electricity_price_commercial
##                                        0                                        0
##              electricity_price_industrial         electricity_price_transportation
##                                        0                                        0
##                   electricity_price_overall          electricity_consume_residential
##                                        0                                        0
##            electricity_consume_commercial           electricity_consume_industrial
##                                        0                                        0
##                electricity_consume_total                          household_count
##                                        0                                        0
##                   average_household_size                       housing_unit_count
##                                        0                                        0
##                  housing_unit_median_value                                elevation
##                                        0                                        0
##                heating_design_temperature                cooling_design_temperature
##                                        0                                        0
##              earth_temperature_amplitude                               frost_days
##                                        0                                        0
##                          air_temperature                        relative_humidity
##                                        0                                        0
##                     daily_solar_radiation                     atmospheric_pressure
##                                        0                                        0
##                               wind_speed                        earth_temperature
##                                        0                                        0
##                      heating_degree_days                      cooling_degree_days
##                                        0                                        0
##             occupation_construction_rate                   occupation_public_rate
##                                        0                                        0
##              occupation_information_rate                  occupation_finance_rate
##                                        0                                        0
##                occupation_education_rate           occupation_administrative_rate
##                                        0                                        0
##            occupation_manufacturing_rate                occupation_wholesale_rate
##                                        0                                        0
##                   occupation_retail_rate           occupation_transportation_rate
##                                        0                                        0
##                     occupation_arts_rate              occupation_agriculture_rate
##                                        0                                        0
##                    occupancy_vacant_rate                voting_2016_dem_percentage
##                                        0                                        0
##                voting_2016_gop_percentage                      voting_2016_dem_win
##                                        0                                        0
##                voting_2012_dem_percentage                voting_2012_gop_percentage
##                                        0                                        0
```

```
##                      voting_2012_dem_win            number_of_years_of_education
##                                        0                                       0
##                                diversity
##
```

```
#Non-numeric data is removed from the data

new_data<-data[,-c(1,2)]
new_data<-new_data[,-c(74,77)]




#Correlation matrix is created
cor_mat<-cor(new_data)
#The variables with highest correlation is mapped by the column number
corr_variable<-findCorrelation(cor_mat,cutoff = 0.7,)
corr_variable
```

```
##  [1] 45 76 15 56 53 50 57 51 11 47  5 14  1 34 73 72 37 38 35 74 39 28 33 52 41
## [26] 77 54  2 44  6  4
```

```
new_data<-new_data[,-c(45, 76 ,15 ,56 ,53 ,50 ,57 ,51  ,47 , 5, 14 , 1, 34, 73, 72, 37
,38, 35, 74, 39  ,33,  41 ,77,  2 ,44 , 6 , 4)]

#Correlation matrix is created
cor_mat<-cor(new_data)

#Correlation is checked again
corr_variable<-findCorrelation(cor_mat,cutoff = 0.8)
corr_variable
```

```
## [1] 34 21
```

```
#Bind the rest of the column to the new_data with 50 variables
temp_data<-new_data
#Scaling is performed since the variation among the variables was comparitively high


temp_data<-scale(temp_data)
#Column with non-numeric data is binded
temp_data<-cbind(temp_data,data[,c(2,76,79,1)])



#Data partition for model building and testing, p=0.75 means 75% of the data is traini
ng and rest of it is the testing data


train_data<-createDataPartition(temp_data$solar_system_count,p=.75,list = F)
training<-temp_data[train_data,]
testing<-temp_data[-train_data,]
```

##Logisitic Regression

```
lgmmodel<-glm(solar_system_count~.,data = training,family = "binomial")
summary(lgmmodel)
```

