ABI Group Work 2

Loading the dataset to R

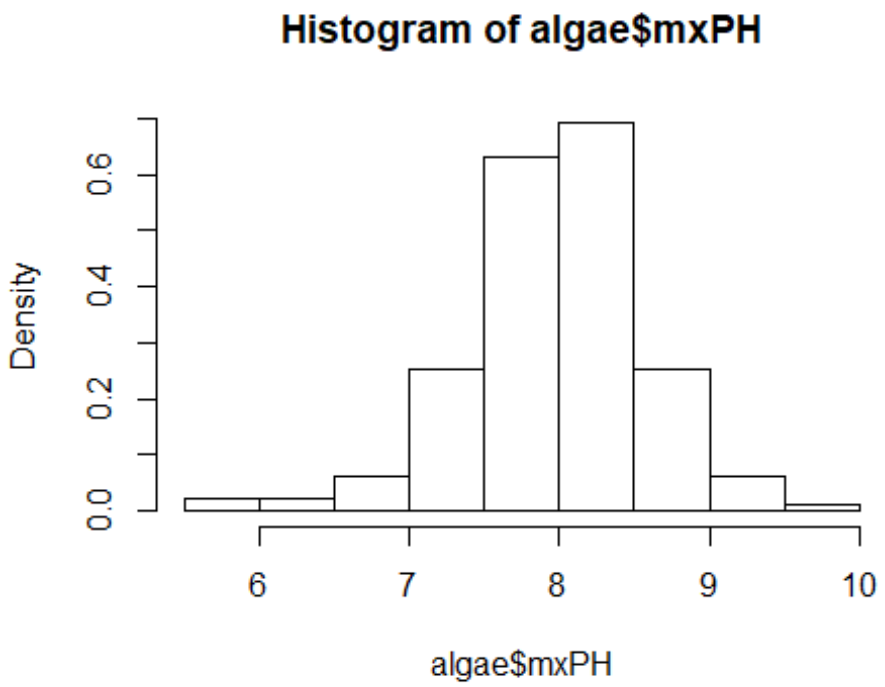
```
#Installing the required packages
#install.packages("DMwR")

#Loading the required library
library(DMwR)

## Loading required package: lattice

## Loading required package: grid

#Printing the head of the dataset
head(algae)

##   season  size  speed mxPH mnO2     Cl    NO3     NH4    oPO4     PO4 Chla
## 1 winter small medium 8.00  9.8 60.800  6.238 578.000 105.000 170.000 50.0
## 2 spring small medium 8.35  8.0 57.750  1.288 370.000 428.750 558.750  1.3
## 3 autumn small medium 8.10 11.4 40.020  5.330 346.667 125.667 187.057 15.6
## 4 spring small medium 8.07  4.8 77.364  2.302  98.182  61.182 138.700  1.4
## 5 autumn small medium 8.06  9.0 55.350 10.416 233.700  58.222  97.580 10.5
## 6 winter small   high 8.25 13.1 65.750  9.248 430.000  18.250  56.667 28.4
##     a1   a2   a3  a4   a5   a6  a7
## 1  0.0  0.0  0.0 0.0 34.2  8.3 0.0
## 2  1.4  7.6  4.8 1.9  6.7  0.0 2.1
## 3  3.3 53.6  1.9 0.0  0.0  0.0 9.7
## 4  3.1 41.0 18.9 0.0  1.4  0.0 1.4
## 5  9.2  2.9  7.5 0.0  7.5  4.1 1.0
## 6 15.1 14.6  1.4 0.0 22.5 12.6 2.9

#Printing the Summary of the dataset
summary(algae)

##     season       size       speed         mxPH            mnO2
##  autumn:40   large :45   high  :84   Min.   :5.600   Min.   : 1.500
##  spring:53   medium:84   low   :33   1st Qu.:7.700   1st Qu.: 7.725
##  summer:45   small :71   medium:83   Median :8.060   Median : 9.800
##  winter:62                           Mean   :8.012   Mean   : 9.118
##                                      3rd Qu.:8.400   3rd Qu.:10.800
##                                      Max.   :9.700   Max.   :13.400
##                                      NA's   :1       NA's   :2
##        Cl               NO3              NH4                oPO4
##  Min.   :  0.222   Min.   : 0.050   Min.   :    5.00   Min.   :  1.00
```

```
##  1st Qu.: 10.981   1st Qu.: 1.296   1st Qu.:   38.33   1st Qu.: 15.70
##  Median : 32.730   Median : 2.675   Median :  103.17   Median : 40.15
##  Mean   : 43.636   Mean   : 3.282   Mean   :  501.30   Mean   : 73.59
##  3rd Qu.: 57.824   3rd Qu.: 4.446   3rd Qu.:  226.95   3rd Qu.: 99.33
##  Max.   :391.500   Max.   :45.650   Max.   :24064.00   Max.   :564.60
##  NA's   :10        NA's   :2        NA's   :2          NA's   :2
##       PO4              Chla               a1              a2
##  Min.   :  1.00   Min.   :  0.200   Min.   : 0.00   Min.   : 0.000
##  1st Qu.: 41.38   1st Qu.:  2.000   1st Qu.: 1.50   1st Qu.: 0.000
##  Median :103.29   Median :  5.475   Median : 6.95   Median : 3.000
##  Mean   :137.88   Mean   : 13.971   Mean   :16.92   Mean   : 7.458
##  3rd Qu.:213.75   3rd Qu.: 18.308   3rd Qu.:24.80   3rd Qu.:11.375
##  Max.   :771.60   Max.   :110.456   Max.   :89.80   Max.   :72.600
##  NA's   :2        NA's   :12
##        a3               a4               a5               a6
##  Min.   : 0.000   Min.   : 0.000   Min.   : 0.000   Min.   : 0.000
##  1st Qu.: 0.000   1st Qu.: 0.000   1st Qu.: 0.000   1st Qu.: 0.000
##  Median : 1.550   Median : 0.000   Median : 1.900   Median : 0.000
##  Mean   : 4.309   Mean   : 1.992   Mean   : 5.064   Mean   : 5.964
##  3rd Qu.: 4.925   3rd Qu.: 2.400   3rd Qu.: 7.500   3rd Qu.: 6.925
##  Max.   :42.800   Max.   :44.600   Max.   :44.400   Max.   :77.600
##
##        a7
##  Min.   : 0.000
##  1st Qu.: 0.000
##  Median : 1.000
##  Mean   : 2.495
##  3rd Qu.: 2.400
##  Max.   :31.600
##

#View(algae)
```

By looking at the above summary we can see that most of the data is of Winter and for numeric values we get to know the number of NA values i.e missing values and the spread of data with the help of 5 distinct values of min 1st quadrant median 3rd quadrant.

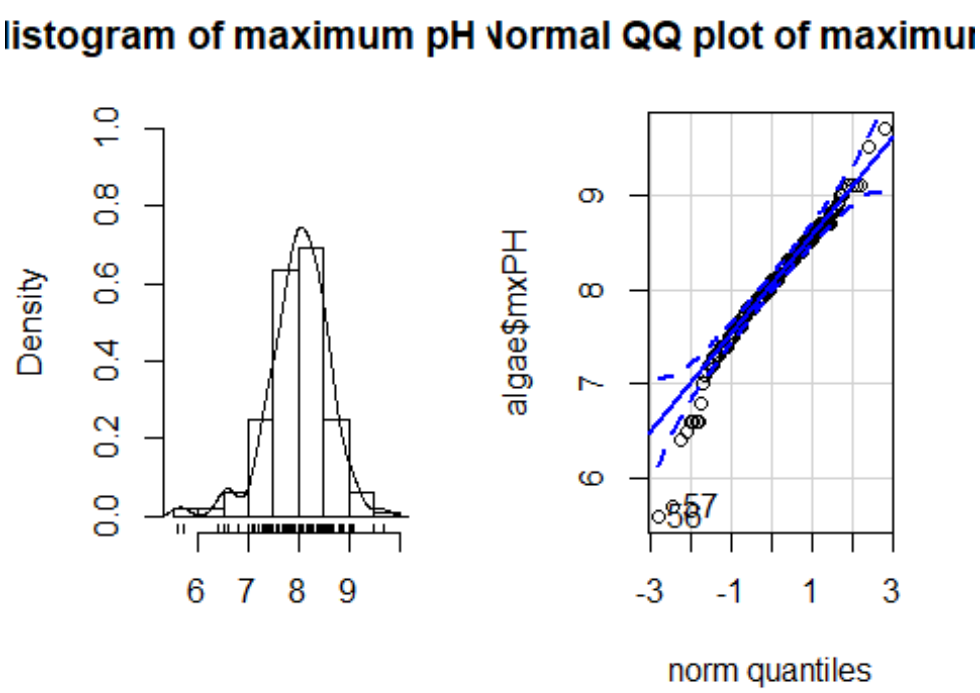
We would get better idea of the numerical data with the help of graphical visualization

```
#Plots the histogram of the mxPH column of the dataframe with probability
values
hist(algae$mxPH, prob = T)
```



The above figure gives us an idea that the values follow a distribution very near the normal distribution, with the values very nicely clustered around the mean value.

```
#Loading the required libraries
library(car)

## Loading required package: carData

#Setting the plotting parameters
par(mfrow=c(1,2))

#Histogram plot with Density and Rug plot to show the distribution
hist(algae$mxPH, prob=T, xlab='', main='Histogram of maximum pH
value',ylim=0:1)
lines(density(algae$mxPH,na.rm=T))
rug(jitter(algae$mxPH))

#Normal qq plot of mxPH
qqPlot(algae$mxPH,main='Normal QQ plot of maximum pH')
```

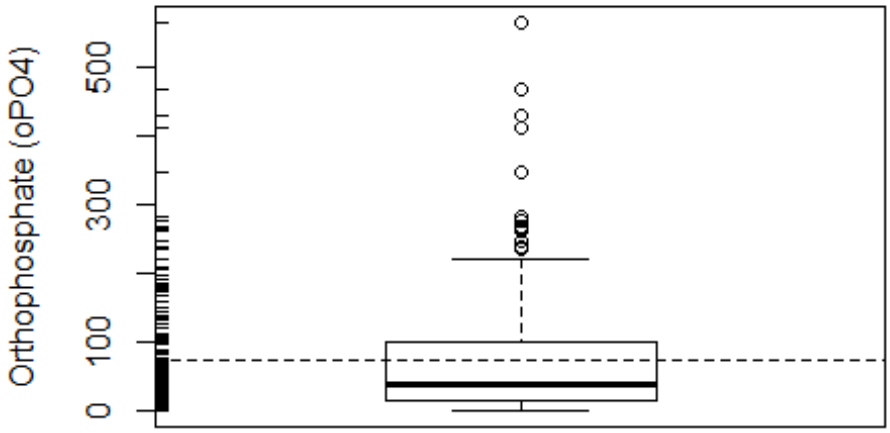



```
## [1] 56 57

par(mfrow=c(1,1))
```

The above figure states that there are some values in the normal qq plot which says that some points clearly breakes 95% confidence interval.