```
##
## Call:
## glm(formula = solar_system_count ~ ., family = "binomial", data = training)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.8898  -0.3451  -0.0851   0.3463   3.5000
##
## Coefficients: (2 not defined because of singularities)
##                                           Estimate Std. Error z value Pr(>|z|)
## (Intercept)                             -2.146e+00  3.395e-01  -6.320 2.62e-10
## gini_index                              -4.262e-02  3.730e-02  -1.142 0.253279
## population_density                       1.760e+00  7.315e-02  24.065  < 2e-16
## total_area                               1.981e-03  2.431e-02   0.081 0.935060
## unemployed                              -4.947e-01  7.198e-02  -6.872 6.31e-12
## water_area                               1.072e-02  3.194e-02   0.335 0.737256
## education_less_than_high_school_rate     3.197e-01  7.730e-02   4.136 3.53e-05
## education_high_school_graduate_rate     -9.497e-02  4.930e-02  -1.926 0.054066
## education_college_rate                  -3.759e-02  5.260e-02  -0.715 0.474875
## education_professional_school_rate       6.714e-02  5.048e-02   1.330 0.183503
## education_doctoral_rate                 -5.505e-02  4.491e-02  -1.226 0.220257
## race_white_rate                          1.298e+08  1.712e+10   0.008 0.993949
## race_black_africa_rate                   9.732e+07  1.283e+10   0.008 0.993949
## race_indian_alaska_rate                  1.750e+07  2.308e+09   0.008 0.993949
## race_asian_rate                          6.735e+07  8.880e+09   0.008 0.993949
## race_islander_rate                       3.788e+06  4.995e+08   0.008 0.993949
## race_other_rate                          6.103e+07  8.047e+09   0.008 0.993949
## race_two_more_rate                       1.460e+07  1.925e+09   0.008 0.993949
## employ_rate                             -4.091e-01  7.168e-02  -5.708 1.14e-08
## poverty_family_below_poverty_level_rate  2.460e-01  5.446e-02   4.518 6.24e-06
## heating_fuel_gas_rate                    3.602e+00  2.511e+00   1.434 0.151440
## heating_fuel_electricity_rate            3.607e+00  2.486e+00   1.451 0.146783
## heating_fuel_fuel_oil_kerosene_rate      1.838e+00  1.328e+00   1.384 0.166207
## heating_fuel_coal_coke_rate              1.134e+00  4.089e-01   2.773 0.005556
## heating_fuel_other_rate                  2.385e-01  1.144e-01   2.084 0.037117
## heating_fuel_none_rate                   8.069e-01  4.286e-01   1.883 0.059727
## electricity_price_transportation         5.464e+00  7.573e-01   7.215 5.41e-13
## electricity_consume_industrial          -2.703e+01  3.786e+00  -7.139 9.39e-13
## household_count                         -3.560e-01  5.107e-02  -6.971 3.15e-12
## average_household_size                  -6.781e-01  4.795e-02 -14.141  < 2e-16
## elevation                               -2.493e-01  1.429e+00  -0.175 0.861446
## cooling_design_temperature              -5.737e-01  1.661e-01  -3.454 0.000553
## earth_temperature_amplitude              9.567e-01  1.409e-01   6.790 1.12e-11
## relative_humidity                        1.687e+00  2.002e-01   8.425  < 2e-16
## atmospheric_pressure                    -6.491e-01  1.482e+00  -0.438 0.661456
## wind_speed                               6.497e-02  4.025e-02   1.614 0.106489
## cooling_degree_days                      1.185e+00  1.411e-01   8.399  < 2e-16
## occupation_construction_rate            -2.462e-02  5.595e-02  -0.440 0.659945
## occupation_public_rate                  -2.428e-01  4.727e-02  -5.136 2.81e-07
## occupation_information_rate             -5.882e-02  4.345e-02  -1.354 0.175807
## occupation_finance_rate                 -1.719e-01  5.610e-02  -3.063 0.002188
## occupation_education_rate               -9.041e-02  8.516e-02  -1.062 0.288395
## occupation_administrative_rate          -2.270e-01  6.979e-02  -3.253 0.001142
## occupation_manufacturing_rate            2.016e-01  7.122e-02   2.831 0.004642
## occupation_wholesale_rate                3.711e-02  3.621e-02   1.025 0.305453
## occupation_retail_rate                  -4.594e-02  5.283e-02  -0.870 0.384494
## occupation_transportation_rate          -3.519e-02  4.554e-02  -0.773 0.439780
## occupation_arts_rate                    -1.292e-01  6.168e-02  -2.094 0.036239
## occupation_agriculture_rate              1.542e-01  5.693e-02   2.709 0.006755
```

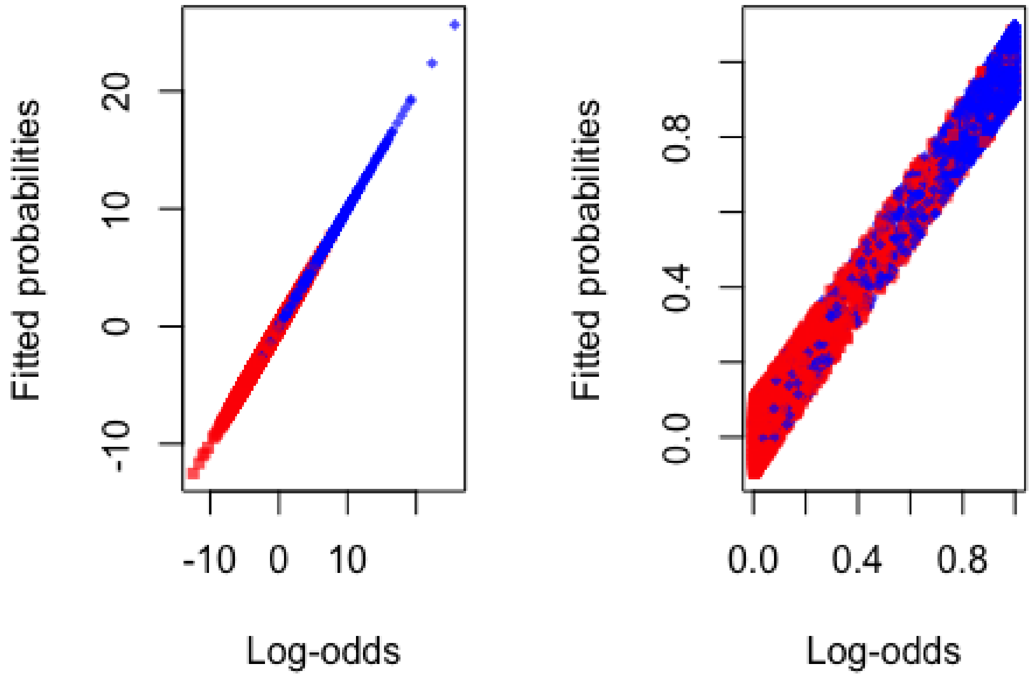
```
## occupancy_vacant_rate                      1.704e-01  3.407e-02   5.001 5.70e-07
## voting_2012_gop_percentage                -8.076e-03  5.953e-02  -0.136 0.892085
## stateca                                   -1.901e+01  2.540e+00  -7.485 7.14e-14
## stateil                                    7.722e+01  1.042e+01   7.412 1.24e-13
## statemi                                    3.936e+01  5.240e+00   7.511 5.88e-14
## statenj                                   -1.687e+01  2.289e+00  -7.370 1.70e-13
## stateny                                           NA         NA      NA       NA
## statetx                                           NA         NA      NA       NA
## voting_2016_dem_winTrue                    3.345e-01  1.041e-01   3.214 0.001308
## voting_2012_dem_winTrue                   -7.880e-01  1.110e-01  -7.099 1.26e-12
##
## (Intercept)                               ***
## gini_index
## population_density                        ***
## total_area
## unemployed                                ***
## water_area
## education_less_than_high_school_rate      ***
## education_high_school_graduate_rate       .
## education_college_rate
## education_professional_school_rate
## education_doctoral_rate
## race_white_rate
## race_black_africa_rate
## race_indian_alaska_rate
## race_asian_rate
## race_islander_rate
## race_other_rate
## race_two_more_rate
## employ_rate                               ***
## poverty_family_below_poverty_level_rate ***
## heating_fuel_gas_rate
## heating_fuel_electricity_rate
## heating_fuel_fuel_oil_kerosene_rate
## heating_fuel_coal_coke_rate               **
## heating_fuel_other_rate                   *
## heating_fuel_none_rate                    .
## electricity_price_transportation          ***
## electricity_consume_industrial            ***
## household_count                           ***
## average_household_size                    ***
## elevation
## cooling_design_temperature                ***
## earth_temperature_amplitude               ***
## relative_humidity                         ***
## atmospheric_pressure
## wind_speed
## cooling_degree_days                       ***
## occupation_construction_rate
## occupation_public_rate                    ***
## occupation_information_rate
## occupation_finance_rate                   **
## occupation_education_rate
## occupation_administrative_rate            **
## occupation_manufacturing_rate             **
## occupation_wholesale_rate
## occupation_retail_rate
## occupation_transportation_rate
## occupation_arts_rate                      *
```