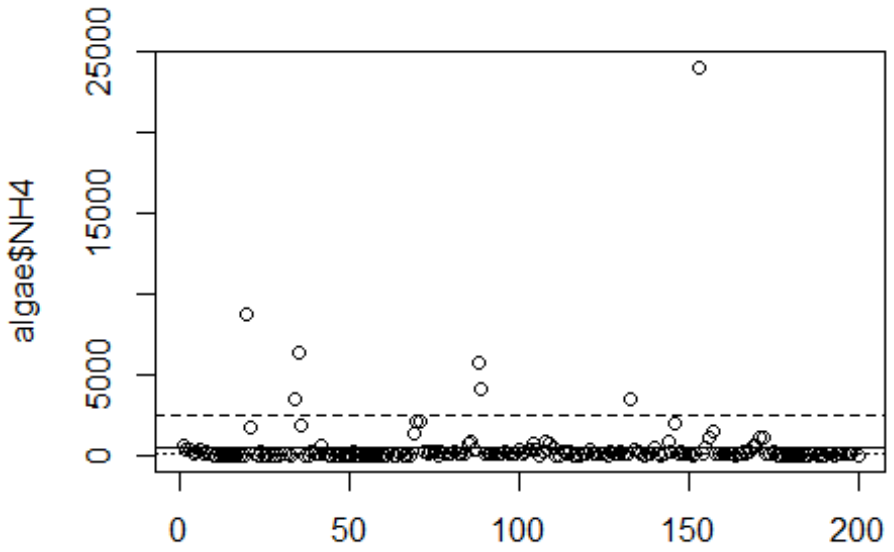
Now we plot a boxplot of oPo4 to get the idea of the dataset

```
#Shows the boxplot of oPO4
boxplot(algae$oPO4, ylab = "Orthophosphate (oPO4)")
rug(jitter(algae$oPO4), side = 2)
abline(h = mean(algae$oPO4, na.rm = T), lty = 2)
```



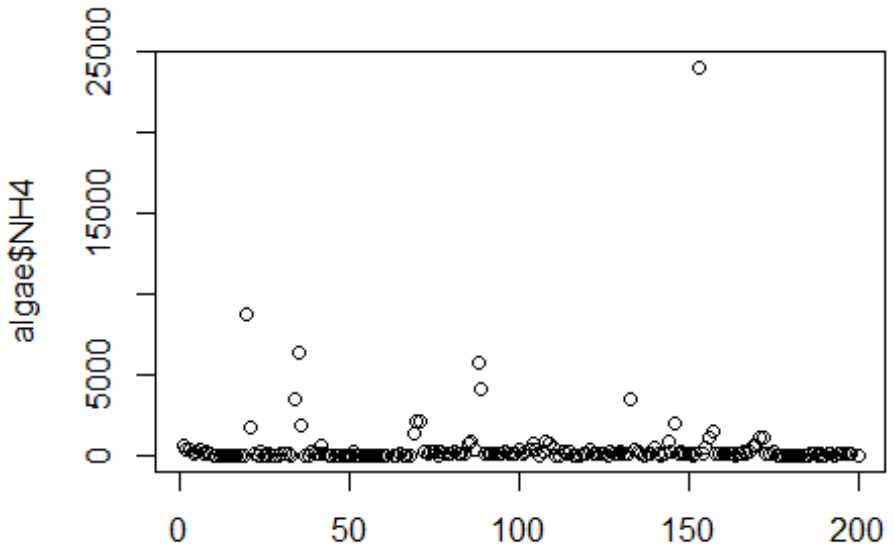
Below we show a boxplot of NH4

```
#Shows a scatterplot of algae$NH4
plot(algae$NH4, xlab = "")
abline(h = mean(algae$NH4, na.rm = T), lty = 1)
abline(h = mean(algae$NH4, na.rm = T) + sd(algae$NH4, na.rm = T),lty = 2)
abline(h = median(algae$NH4, na.rm = T), lty = 3)
identify(algae$NH4)
```




```
## integer(0)
```

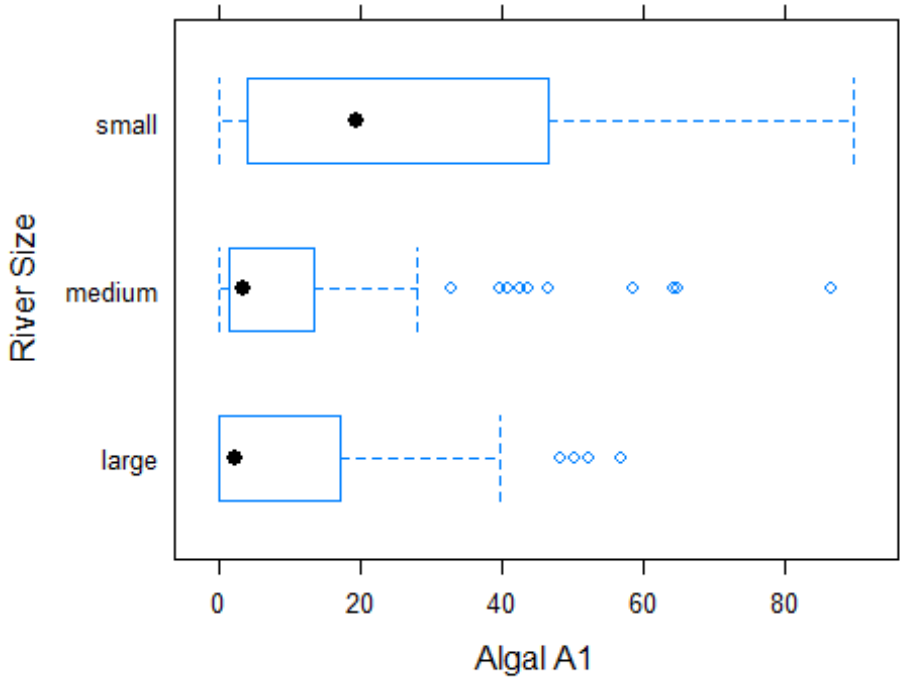
```
plot(algae$NH4, xlab = "")
clicked.lines <- identify(algae$NH4)
```




```
algae[clicked.lines, ]
```

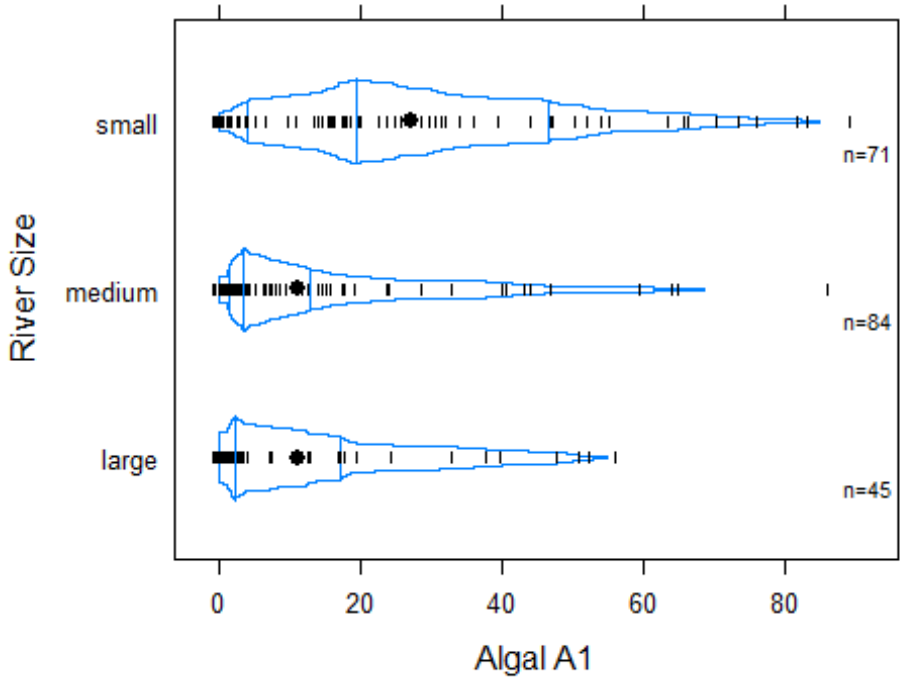
```
##  [1] season size   speed  mxPH   mnO2   Cl     NO3    NH4    oPO4   PO4
## [11] Chla   a1     a2     a3     a4     a5     a6     a7
## <0 rows> (or 0-length row.names)
```

```
#Importing the required libraries
library(lattice)

#Shows a boxplot of algae1 with respect to size of river
bwplot(size ~ a1, data=algae, ylab='River Size',xlab='Algal A1')
```



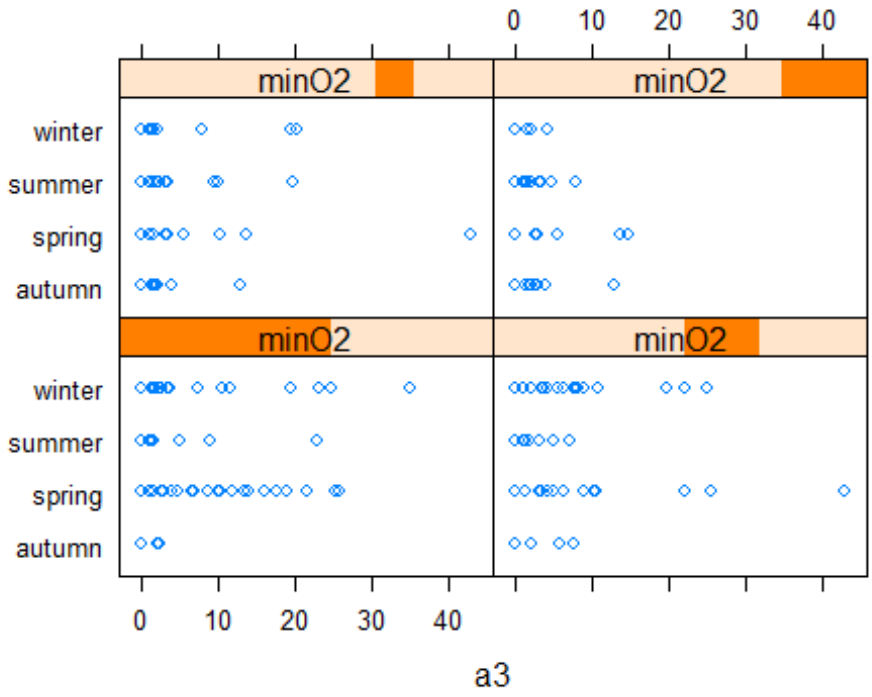
The above figure allows us to observe that higher frequencies of algal a1 are expected in smaller rivers, which can be valuable knowledge.

```
#Install required packages
#install.packages("Hmisc")
library(Hmisc)

## Loading required package: survival

## Loading required package: Formula

## Loading required package: ggplot2

##
## Attaching package: 'Hmisc'

## The following objects are masked from 'package:base':
##
##     format.pval, units

bwplot(size ~ a1, data=algae,panel=panel.bpplot, probs=seq(.01,.49,by=.01),
datadensity=TRUE, ylab='River Size',xlab='Algal A1')
```



The dots are the mean value of the frequency of the algal for the di???erent river sizes. Vertical lines represent the 1st quartile, median, and 3rd quartile, in that order. The graphs show us the actual values of the data with small dashes, and the information of the distribution of these values is provided by the quantile plots.

```
#Converts the continuous variable to create factorized version
minO2 <- equal.count(na.omit(algae$mnO2),number=4,overlap=1/5)

#Plots a strip plot
stripplot(season ~ a3|minO2,data=algae[!is.na(algae$mnO2),])
```



Removing the Observations with Unknown Values

```
#Prints all the rows with atleast 1 NA value
algae[!complete.cases(algae),]
```

```
##     season   size  speed mxPH mnO2    Cl   NO3 NH4    oPO4     PO4  Chla
## 28  autumn  small   high 6.80 11.1 9.000 0.630  20   4.000      NA  2.70
## 38  spring  small   high 8.00   NA 1.450 0.810  10   2.500   3.000  0.30
## 48  winter  small    low   NA 12.6 9.000 0.230  10   5.000   6.000  1.10
## 55  winter  small   high 6.60 10.8    NA 3.245  10   1.000   6.500    NA
## 56  spring  small medium 5.60 11.8    NA 2.220   5   1.000   1.000    NA
## 57  autumn  small medium 5.70 10.8    NA 2.550  10   1.000   4.000    NA
## 58  spring  small   high 6.60  9.5    NA 1.320  20   1.000   6.000    NA
## 59  summer  small   high 6.60 10.8    NA 2.640  10   2.000  11.000    NA
## 60  autumn  small medium 6.60 11.3    NA 4.170  10   1.000   6.000    NA
## 61  spring  small medium 6.50 10.4    NA 5.970  10   2.000  14.000    NA
## 62  summer  small medium 6.40   NA    NA    NA  NA      NA  14.000    NA
## 63  autumn  small   high 7.83 11.7 4.083 1.328  18   3.333   6.667    NA
## 116 winter medium   high 9.70 10.8 0.222 0.406  10  22.444  10.111    NA
## 161 spring  large    low 9.00  5.8    NA 0.900 142 102.000 186.000 68.05
## 184 winter  large   high 8.00 10.9 9.055 0.825  40  21.083  56.091    NA
## 199 winter  large medium 8.00  7.6    NA    NA  NA      NA      NA    NA
##       a1   a2  a3   a4  a5  a6  a7
## 28  30.3  1.9 0.0  0.0 2.1 1.4 2.1
## 38  75.8  0.0 0.0  0.0 0.0 0.0 0.0
## 48  35.5  0.0 0.0  0.0 0.0 0.0 0.0
## 55  24.3  0.0 0.0  0.0 0.0 0.0 0.0
```

```
## 56  82.7  0.0 0.0  0.0 0.0 0.0 0.0
## 57  16.8  4.6 3.9 11.5 0.0 0.0 0.0
## 58  46.8  0.0 0.0 28.8 0.0 0.0 0.0
## 59  46.9  0.0 0.0 13.4 0.0 0.0 0.0
## 60  47.1  0.0 0.0  0.0 0.0 1.2 0.0
## 61  66.9  0.0 0.0  0.0 0.0 0.0 0.0
## 62  19.4  0.0 0.0  2.0 0.0 3.9 1.7
## 63  14.4  0.0 0.0  0.0 0.0 0.0 0.0
## 116 41.0  1.5 0.0  0.0 0.0 0.0 0.0
## 161  1.7 20.6 1.5  2.2 0.0 0.0 0.0
## 184 16.8 19.6 4.0  0.0 0.0 0.0 0.0
## 199  0.0 12.5 3.7  1.0 0.0 0.0 4.9
```

```
#Prints the number of rows that has atleast 1 NA value
nrow(algae[!complete.cases(algae),])
```

```
## [1] 16
```

```
#Filters the data and removes all the rows that has one or more NA values in
them and stores the new data in variable algae
algae <- na.omit(algae)
```

Alternate method of dealing with NA values. Since we already have filtered the data we need to read the new data again

```
#Loading the new data again
data(algae)

#printing the structure of the new data
str(algae)
```

```
## 'data.frame':    200 obs. of  18 variables:
##  $ season: Factor w/ 4 levels "autumn","spring",..: 4 2 1 2 1 4 3 1 4 4
...
##  $ size  : Factor w/ 3 levels "large","medium",..: 3 3 3 3 3 3 3 3 3 3 ...
##  $ speed : Factor w/ 3 levels "high","low","medium": 3 3 3 3 3 1 1 1 3 1
...
##  $ mxPH  : num  8 8.35 8.1 8.07 8.06 8.25 8.15 8.05 8.7 7.93 ...
##  $ mnO2  : num  9.8 8 11.4 4.8 9 13.1 10.3 10.6 3.4 9.9 ...
##  $ Cl    : num  60.8 57.8 40 77.4 55.4 ...
##  $ NO3   : num  6.24 1.29 5.33 2.3 10.42 ...
##  $ NH4   : num  578 370 346.7 98.2 233.7 ...
##  $ oPO4  : num  105 428.8 125.7 61.2 58.2 ...
##  $ PO4   : num  170 558.8 187.1 138.7 97.6 ...
##  $ Chla  : num  50 1.3 15.6 1.4 10.5 ...
##  $ a1    : num  0 1.4 3.3 3.1 9.2 15.1 2.4 18.2 25.4 17 ...
##  $ a2    : num  0 7.6 53.6 41 2.9 14.6 1.2 1.6 5.4 0 ...
##  $ a3    : num  0 4.8 1.9 18.9 7.5 1.4 3.2 0 2.5 0 ...
##  $ a4    : num  0 1.9 0 0 0 0 3.9 0 0 2.9 ...
##  $ a5    : num  34.2 6.7 0 1.4 7.5 22.5 5.8 5.5 0 0 ...
```

```
##  $ a6    : num  8.3 0 0 0 4.1 12.6 6.8 8.7 0 0 ...
##  $ a7    : num  0 2.1 9.7 1.4 1 2.9 0 0 0 1.7 ...
```

```
#Prints all the rows with atleast 1 NA value
algae[!complete.cases(algae),]
```

```
##     season   size  speed mxPH mnO2    Cl   NO3 NH4    oPO4     PO4  Chla
## 28  autumn  small   high 6.80 11.1 9.000 0.630  20   4.000      NA  2.70
## 38  spring  small   high 8.00   NA 1.450 0.810  10   2.500   3.000  0.30
## 48  winter  small    low   NA 12.6 9.000 0.230  10   5.000   6.000  1.10
## 55  winter  small   high 6.60 10.8    NA 3.245  10   1.000   6.500    NA
## 56  spring  small medium 5.60 11.8    NA 2.220   5   1.000   1.000    NA
## 57  autumn  small medium 5.70 10.8    NA 2.550  10   1.000   4.000    NA
## 58  spring  small   high 6.60  9.5    NA 1.320  20   1.000   6.000    NA
## 59  summer  small   high 6.60 10.8    NA 2.640  10   2.000  11.000    NA
## 60  autumn  small medium 6.60 11.3    NA 4.170  10   1.000   6.000    NA
## 61  spring  small medium 6.50 10.4    NA 5.970  10   2.000  14.000    NA
## 62  summer  small medium 6.40   NA    NA    NA  NA      NA  14.000    NA
## 63  autumn  small   high 7.83 11.7 4.083 1.328  18   3.333   6.667    NA
## 116 winter medium   high 9.70 10.8 0.222 0.406  10  22.444  10.111    NA
## 161 spring  large    low 9.00  5.8    NA 0.900 142 102.000 186.000 68.05
## 184 winter  large   high 8.00 10.9 9.055 0.825  40  21.083  56.091    NA
## 199 winter  large medium 8.00  7.6    NA    NA  NA      NA      NA    NA
##       a1   a2  a3   a4  a5  a6  a7
## 28  30.3  1.9 0.0  0.0 2.1 1.4 2.1
## 38  75.8  0.0 0.0  0.0 0.0 0.0 0.0
## 48  35.5  0.0 0.0  0.0 0.0 0.0 0.0
## 55  24.3  0.0 0.0  0.0 0.0 0.0 0.0
## 56  82.7  0.0 0.0  0.0 0.0 0.0 0.0
## 57  16.8  4.6 3.9 11.5 0.0 0.0 0.0
## 58  46.8  0.0 0.0 28.8 0.0 0.0 0.0
## 59  46.9  0.0 0.0 13.4 0.0 0.0 0.0
## 60  47.1  0.0 0.0  0.0 0.0 1.2 0.0
## 61  66.9  0.0 0.0  0.0 0.0 0.0 0.0
## 62  19.4  0.0 0.0  2.0 0.0 3.9 1.7
## 63  14.4  0.0 0.0  0.0 0.0 0.0 0.0
## 116 41.0  1.5 0.0  0.0 0.0 0.0 0.0
## 161  1.7 20.6 1.5  2.2 0.0 0.0 0.0
## 184 16.8 19.6 4.0  0.0 0.0 0.0 0.0
## 199  0.0 12.5 3.7  1.0 0.0 0.0 4.9
```