```
## occupation_agriculture_rate                **
## occupancy_vacant_rate                      ***
## voting_2012_gop_percentage
## stateca                                    ***
## stateil                                    ***
## statemi                                    ***
## statenj                                    ***
## stateny
## statetx
## voting_2016_dem_winTrue                    **
## voting_2012_dem_winTrue                    ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 21518.7  on 15551  degrees of freedom
## Residual deviance:  8689.9  on 15495  degrees of freedom
## AIC: 8803.9
##
## Number of Fisher Scoring iterations: 6
```

```
trainpred<-predict(lgmmodel)
lgmpred<-predict(lgmmodel,newdata = testing,type = "response")
```

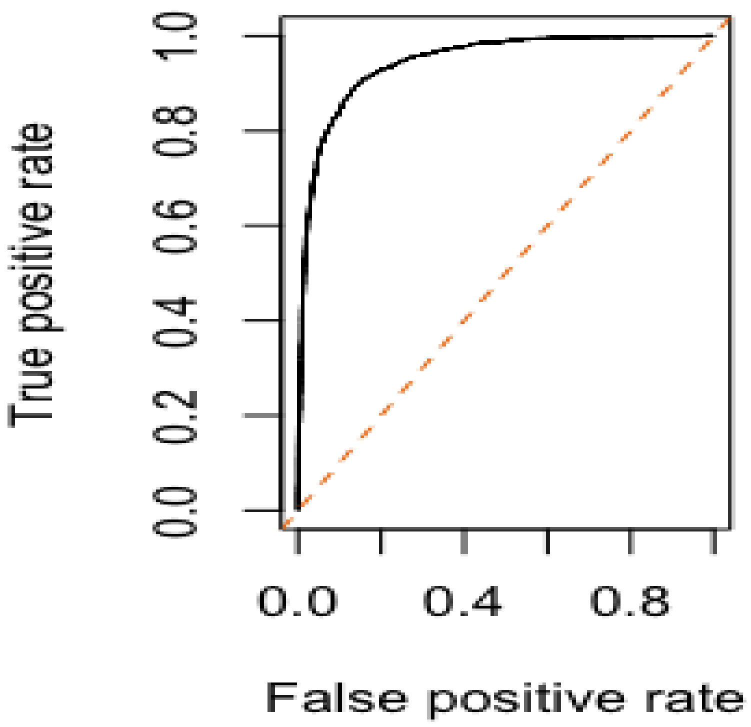
```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from a rank-deficient fit may be misleading
```

```
#Plot of Fitted Probability Vs Log Odds
par(mfrow=c(1,2))
symb<- c(15,18)
col<- c("red","blue")
#For the training data
plot(trainpred , jitter(trainpred,amount = 0.1), pch=symb[training$solar_system_count]
, col= adjustcolor(col[training$solar_system_count],0.7), cex=0.7, xlab = "Log-odds",y
lab = "Fitted probabilities")
#for the testing data
plot(lgmpred , jitter(lgmpred,amount = 0.1), pch=symb[testing$solar_system_count], col
= adjustcolor(col[testing$solar_system_count],0.7), cex=0.7, xlab = "Log-odds",ylab =
"Fitted probabilities")
```




```
#Check the accuracy of the training model
tau<- 0.5
p<- fitted(lgmmodel)
predlgm<-ifelse(p> tau,1,0)
tabl<-table(training$solar_system_count,predlgm)
glmaccr<-sum(diag(tabl)/sum(tabl))
#Accuracy of the training model
glmaccr

## [1] 0.8868313

library(ROCR)

## Loading required package: gplots

##
## Attaching package: 'gplots'

## The following object is masked from 'package:stats':
##
##     lowess

# Plot the ROC Curve for the testing predicted data
predObj <-prediction(lgmpred, testing$solar_system_count)
 roc<-performance(predObj,"tpr","fpr")
plot(roc)
abline(0,1,col ="darkorange2",lty =2)
auc<-performance(predObj,"auc")
#Area under the ROC curve
auc@y.values

## [[1]]
## [1] 0.9460358
```

```
#The accuracy under the ROC curve is 95.14%
```



Decision Tree

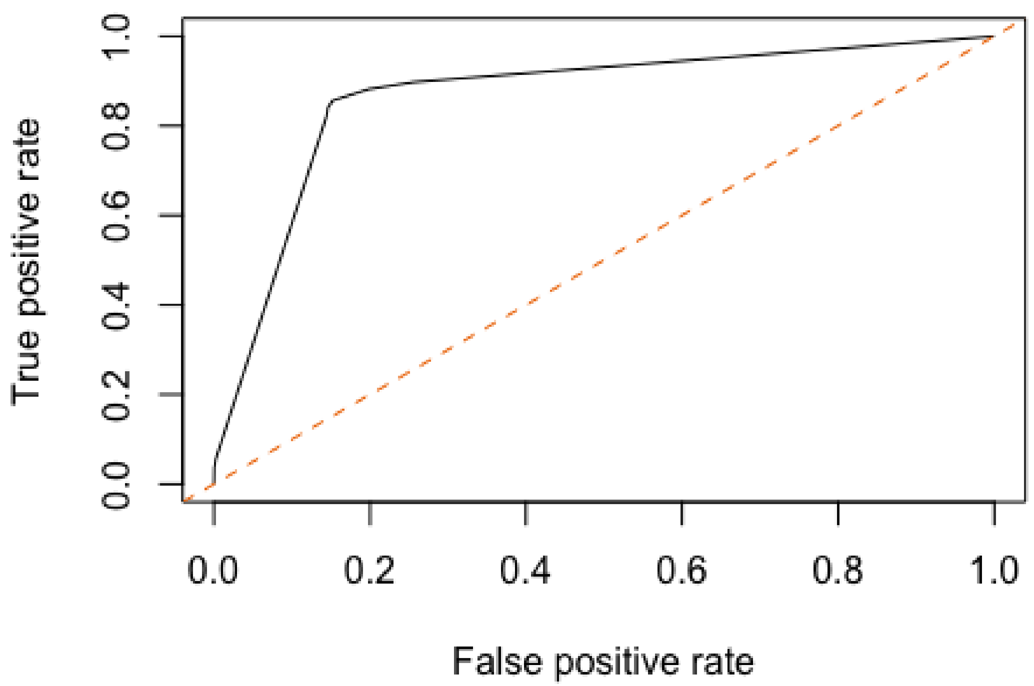
```
#Load the Library

library(rpart)
library(partykit)

library(ROCR)