By looking at the above results we can see that both the samples 62 and 199 have six of the eleven explanatory variables with unknown values. In such cases, it is wise to simply ignore these observations by removing them:

```
#Removes the row 62 and 199 and stores the entire rows in the variable algae
algae <- algae[-c(62, 199), ]

#Shows the number of NA variables per row of the dataframe
apply(algae, 1, function(x) sum(is.na(x)))
```

```
##   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18
##   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
##  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36
##   0   0   0   0   0   0   0   0   0   1   0   0   0   0   0   0   0   0
##  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54
##   0   1   0   0   0   0   0   0   0   0   0   1   0   0   0   0   0   0
##  55  56  57  58  59  60  61  63  64  65  66  67  68  69  70  71  72  73
##   2   2   2   2   2   2   2   1   0   0   0   0   0   0   0   0   0   0
##  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91
##   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
##  92  93  94  95  96  97  98  99 100 101 102 103 104 105 106 107 108 109
##   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
## 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127
##   0   0   0   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0
## 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145
##   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
## 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163
##   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   1   0   0
## 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180 181
##   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
## 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 200
##   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
```

we could have supplied the name of a "normal" function. The temporary function basically calculates the number of NAs on the object x, its argument. It takes advantage of the fact that a true value in R is equivalent to the number 1, and the false to the value 0, which means that when you sum a vector of Boolean values, you obtain how many trues exist in the vector. Based on this code we can create a function that gives us the rows in algae that have a certain number of unknowns. Such function is available in the book package and we can use it as follows:

```
#Loading the dataset in R
data(algae)

#Prints the NA values with more than 20% NA values in them
manyNAs(algae, 0.2)

## [1]  62 199

#Removing the multiple NA values and in this case we have used the default
value of the second argument of manyNAs(), which is 0.2
algae <- algae[-manyNAs(algae), ]

#Storing median value of algae$Chla where there are NA values
algae[is.na(algae$Chla), "Chla"] <- median(algae$Chla, na.rm = T)
```

Let us now implement data cleaning using centralImputation

```
#Loading the dataset
data(algae)
```

```r
#Filtering rows with more than 20% NA values
algae <- algae[-manyNAs(algae), ]

#Applying Central imputations to handle NA values in the dataset
algae <- centralImputation(algae)
```

Filling in the Unknown Values by Exploring Correlations

In this section we will be filtering the data based on Correlations

```r
#Prints the correlation matrix of variables 4 till 18
cor(algae[, 4:18], use = "complete.obs")
```

```
##             mxPH        mnO2          Cl         NO3         NH4
## mxPH  1.00000000 -0.16749178  0.13285681 -0.13103951 -0.09360612
## mnO2 -0.16749178  1.00000000 -0.27873229  0.09837676 -0.08780541
## Cl    0.13285681 -0.27873229  1.00000000  0.22504071  0.07407466
## NO3  -0.13103951  0.09837676  0.22504071  1.00000000  0.72144352
## NH4  -0.09360612 -0.08780541  0.07407466  0.72144352  1.00000000
## oPO4  0.15850785 -0.41655069  0.39230733  0.14458782  0.22723723
## PO4   0.18033494 -0.48772564  0.45652107  0.16931401  0.20844445
## Chla  0.39121495 -0.16678069  0.15082753  0.14290962  0.09375115
## a1   -0.26823725  0.28389830 -0.36078101 -0.24121109 -0.13265601
## a2    0.32584814 -0.09935631  0.08949837  0.02368832 -0.02968344
## a3    0.03077250 -0.25155437  0.09429722 -0.07621407 -0.10143974
## a4   -0.24876290 -0.31513753  0.12045912 -0.02578257  0.22822914
## a5   -0.01697947  0.17008979  0.16514900  0.22359794  0.02745909
## a6   -0.08388657  0.15864906  0.18369968  0.54640569  0.40571045
## a7   -0.08726106 -0.12117098 -0.02793640  0.08509789 -0.01672691
##             oPO4         PO4        Chla          a1          a2
## mxPH  0.15850785  0.18033494  0.39121495 -0.26823725  0.32584814
## mnO2 -0.41655069 -0.48772564 -0.16678069  0.28389830 -0.09935631
## Cl    0.39230733  0.45652107  0.15082753 -0.36078101  0.08949837
## NO3   0.14458782  0.16931401  0.14290962 -0.24121109  0.02368832
## NH4   0.22723723  0.20844445  0.09375115 -0.13265601 -0.02968344
## oPO4  1.00000000  0.91387767  0.12941615 -0.41735761  0.14768993
## PO4   0.91387767  1.00000000  0.26758873 -0.48730097  0.16246963
## Chla  0.12941615  0.26758873  1.00000000 -0.28380049  0.38192280
## a1   -0.41735761 -0.48730097 -0.28380049  1.00000000 -0.29251967
## a2    0.14768993  0.16246963  0.38192280 -0.29251967  1.00000000
## a3    0.03362906  0.06587312 -0.04975884 -0.14695028  0.03031095
## a4    0.29574585  0.30462623 -0.08364618 -0.03892441 -0.17168171
## a5    0.15147500  0.19111521 -0.05945318 -0.29503346 -0.16186215
## a6    0.02876159  0.08316987  0.01815732 -0.27602608 -0.11613061
## a7    0.04849832  0.10671057  0.02405581 -0.21142489  0.04749242
##               a3          a4          a5          a6          a7
## mxPH  0.03077250 -0.24876290 -0.01697947 -0.08388657 -0.08726106
## mnO2 -0.25155437 -0.31513753  0.17008979  0.15864906 -0.12117098
## Cl    0.09429722  0.12045912  0.16514900  0.18369968 -0.02793640
```

```
## NO3  -0.07621407 -0.02578257  0.22359794  0.54640569  0.08509789
## NH4  -0.10143974  0.22822914  0.02745909  0.40571045 -0.01672691
## oPO4  0.03362906  0.29574585  0.15147500  0.02876159  0.04849832
## PO4   0.06587312  0.30462623  0.19111521  0.08316987  0.10671057
## Chla -0.04975884 -0.08364618 -0.05945318  0.01815732  0.02405581
## a1   -0.14695028 -0.03892441 -0.29503346 -0.27602608 -0.21142489
## a2    0.03031095 -0.17168171 -0.16186215 -0.11613061  0.04749242
## a3    1.00000000  0.01218370 -0.11111997 -0.17283566  0.05618729
## a4    0.01218370  1.00000000 -0.11006558 -0.09074936  0.04362334
## a5   -0.11111997 -0.11006558  1.00000000  0.40360881 -0.02686306
## a6   -0.17283566 -0.09074936  0.40360881  1.00000000 -0.01244488
## a7    0.05618729  0.04362334 -0.02686306 -0.01244488  1.00000000
```

```
#Pass the function cor to symnum for better output
symnum(cor(algae[,4:18],use="complete.obs"))
```

```
##      mP mO Cl NO NH o P Ch a1 a2 a3 a4 a5 a6 a7
## mxPH 1
## mnO2    1
## Cl         1
## NO3           1
## NH4           ,  1
## oPO4    .  .        1
## PO4     .  .        * 1
## Chla .                   1
## a1          .        . .    1
## a2   .                    .     1
## a3                              1
## a4      .             .            1
## a5                                   1
## a6            .  .                   .  1
## a7                                        1
## attr(,"legend")
## [1] 0 ' ' 0.3 '.' 0.6 ',' 0.8 '+' 0.9 '*' 0.95 'B' 1
```

From the above output we can say that there is strong correlation between PO4 and oPO4. With respect to PO4 and oPO4, the discovery of this correlation allows us to fill in the unknowns on these variables. In order to achieve this we need to find the form of the linear correlation between these variables. This can be done as follows:

```
#Importing the data
data(algae)

#Eleminating the rows with more NA values
algae <- algae[-manyNAs(algae), ]

#Applying the linear model
lm(PO4 ~ oPO4, data = algae)
```

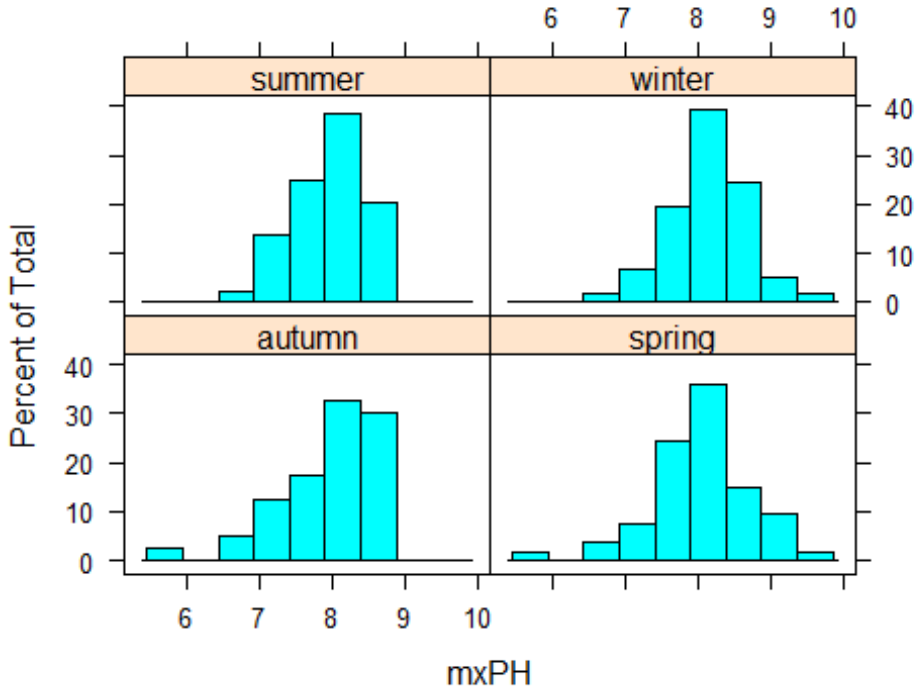
```
##
## Call:
## lm(formula = PO4 ~ oPO4, data = algae)
##
## Coefficients:
## (Intercept)         oPO4
##      42.897        1.293
```

Now let is fill the NA values for PO4 with the help of the coeifficients we found

```
#Fills record 28 for PO4 with the help of coeifficients we found above
algae[28, "PO4"] <- 42.897 + 1.293 * algae[28, "oPO4"]
```

The best would be to create a function that would return the value of PO4 given the value of oPO4, and then apply this function to all unknown values:

```
#Loading the data in R
data(algae)

#Deleting the records with more than 0.2 NA values
algae <- algae[-manyNAs(algae), ]

#Creates a function which outputs
fillPO4 <- function(oP) {
 if (is.na(oP))
 return(NA)
 else return(42.897 + 1.293 * oP)
 }

#Filling the NA values as per the function
algae[is.na(algae$PO4), "PO4"] <- sapply(algae[is.na(algae$PO4), "oPO4"],
fillPO4)

#Produces a histogram of mxPH as per season
histogram(~mxPH | season, data = algae)
```

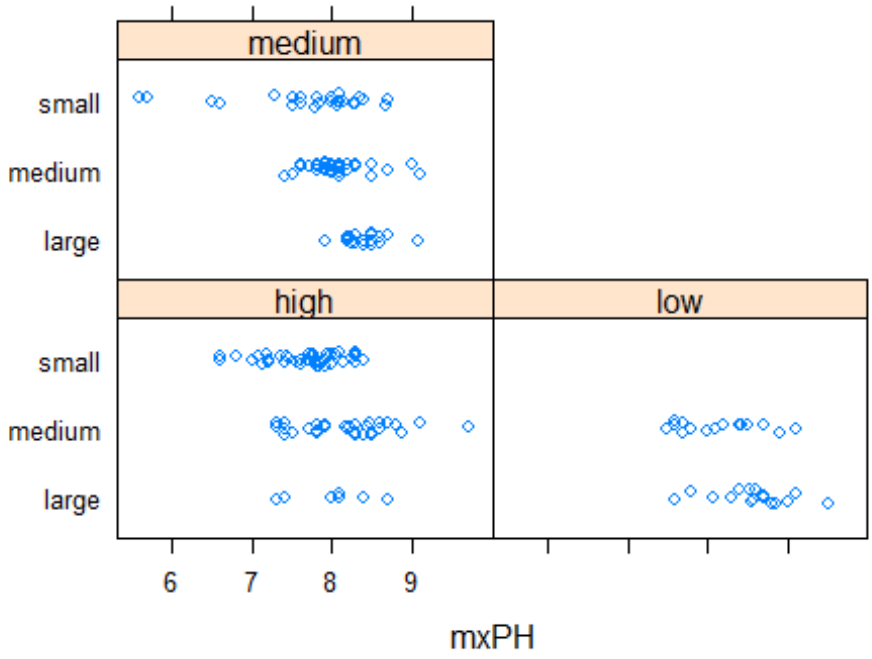



```
#Factoring the season variable
algae$season <- factor(algae$season, levels = c("spring", "summer", "autumn",
"winter"))
```

By looking at the above histogram we getr to know thatr season does not affect mxPh value

Now we plot histogram as per size of the river

```
#Plots a histogram
#histogram(~mxPH | size * speed, data = algae)

#Shows a strip plot
stripplot(size ~ mxPH | speed, data = algae, jitter = T)
```



Filling in the Unknown Values by Exploring Similarities between Cases

```
#Reading the data
data(algae)

#Filtering rows with most NA values
algae <- algae[-manyNAs(algae), ]

#using the knnImputation to fill NA values
algae <- knnImputation(algae, k = 10)

# the strategy of using the median values for filling in the unknowns
algae <- knnImputation(algae, k = 10, meth = "median")

## Warning in knnImputation(algae, k = 10, meth = "median"): No case has
## missing values. Stopping as there is nothing to do.

#Prints the summary of the dataset
summary(algae)

##     season       size       speed         mxPH            mnO2
##  autumn:40   large :44   high  :84   Min.   :5.600   Min.   : 1.500
##  spring:53   medium:84   low   :33   1st Qu.:7.705   1st Qu.: 7.825
##  summer:44   small :70   medium:81   Median :8.060   Median : 9.800
##  winter:61                           Mean   :8.019   Mean   : 9.132
##                                      3rd Qu.:8.400   3rd Qu.:10.800
##                                      Max.   :9.700   Max.   :13.400
```

```
##        Cl                NO3               NH4                 oPO4
##  Min.   :  0.222   Min.   : 0.050   Min.   :    5.00   Min.   :  1.00
##  1st Qu.: 10.514   1st Qu.: 1.296   1st Qu.:   38.33   1st Qu.: 15.70
##  Median : 32.178   Median : 2.675   Median :  103.17   Median : 40.15
##  Mean   : 42.594   Mean   : 3.282   Mean   :  501.30   Mean   : 73.59
##  3rd Qu.: 57.750   3rd Qu.: 4.446   3rd Qu.:  226.95   3rd Qu.: 99.33
##  Max.   :391.500   Max.   :45.650   Max.   :24064.00   Max.   :564.60
##       PO4              Chla              a1                a2
##  Min.   :  1.00   Min.   :  0.20   Min.   : 0.000   Min.   : 0.000
##  1st Qu.: 41.38   1st Qu.:  2.00   1st Qu.: 1.525   1st Qu.: 0.000
##  Median :103.29   Median :  5.20   Median : 6.950   Median : 3.000
##  Mean   :137.89   Mean   : 13.47   Mean   :16.996   Mean   : 7.471
##  3rd Qu.:213.75   3rd Qu.: 17.20   3rd Qu.:24.800   3rd Qu.:11.275
##  Max.   :771.60   Max.   :110.46   Max.   :89.800   Max.   :72.600
##        a3               a4               a5                a6
##  Min.   : 0.000   Min.   : 0.000   Min.   : 0.000   Min.   : 0.000
##  1st Qu.: 0.000   1st Qu.: 0.000   1st Qu.: 0.000   1st Qu.: 0.000
##  Median : 1.550   Median : 0.000   Median : 2.000   Median : 0.000
##  Mean   : 4.334   Mean   : 1.997   Mean   : 5.116   Mean   : 6.005
##  3rd Qu.: 4.975   3rd Qu.: 2.400   3rd Qu.: 7.500   3rd Qu.: 6.975
##  Max.   :42.800   Max.   :44.600   Max.   :44.400   Max.   :77.600
##        a7
##  Min.   : 0.000
##  1st Qu.: 0.000
##  Median : 1.000
##  Mean   : 2.487
##  3rd Qu.: 2.400
##  Max.   :31.600
```

By looking at the Summary we know that knnImputation has filled the NA values.

Obtaining Prediction Models

```r
#Loading the data
data(algae)

#Filtering the values with more than 20% NA values
algae <- algae[-manyNAs(algae), ]

#Applying KnnImputation to interpret NA values with k=10
clean.algae <- knnImputation(algae, k = 10)

#Applies a linear model with 12 predictors to predict a1
lm.a1 <- lm(a1 ~ ., data = clean.algae[, 1:12])
```

Now let us print the summary of the model

```r
#Prints the summary of the model
summary(lm.a1)
```

```
##
## Call:
## lm(formula = a1 ~ ., data = clean.algae[, 1:12])
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -37.679 -11.893  -2.567   7.410  62.190
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  42.942055  24.010879   1.788  0.07537 .
## seasonspring  3.726978   4.137741   0.901  0.36892
## seasonsummer  0.747597   4.020711   0.186  0.85270
## seasonwinter  3.692955   3.865391   0.955  0.34065
## sizemedium    3.263728   3.802051   0.858  0.39179
## sizesmall     9.682140   4.179971   2.316  0.02166 *
## speedlow      3.922084   4.706315   0.833  0.40573
## speedmedium   0.246764   3.241874   0.076  0.93941
## mxPH         -3.589118   2.703528  -1.328  0.18598
## mnO2          1.052636   0.705018   1.493  0.13715
## Cl           -0.040172   0.033661  -1.193  0.23426
## NO3          -1.511235   0.551339  -2.741  0.00674 **
## NH4           0.001634   0.001003   1.628  0.10516
## oPO4         -0.005435   0.039884  -0.136  0.89177
## PO4          -0.052241   0.030755  -1.699  0.09109 .
## Chla         -0.088022   0.079998  -1.100  0.27265
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 17.65 on 182 degrees of freedom
## Multiple R-squared:  0.3731, Adjusted R-squared:  0.3215
## F-statistic: 7.223 on 15 and 182 DF,  p-value: 2.444e-12
```

```
#Prints all the relevant summary of the model
anova(lm.a1)
```

```
## Analysis of Variance Table
##
## Response: a1
##            Df Sum Sq Mean Sq F value    Pr(>F)
## season      3     85    28.2  0.0905 0.9651944
## size        2  11401  5700.7 18.3088  5.69e-08 ***
## speed       2   3934  1967.2  6.3179 0.0022244 **
## mxPH        1   1329  1328.8  4.2677 0.0402613 *
## mnO2        1   2287  2286.8  7.3444 0.0073705 **
## Cl          1   4304  4304.3 13.8239 0.0002671 ***
## NO3         1   3418  3418.5 10.9789 0.0011118 **
## NH4         1    404   403.6  1.2963 0.2563847
## oPO4        1   4788  4788.0 15.3774 0.0001246 ***
## PO4         1   1406  1405.6  4.5142 0.0349635 *
```

```
## Chla         1    377   377.0  1.2107 0.2726544
## Residuals 182  56668   311.4
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

These results indicate that the variable season is the variable that least contributes to the reduction of the fitting error of the model.

Let us remove season factor from the model

```
#Updates the model lm.al and removes the season parameter
lm2.a1 <- update(lm.a1, . ~ . - season)

#Prints the summary of the model
summary(lm2.a1)

##
## Call:
## lm(formula = a1 ~ size + speed + mxPH + mnO2 + Cl + NO3 + NH4 +
##     oPO4 + PO4 + Chla, data = clean.algae[, 1:12])
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -36.460 -11.953  -3.044   7.444  63.730
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) 44.9532874 23.2378377   1.934  0.05458 .
## sizemedium   3.3092102  3.7825221   0.875  0.38278
## sizesmall   10.2730961  4.1223163   2.492  0.01358 *
## speedlow     3.0546270  4.6108069   0.662  0.50848
## speedmedium -0.2976867  3.1818585  -0.094  0.92556
## mxPH        -3.2684281  2.6576592  -1.230  0.22033
## mnO2         0.8011759  0.6589644   1.216  0.22561
## Cl          -0.0381881  0.0333791  -1.144  0.25407
## NO3         -1.5334300  0.5476550  -2.800  0.00565 **
## NH4          0.0015777  0.0009951   1.586  0.11456
## oPO4        -0.0062392  0.0395086  -0.158  0.87469
## PO4         -0.0509543  0.0305189  -1.670  0.09669 .
## Chla        -0.0841371  0.0794459  -1.059  0.29096
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 17.57 on 185 degrees of freedom
## Multiple R-squared:  0.3682, Adjusted R-squared:  0.3272
## F-statistic: 8.984 on 12 and 185 DF,  p-value: 1.762e-13
```

The fit has improved a bit (32.8%) but it is still not too impressive.

We can carry out a more formal comparison between the two models by using again the anova() function, but this time with both models as arguments:

```
#Prints the comparision of both the models
anova(lm.a1,lm2.a1)

## Analysis of Variance Table
##
## Model 1: a1 ~ season + size + speed + mxPH + mnO2 + Cl + NO3 + NH4 + oPO4
+
##     PO4 + Chla
## Model 2: a1 ~ size + speed + mxPH + mnO2 + Cl + NO3 + NH4 + oPO4 + PO4 +
##     Chla
##   Res.Df   RSS Df Sum of Sq      F Pr(>F)
## 1    182 56668
## 2    185 57116 -3   -447.62 0.4792 0.6971
```

In this case, although the sum of the squared errors has decreased (448), the comparison shows that the di???erences are not significant (a value of 0.6971 tells us that with only around 30% confidence we can say they are di???erent). Still, we should recall that this new model is simpler. In order to check if we can remove more coefficients, we would again use the anova() function, applied to the lm2.a1 model. This process would continue until we have no candidate coecients for removal. However, to simplify our backward elimination process,

Below we create a linear model that results from applying the backward elimination method to the initial model we have obtained (lm.a1)

```
#Eleminates each variable and performs linear regression
final.lm <- step(lm.a1)

## Start:  AIC=1152.03
## a1 ~ season + size + speed + mxPH + mnO2 + Cl + NO3 + NH4 + oPO4 +
##     PO4 + Chla
##
##          Df Sum of Sq   RSS    AIC
## - season  3    447.62 57116 1147.6
## - speed   2    269.60 56938 1149.0
## - oPO4    1      5.78 56674 1150.0
## - Chla    1    376.96 57045 1151.3
## - Cl      1    443.46 57112 1151.6
## - mxPH    1    548.76 57217 1151.9
## <none>                56668 1152.0
## - mnO2    1    694.11 57363 1152.4
## - NH4     1    825.67 57494 1152.9
## - PO4     1    898.42 57567 1153.1
## - size    2   1857.16 58526 1154.4
## - NO3     1   2339.36 59008 1158.0
##
## Step:  AIC=1147.59
## a1 ~ size + speed + mxPH + mnO2 + Cl + NO3 + NH4 + oPO4 + PO4 +
##     Chla
##
```

```
##         Df Sum of Sq   RSS    AIC
## - speed  2    210.64 57327 1144.3
## - oPO4   1      7.70 57124 1145.6
## - Chla   1    346.27 57462 1146.8
## - Cl     1    404.10 57520 1147.0
## - mnO2   1    456.37 57572 1147.2
## - mxPH   1    466.95 57583 1147.2
## <none>               57116 1147.6
## - NH4    1    776.11 57892 1148.3
## - PO4    1    860.62 57977 1148.5
## - size   2   2175.59 59292 1151.0
## - NO3    1   2420.47 59537 1153.8
##
## Step:  AIC=1144.31
## a1 ~ size + mxPH + mnO2 + Cl + NO3 + NH4 + oPO4 + PO4 + Chla
##
##        Df Sum of Sq   RSS    AIC
## - oPO4  1     16.29 57343 1142.4
## - Chla  1    223.29 57550 1143.1
## - mnO2  1    413.77 57740 1143.7
## - Cl    1    472.70 57799 1143.9
## - mxPH  1    483.56 57810 1144.0
## <none>              57327 1144.3
## - NH4   1    720.19 58047 1144.8
## - PO4   1    809.30 58136 1145.1
## - size  2   2060.95 59388 1147.3
## - NO3   1   2379.75 59706 1150.4
##
## Step:  AIC=1142.37
## a1 ~ size + mxPH + mnO2 + Cl + NO3 + NH4 + PO4 + Chla
##
##        Df Sum of Sq   RSS    AIC
## - Chla  1     207.7 57551 1141.1
## - mnO2  1     402.6 57746 1141.8
## - Cl    1     470.7 57814 1142.0
## - mxPH  1     519.7 57863 1142.2
## <none>              57343 1142.4
## - NH4   1     704.4 58047 1142.8
## - size  2    2050.3 59393 1145.3
## - NO3   1    2370.4 59713 1148.4
## - PO4   1    5818.4 63161 1159.5
##
## Step:  AIC=1141.09
## a1 ~ size + mxPH + mnO2 + Cl + NO3 + NH4 + PO4
##
##        Df Sum of Sq   RSS    AIC
## - mnO2  1     435.3 57986 1140.6
## - Cl    1     438.1 57989 1140.6
## <none>              57551 1141.1
## - NH4   1     746.9 58298 1141.6
```

```
## - mxPH  1     833.1 58384 1141.9
## - size  2    2217.5 59768 1144.6
## - NO3   1    2667.1 60218 1148.1
## - PO4   1    6309.7 63860 1159.7
##
## Step:  AIC=1140.58
## a1 ~ size + mxPH + Cl + NO3 + NH4 + PO4
##
##        Df Sum of Sq   RSS    AIC
## - NH4   1     531.0 58517 1140.4
## - Cl    1     584.9 58571 1140.6
## <none>              57986 1140.6
## - mxPH  1     819.1 58805 1141.4
## - size  2    2478.2 60464 1144.9
## - NO3   1    2251.4 60237 1146.1
## - PO4   1    9097.9 67084 1167.4
##
## Step:  AIC=1140.38
## a1 ~ size + mxPH + Cl + NO3 + PO4
##
##        Df Sum of Sq   RSS    AIC
## <none>              58517 1140.4
## - mxPH  1     784.1 59301 1141.0
## - Cl    1     835.6 59353 1141.2
## - NO3   1    1987.9 60505 1145.0
## - size  2    2664.3 61181 1145.2
## - PO4   1    8575.8 67093 1165.5
```

```
#Prints the summary of the model
summary(final.lm)
```

```
##
## Call:
## lm(formula = a1 ~ size + mxPH + Cl + NO3 + PO4, data = clean.algae[,
##     1:12])
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -28.874 -12.732  -3.741   8.424  62.926
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 57.28555   20.96132   2.733  0.00687 **
## sizemedium   2.80050    3.40190   0.823  0.41141
## sizesmall   10.40636    3.82243   2.722  0.00708 **
## mxPH        -3.97076    2.48204  -1.600  0.11130
## Cl          -0.05227    0.03165  -1.651  0.10028
## NO3         -0.89529    0.35148  -2.547  0.01165 *
## PO4         -0.05911    0.01117  -5.291 3.32e-07 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 17.5 on 191 degrees of freedom
## Multiple R-squared:  0.3527, Adjusted R-squared:  0.3324
## F-statistic: 17.35 on 6 and 191 DF,  p-value: 5.554e-16
```

By looking at the R swuared value we may conclude that the model which is refined is also able to explain just 35% of the data.

Regression Trees

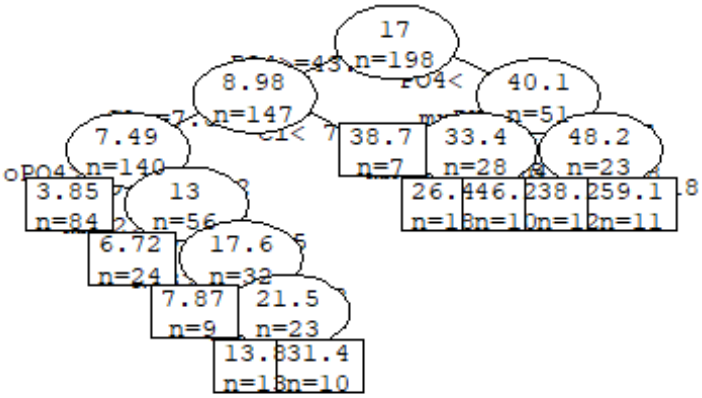
As these models handle datasetswith missing values, we only need to remove samples 62 and 199 for the reasons mentioned before.

```
#Loading the required library
library(rpart)

##
## Attaching package: 'rpart'

## The following object is masked from 'package:survival':
##
##     solder

#Loading the data
data(algae)

#Filtering the rows with more than 20% NA values
algae <- algae[-manyNAs(algae), ]

#Obtains the regression tree
rt.a1 <- rpart(a1 ~ ., data = algae[, 1:12])


#Prints the content of the variable
rt.a1

## n= 198
##
## node), split, n, deviance, yval
##       * denotes terminal node
##
##  1) root 198 90401.290 16.996460
##    2) PO4>=43.818 147 31279.120  8.979592
##      4) Cl>=7.8065 140 21622.830  7.492857
##        8) oPO4>=51.118 84  3441.149  3.846429 *
##        9) oPO4< 51.118 56 15389.430 12.962500
##         18) mnO2>=10.05 24  1248.673  6.716667 *
##         19) mnO2< 10.05 32 12502.320 17.646870
##           38) NO3>=3.1875 9   257.080  7.866667 *
##           39) NO3< 3.1875 23 11047.500 21.473910
```

```
##             78) mnO2< 8 13  2919.549 13.807690 *
##             79) mnO2>=8 10  6370.704 31.440000 *
##      5) Cl< 7.8065 7  3157.769 38.714290 *
##    3) PO4< 43.818 51 22442.760 40.103920
##      6) mxPH< 7.87 28 11452.770 33.450000
##       12) mxPH>=7.045 18  5146.169 26.394440 *
##       13) mxPH< 7.045 10  3797.645 46.150000 *
##      7) mxPH>=7.87 23  8241.110 48.204350
##       14) PO4>=15.177 12  3047.517 38.183330 *
##       15) PO4< 15.177 11  2673.945 59.136360 *
```

Let us plot the graphical representation of the tree

```
#Prints the graphical representation of the regression decision trees
prettyTree(rt.a1)
```



Given a tree obtained with the rpart() function, R can produce a set of sub-trees of this tree and estimate their predictive performance. This information can be obtained using the function printcp():

```
#Prints the c
printcp(rt.a1)

##
## Regression tree:
## rpart(formula = a1 ~ ., data = algae[, 1:12])
##
```

```
## Variables actually used in tree construction:
## [1] Cl   mnO2 mxPH NO3  oPO4 PO4
##
## Root node error: 90401/198 = 456.57
##
## n= 198
##
##         CP nsplit rel error  xerror    xstd
## 1 0.405740      0   1.00000 1.01613 0.13119
## 2 0.071885      1   0.59426 0.70248 0.11833
## 3 0.030887      2   0.52237 0.75736 0.12314
## 4 0.030408      3   0.49149 0.73712 0.12654
## 5 0.027872      4   0.46108 0.71791 0.12600
## 6 0.027754      5   0.43321 0.72559 0.12679
## 7 0.018124      6   0.40545 0.72317 0.12198
## 8 0.016344      7   0.38733 0.71537 0.11831
## 9 0.010000      9   0.35464 0.71277 0.11898
```