## Loading required package: grid

## Loading required package: libcoin

## Loading required package: mvtnorm

#Build the model

dtmodel<-rpart(solar_system_count~.,data=training)
summary(dtmodel)

## Call:
## rpart(formula = solar_system_count ~ ., data = training)
##   n= 15552
##
##           CP nsplit rel error    xerror        xstd
## 1 0.63006642      0 1.0000000 1.0000000 0.008441330
## 2 0.01904568      1 0.3699336 0.3699336 0.006430188
## 3 0.01314898      3 0.3318422 0.3292666 0.006137018
## 4 0.01233564      4 0.3186932 0.3219466 0.006080890
## 5 0.01009896      5 0.3063576 0.3172021 0.006043928
## 6 0.01000000      7 0.2861597 0.3068998 0.005962043
##
## Variable importance
##                                state                   relative_humidity
```

```
##                                       20                                  19
##      electricity_consume_industrial    electricity_price_transportation
##                                       18                                  11
##                 atmospheric_pressure                           elevation
##                                       11                                  11
##                           total_area                  population_density
##                                        4                                   3
## heating_fuel_fuel_oil_kerosene_rate          voting_2012_gop_percentage
##                                        2                                   1
##
```

```
#Predict for the testing data

preddt<-predict(dtmodel,newdata =testing,type = "class")

#Create a cross table
dttble<-table(preddt,testing$solar_system_count)

#get the accuracy
dtacrcy<-(sum(diag(dttble)))/sum(dttble)
dtacrcy
```

```
## [1] 0.8518519
```

```
#Plot the ROC Curve for the model
phat<-predict(dtmodel,newdata = testing)
predObj <-prediction(phat[,2], testing$solar_system_count)
 roc<-performance(predObj,"tpr","fpr")
plot(roc)
abline(0,1,col ="darkorange2",lty =2)
```

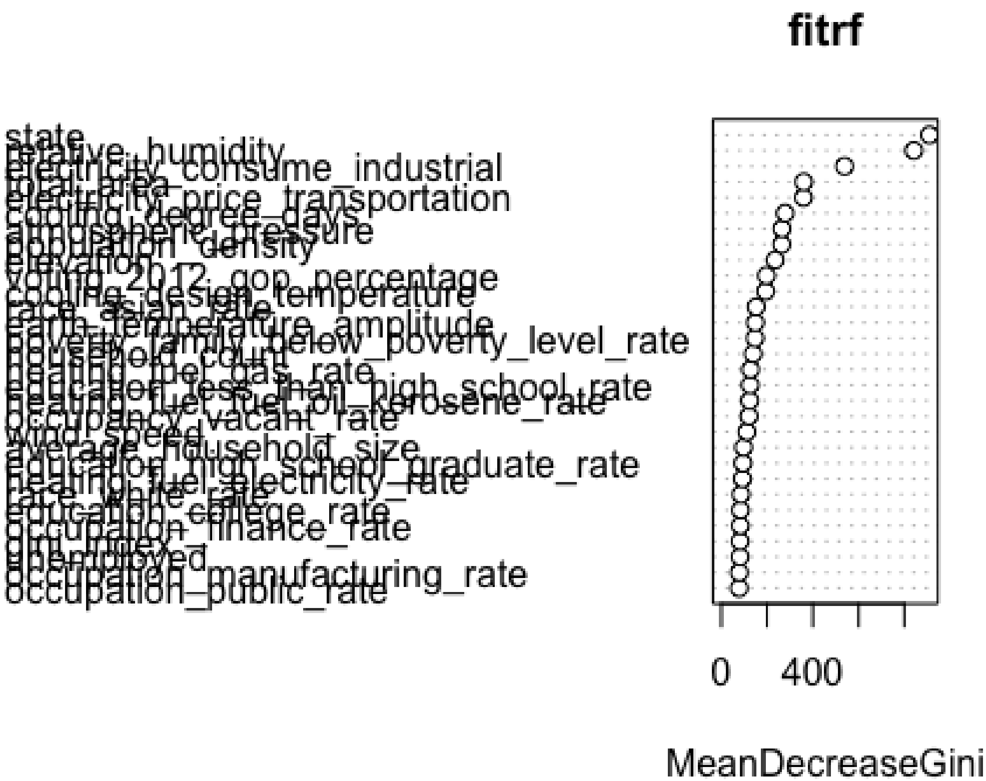



```
auc<-performance(predObj,"auc")
auc@y.values
```

```
## [[1]]
## [1] 0.8662943

#Accuracy is 86.62% under the ROC curve
```

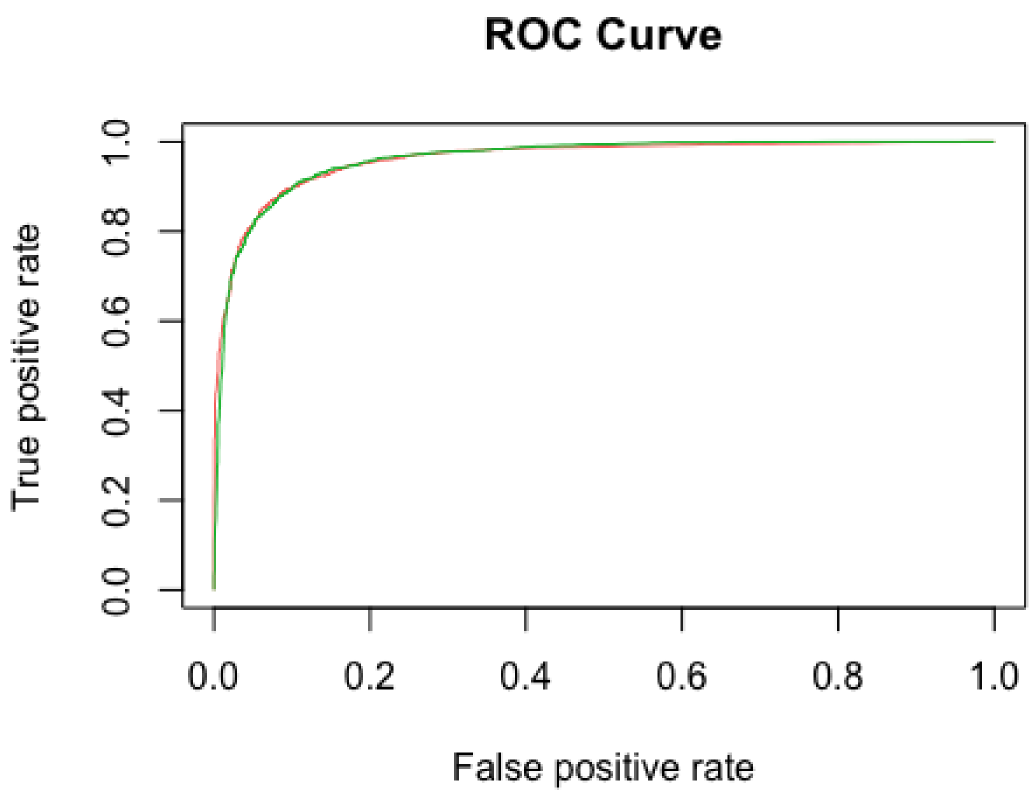
Random Forest

```
library(randomForest)

## randomForest 4.6-14

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:ggplot2':
##
##     margin

# implement the random forest algorithm oon the training data
fitrf <-randomForest(solar_system_count~.,data =training)

varImpPlot(fitrf)
```




```
#Calculate the accuracy of the training model no the testing data
rdpred <-predict(fitrf,newdata=testing,type="class",importance=TRUE)
rftable<-table(rdpred,testing$solar_system_count)
rfacc<-(sum(diag(rftable)))/sum(rftable)

#Accuracy is 90.58%
```

```
colours <- c("#F8766D","#00BA38")
#Predict the response label for the testing data and Plot the ROC Curve

rdpred <-predict(fitrf,newdata=testing,type="prob",importance=TRUE)
classes <- levels(testing$solar_system_count)
# For each class
for (i in 1:2)
{
 # Define which observations belong to class[i]
 true_values <- ifelse(testing[,54]==classes[i],1,0)
 # Assess the performance of classifier for class[i]
 pred <- prediction(rdpred[,i],true_values)
 perf <- performance(pred, "tpr", "fpr")
 if (i==1)
 {
     plot(perf,main="ROC Curve",col=colours[i])
 }
 else
 {
     plot(perf,main="ROC Curve",col=colours[i],add=TRUE)
 }
 # Calculate the AUC and print it to screen
 auc.perf <- performance(pred, measure = "auc")
 print(auc.perf@y.values)
}
```