```
#Prunes the tree with cp=0.08
rt2.a1 <- prune(rt.a1, cp = 0.08)

#Prints the pruned tree
rt2.a1
```

```
## n= 198
##
## node), split, n, deviance, yval
##       * denotes terminal node
##
## 1) root 198 90401.29 16.996460
##   2) PO4>=43.818 147 31279.12  8.979592 *
##   3) PO4< 43.818 51 22442.76 40.103920 *
```

```
#Automated pruning of the tree
(rt.a1 <- rpartXse(a1 ~ ., data = algae[, 1:12]))
```

```
## n= 198
##
## node), split, n, deviance, yval
##       * denotes terminal node
##
## 1) root 198 90401.290 16.996460
##   2) PO4>=43.818 147 31279.120  8.979592
##     4) Cl>=7.1665 142 21763.160  7.530282 *
##     5) Cl< 7.1665 5   746.792 50.140000 *
##   3) PO4< 43.818 51 22442.760 40.103920 *
```

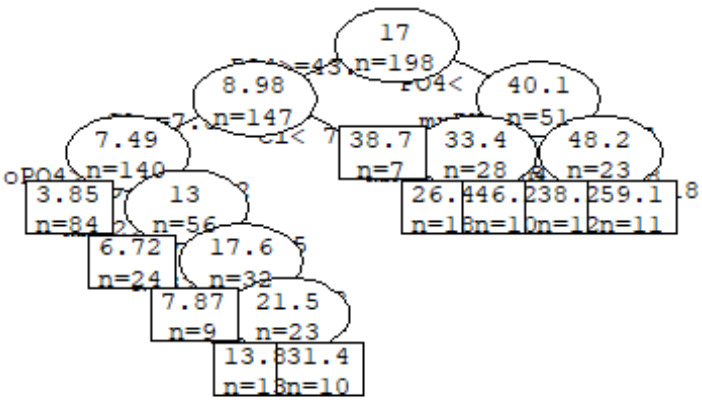
Method 2 of pruning the tree using snpr.part

```
#Creates the tree with all the parameters
first.tree <- rpart(a1 ~ ., data = algae[, 1:12])
```

```
#Prunes the tree at nodes 4 and 7
snip.rpart(first.tree, c(4, 7))
```

```
## n= 198
##
## node), split, n, deviance, yval
##       * denotes terminal node
##
##  1) root 198 90401.290 16.996460
##    2) PO4>=43.818 147 31279.120  8.979592
##      4) Cl>=7.8065 140 21622.830  7.492857 *
##      5) Cl< 7.8065 7  3157.769 38.714290 *
##    3) PO4< 43.818 51 22442.760 40.103920
##      6) mxPH< 7.87 28 11452.770 33.450000
##       12) mxPH>=7.045 18  5146.169 26.394440 *
##       13) mxPH< 7.045 10  3797.645 46.150000 *
##      7) mxPH>=7.87 23  8241.110 48.204350 *
```

```
#Prints the graphical representation of the tree
prettyTree(first.tree)
```




```
#snip.rpart(first.tree)
```

Model Evaluation and Selection

```
#Predicts a1 based on the best linear model
lm.predictions.a1 <- predict(final.lm, clean.algae)
```

```
#Predicts a1 based on regression tree
rt.predictions.a1 <- predict(rt.a1, algae)
```

Now we calculate error based on the actual values

```
#Calculates mean error and stores it in the variable
(mae.a1.lm <- mean(abs(lm.predictions.a1 - algae[, "a1"])))

## [1] 13.10681

#Calculates mean error and stores
(mae.a1.rt <- mean(abs(rt.predictions.a1 - algae[, "a1"])))

## [1] 10.36242
```

Let us calculate mean squared error and print it

```
#Calculates mean squared error and stores it in the variable
 (mse.a1.lm <- mean((lm.predictions.a1 - algae[, "a1"])^2))

## [1] 295.5407

#Calculates mean squared error and stores it in the variable
(mse.a1.rt <- mean((rt.predictions.a1 - algae[, "a1"])^2))

## [1] 227.0339
```

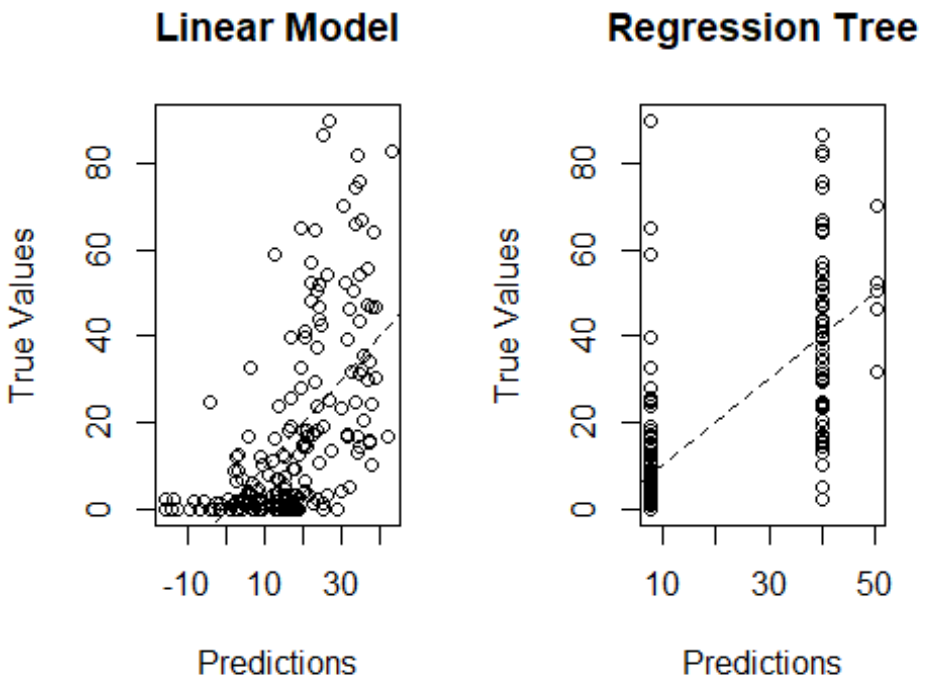
Let us normalize the error parameter

```
#Calculating and printing the normalized error for linear model
(nmse.a1.lm <- mean((lm.predictions.a1-
algae[,'a1'])^2)/mean((mean(algae[,'a1'])-algae[,'a1'])^2))

## [1] 0.6473034

#Calculating the normalized mean error for regression trees model
 (nmse.a1.rt <- mean((rt.predictions.a1-
algae[,'a1'])^2)/mean((mean(algae[,'a1'])-algae[,'a1'])^2))

## [1] 0.4972574
```

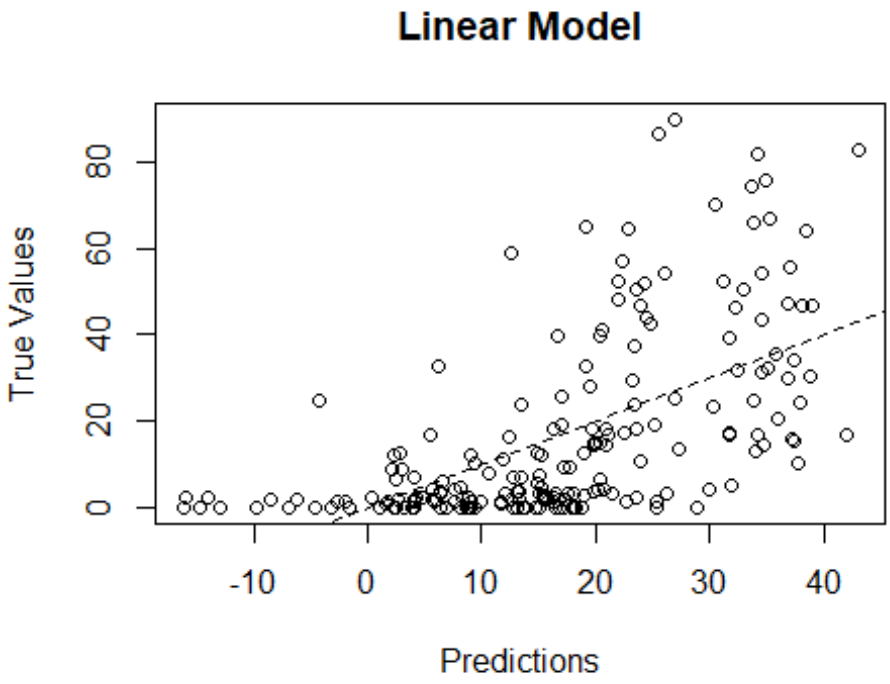
Calculating all the error parameters

```
#Calculating the error parameters
regr.eval(algae[, "a1"], rt.predictions.a1, train.y = algae[, "a1"])

##         mae         mse        rmse        mape        nmse        nmae
##  10.3624227 227.0338940  15.0676439         Inf   0.4972574   0.6202654

#Setting the visualization parameter
old.par <- par(mfrow = c(1, 2))

#Prints the true values against predicted values for linear model
```

```r
plot(lm.predictions.a1,algae[,'a1'],main="Linear
Model",xlab="Predictions",ylab="True Values")

#Plots a straight line
abline(0, 1, lty = 2)

#Plots the true values against the predicted values for Regression Trees
plot(rt.predictions.a1, algae[, "a1"], main = "Regression Tree",xlab =
"Predictions", ylab = "True Values")

#Drawing a straight line through the origin
abline(0, 1, lty = 2)
```




```r
#Setting the visualization parameter
par(old.par)

#Plots the error graph
plot(lm.predictions.a1,algae[,'a1'],main="Linear Model",
xlab="Predictions",ylab="True Values")

#Draws a line passing through origin
abline(0,1,lty=2)

#Prints the values of the rows as per the clicks
algae[identify(lm.predictions.a1,algae[,'a1']),]
```




```
##  [1] season size   speed  mxPH   mnO2   Cl     NO3    NH4    oPO4   PO4
## [11] Chla   a1     a2     a3     a4     a5     a6     a7
## <0 rows> (or 0-length row.names)
```

Improving the performance of the linear model as the min value for the frequency of the algae type 1 could be 0