```
## [[1]]
## [1] 0.9619156
##
## [[1]]
## [1] 0.9619156

#The area under the curve is 96.79% each for both high and low
```

Bagging

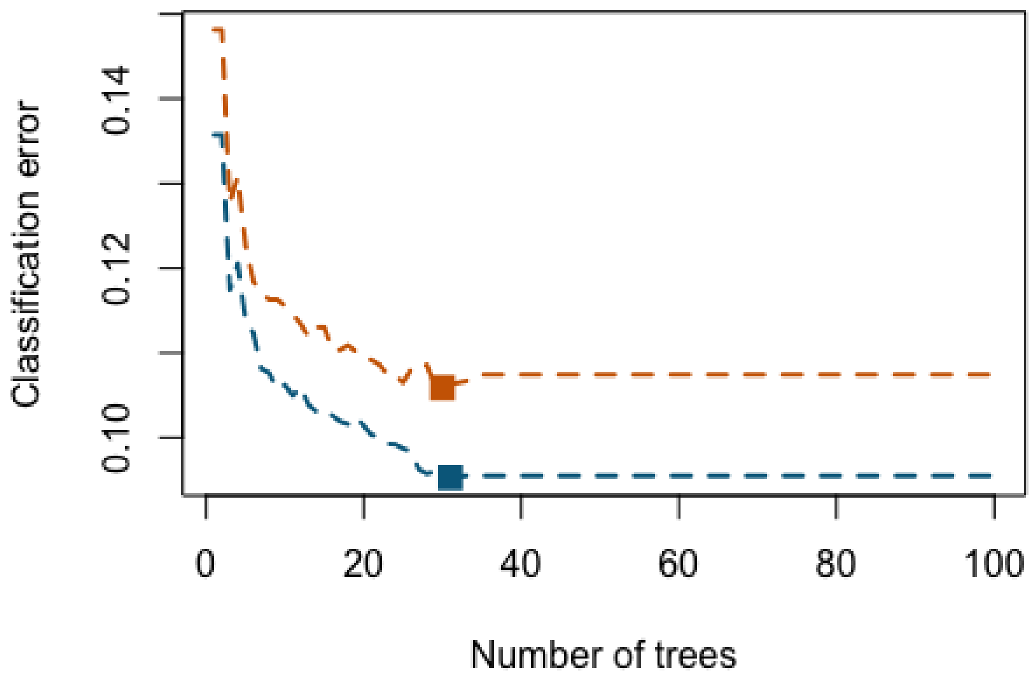
```r
#install.packages("adabag")

#Load the library
library(adabag)

## Loading required package: foreach

## Loading required package: doParallel

## Loading required package: iterators

## Loading required package: parallel

#Build a model for the training data

bagmodel<-bagging(solar_system_count~.,data = training)

#Predict the response label for the testing data
bagpred<-predict(bagmodel,newdata=testing,type="class")

#Confusion matrix
bagtb<-table(bagpred$class,testing$solar_system_count)

#get the accuracy
bagacc<-sum(diag(bagtb)/sum(bagtb))
bagacc
```

```
## [1] 0.8433642
```

Boosting

```r
#Build a model for the training data

boostmodel<-boosting(solar_system_count~.,data = training,coeflearn ="Breiman",boos =F
ALSE)

#Predict the response label for the testing data
boostpred<-predict(boostmodel,newdata=testing,type="class")

#Confusion matrix
boosttb<-table(boostpred$class,testing$solar_system_count)

#get the accuracy
bagacc<-sum(diag(boosttb)/sum(boosttb))

#error rate across each class
boostpred[c("confusion","error")]

## $confusion
##                Observed Class
## Predicted Class high  low
##             high 2456  288
##             low   269 2171
##
## $error
## [1] 0.107446

#Accuracy of 88.59%
#error of 0.114
eBoostTrain <-errorevol(boostmodel, training)$error
eboosttest<-errorevol(boostmodel,testing)$error
```

```
#classification error plot
mat <-cbind(eBoostTrain, eboosttest)

cols <-c("deepskyblue4","darkorange3")

matplot(mat,type ="l",lty =rep(2:1,each =2),col =cols,lwd =2,xlab ="Number of trees",y
lab ="Classification error")

legend(x =80,y =0.08,cex =0.75,legend =c("Boosting train","Boosting test"),col =cols,l
wd =2,bty ="n",lty = 2)

points(apply(mat,2, which.min),apply(mat,2, min),col =cols,pch =rep(c(15,17),each =2),
cex =1.5)
```



Performance of 4 models evaluated

#Load the libraries

```
library(nnet)
library(kernlab)

##
## Attaching package: 'kernlab'

## The following object is masked from 'package:ggplot2':
##
##     alpha

library(rpart)
library(randomForest)

## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:ggplot2':
##
##     margin

library(adabag)

## Loading required package: foreach

## Loading required package: doParallel

## Loading required package: iterators

## Loading required package: parallel

R<-50
out <-matrix(NA, R,7)

colnames(out) <-c("val_class_tree","val_logistic","val_svm","val_bag","val_randomfores
t","best","test")

out <-as.data.frame(out)


for (r in 1:R)
 {
 N<- nrow(temp_data)
 train <-sample(1:N,size =0.50*N)
 val <-sample(setdiff(1:N, train),size =0.25*N )
 test <-setdiff(1:N,union(train, val))

 #fit the classifiers only to training data

 #logistic regression
fitlog<- multinom(solar_system_count~.,data = temp_data[train,],trace=FALSE)

 #Classification tree
 fitct<-rpart(solar_system_count~.,data = temp_data[train,])

 #SVM
fitsvm<-ksvm(solar_system_count~.,data = temp_data[train,])
 #Bagging
fitbag<-bagging(solar_system_count~.,data = temp_data[train,])

#random Forest

fitRF<-randomForest(solar_system_count~.,data = temp_data[train,])



#fit on validation data
#Classification tree
predValCt <-predict(fitct,type ="class",newdata =temp_data[val,])
tabValCt <-table(temp_data$solar_system_count[val], predValCt)
accCt <-sum(diag(tabValCt))/sum(tabValCt)
```

```r
#logistic regression

predValLog <-predict(fitlog,type ="class",newdata =temp_data[val,])
tabValLog <-table(temp_data$solar_system_count[val], predValLog)
accLog <-sum(diag(tabValLog))/sum(tabValLog)


#SVM
predValSvm <-predict(fitsvm,newdata =temp_data[val,])
tabValSvm <-table(temp_data$solar_system_count[val], predValSvm)
accSvm <-sum(diag(tabValSvm))/sum(tabValSvm)

#bagging
predValBag <-predict(fitbag,newdata =temp_data[val,])
tabValBag <-table(temp_data$solar_system_count[val], predValBag$class)
accBag <-sum(diag(tabValBag))/sum(tabValBag)

#RF
predValRF <-predict(fitRF,newdata =temp_data[val,],type="class",importance=TRUE)
tabValRF <-table(temp_data$solar_system_count[val], predValRF)
accRF<-sum(diag(tabValRF))/sum(tabValRF)

 #store the accuracy and check the best
acc <-c(class_tree =accCt,logistic =accLog,svm =accSvm,bag=accBag,rf=accRF)
out[r,1] <-accCt
out[r,2] <-accLog
out[r,3] <-accSvm
out[r,4]<-accBag
out[r,5]<-accRF

best <-names(which.max(acc) )

#The model is tested for testing data and accuracy is calculated and the best is recor
ded
switch (best,
  class_tree ={
  predTestCt <-predict(fitct,type ="class",newdata=temp_data[test,])
  tabTestCt <-table(temp_data$solar_system_count[test], predTestCt)
  accBest <-sum(diag(tabTestCt))/sum(tabTestCt)
  },
  logistic =
    {
      predTestLog <-predict(fitlog,type ="class",newdata =temp_data[test,])
    tabTestLog <-table(temp_data$solar_system_count[test], predTestLog)
    accBest <-sum(diag(tabTestLog))/sum(tabTestLog)
    },
  svm ={
    predTestSvm <-predict(fitsvm,newdata =temp_data[test,])
  tabTestSvm <-table(temp_data$solar_system_count[test], predTestSvm)
  accBest <-sum(diag(tabTestSvm))/sum(tabTestSvm)
  },
   bag ={
    predTestBag <-predict(fitbag,newdata =temp_data[test,])
  tabTestBag <-table(temp_data$solar_system_count[test], predTestBag$class)
  accBest <-sum(diag(tabTestBag))/sum(tabTestBag)
  },
   rf ={
    predTestRF <-predict(fitRF,newdata =temp_data[test,],type="class",importance=TRUE)
  tabTestRF <-table(temp_data$solar_system_count[test], predTestRF)
```

```
  accBest <-sum(diag(tabTestRF))/sum(tabTestRF)
  }

)
#Best accuracy is stored
out[r,6] <-best
out[r,7] <-accBest

}

#% of model turned to be best among the models
table(out[,6])/R

##
##   rf svm
## 0.85 0.15

tapply(out[,7], out[,6], summary)

## $random forest
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.8947  0.8951  0.8998  0.8989  0.9014  0.9072
##
## $svm
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.8961  0.8968  0.8974  0.8976  0.8982  0.8991

#Box plot
boxplot(out$test~out$best)
stripchart(out$test~out$best,add =TRUE,vertical =TRUE,method ="jitter",pch =19,col =ad
justcolor("magenta3",0.2))
```