```
#Making the predicted values of a1 less than 1 =0
sensible.lm.predictions.a1 <- ifelse(lm.predictions.a1 < 0, 0,
lm.predictions.a1)

#Calculating the error matrix of first prediction and a1
regr.eval(algae[, "a1"], lm.predictions.a1, stats = c("mae","mse"))

##       mae       mse
##  13.10681 295.54069

#Calculating the error values of the improved values of a1 with actual values
regr.eval(algae[, "a1"], sensible.lm.predictions.a1, stats = c("mae", "mse"))

##       mae       mse
##  12.48276 286.28541
```

Performing cross validation

```
#Creating the function for cross validation for regression tree
cv.rpart <- function(form,train,test,...) {
m <- rpartXse(form,train,...)
```

```
p <- predict(m,test)
mse <- mean((p-resp(form,test))^2)
c(nmse=mse/mean((mean(resp(form,train))-resp(form,test))^2))
}

#Creating the cross validation function for linear model
cv.lm <- function(form,train,test,...) {
m <- lm(form,train,...)
p <- predict(m,test)
p <- ifelse(p < 0,0,p)
mse <- mean((p-resp(form,test))^2)
c(nmse=mse/mean((mean(resp(form,train))-resp(form,test))^2))
}
```

Performing experimentalComparision function using the above functions

```
#Performs 3 times 10-fold crossvalidation
res <- experimentalComparison(
 c(dataset(a1 ~ .,clean.algae[,1:12],'a1')),
 c(variants('cv.lm'),variants('cv.rpart',se=c(0,0.5,1))),
 cvSettings(3,10,1234))

##
##
## #####  CROSS VALIDATION  EXPERIMENTAL COMPARISON #####
##
## ** DATASET :: a1
##
## ++ LEARNER :: cv.lm  variant ->  cv.lm.v1
##
##  3 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v1
##
##  3 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
```

```
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v2
##
##  3 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v3
##
##  3 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
```

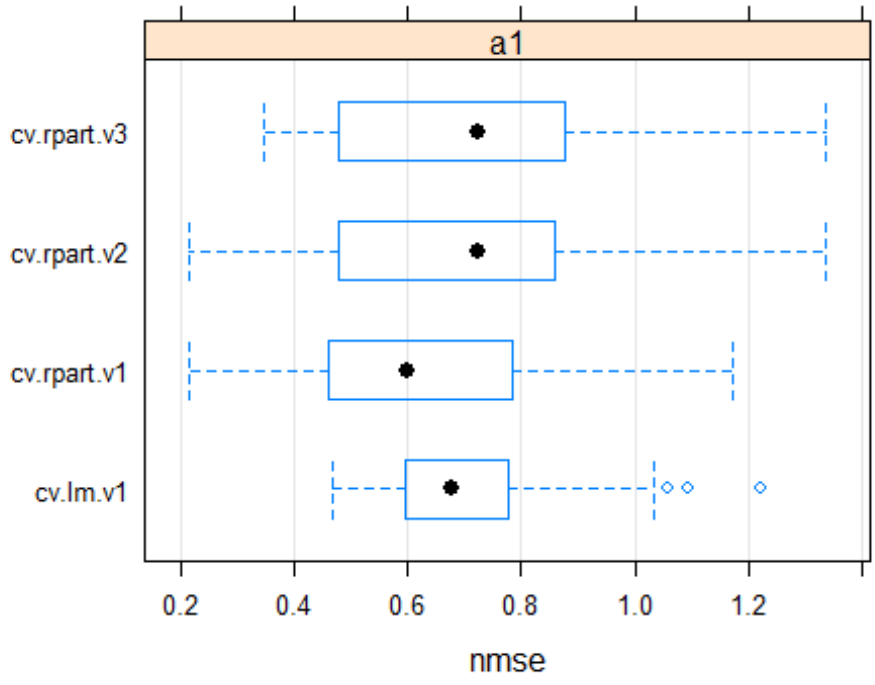
```
#Printing the summary of res
summary(res)
```

```
##
## == Summary of a  Cross Validation  Experiment ==
##
##  3 x 10 - Fold Cross Validation run with seed =  1234
##
## * Data sets ::  a1
## * Learners  ::  cv.lm.v1, cv.rpart.v1, cv.rpart.v2, cv.rpart.v3
##
## * Summary of Experiment Results:
##
##
## -> Datataset:  a1
##
##  *Learner: cv.lm.v1
##              nmse
## avg     0.7196105
## std     0.1833064
## min     0.4678248
## max     1.2218455
## invalid 0.0000000
##
##  *Learner: cv.rpart.v1
##              nmse
## avg     0.6440843
## std     0.2521952
## min     0.2146359
```

```
## max     1.1712674
## invalid 0.0000000
##
##  *Learner: cv.rpart.v2
##              nmse
## avg     0.6873747
## std     0.2669942
## min     0.2146359
## max     1.3356744
## invalid 0.0000000
##
##  *Learner: cv.rpart.v3
##              nmse
## avg     0.7167122
## std     0.2579089
## min     0.3476446
## max     1.3356744
## invalid 0.0000000
```

We can see that there isone nmse with least average error. Nowe we plot a visualization of the results

```
#Prints the visualization of cross validation
plot(res)
```




```
#Gets the parameter setting for rpart
getVariant("cv.rpart.v1", res)
```

```
##
## Learner::  "cv.rpart"
##
## Parameter values
##   se  =  0
```

Now we perform similar comparative experiment for all the seven prediction tasks we are facing at the same time

```
#Creates a function which gives specific dataset
DSs <- sapply(names(clean.algae)[12:18],
function(x,names.attrs) {
f <- as.formula(paste(x,"~ ."))
dataset(f,clean.algae[,c(names.attrs,x)],x)
},
names(clean.algae)[1:11])


#Performs 5 fold cross validation with 10 random values
res.all <- experimentalComparison(
 DSs,
 c(variants('cv.lm'),
 variants('cv.rpart',se=c(0,0.5,1))
 ),
 cvSettings(5,10,1234))

##
##
## #####  CROSS VALIDATION  EXPERIMENTAL COMPARISON #####
##
## ** DATASET :: a1
##
## ++ LEARNER :: cv.lm  variant ->  cv.lm.v1
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v1
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
```

```
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v2
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v3
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ** DATASET :: a2
##
## ++ LEARNER :: cv.lm  variant ->  cv.lm.v1
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
```

```
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v1
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v2
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v3
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
```

```
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ** DATASET :: a3
##
## ++ LEARNER :: cv.lm  variant ->  cv.lm.v1
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v1
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v2
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
```

```
##
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v3
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ** DATASET :: a4
##
## ++ LEARNER :: cv.lm  variant ->  cv.lm.v1
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v1
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v2
##
```

```
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v3
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ** DATASET :: a5
##
## ++ LEARNER :: cv.lm  variant ->  cv.lm.v1
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v1
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
```

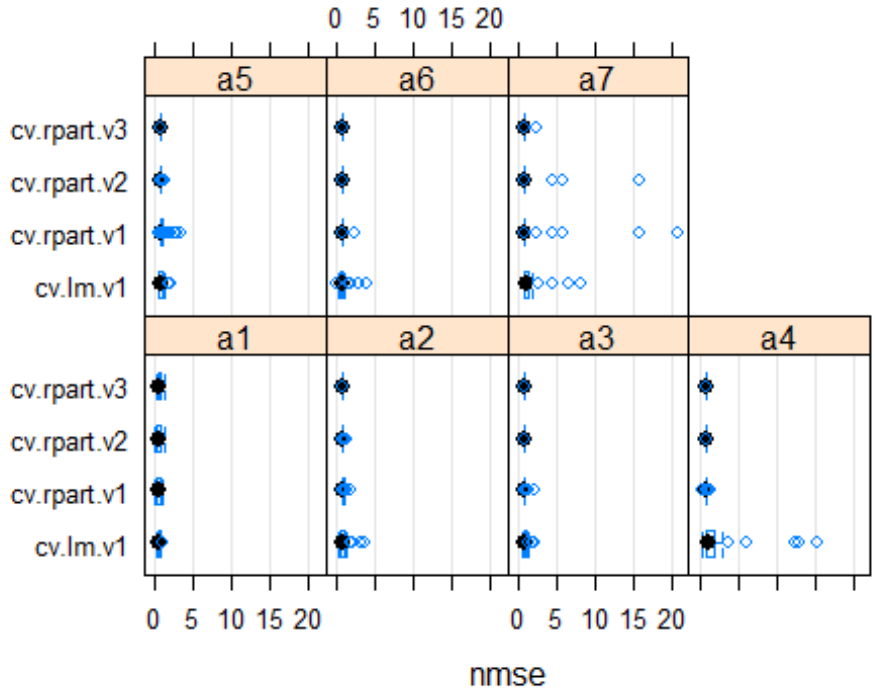
```
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v2
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v3
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ** DATASET :: a6
##
## ++ LEARNER :: cv.lm  variant ->  cv.lm.v1
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
```

```
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v1
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v2
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v3
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
```

```
##
##
## ** DATASET :: a7
##
## ++ LEARNER :: cv.lm  variant ->  cv.lm.v1
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v1
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v2
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v3
```

```
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
```

```
#Gives a boxplot for all the algae
plot(res.all)
```




```
#Prints the best score for all the algae
bestScores(res.all)
```

```
## $a1
##           system   score
## nmse cv.rpart.v1 0.64231
##
## $a2
##           system score
## nmse cv.rpart.v3     1
##
```

```
## $a3
##           system score
## nmse cv.rpart.v2     1
##
## $a4
##           system score
## nmse cv.rpart.v2     1
##
## $a5
##        system     score
## nmse cv.lm.v1 0.9316803
##
## $a6
##        system     score
## nmse cv.lm.v1 0.9359697
##
## $a7
##           system    score
## nmse cv.rpart.v3 1.029505
```

Performing Random Forest

```
#Installing the required packages
#install.packages("randomForest")

#Loading the required libraries
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

#Cross validation function
cv.rf <- function(form,train,test,...) {
m <- randomForest(form,train,...)
p <- predict(m,test)
mse <- mean((p-resp(form,test))^2)
c(nmse=mse/mean((mean(resp(form,train))-resp(form,test))^2))
}

#Performs 5 fold cross validation with 10 random values with
lm,rpart,randomforest
res.all <- experimentalComparison(
DSs,
```

```
c(variants('cv.lm'),
variants('cv.rpart',se=c(0,0.5,1)),
variants('cv.rf',ntree=c(200,500,700))
),
cvSettings(5,10,1234))

##
##
## #####  CROSS VALIDATION  EXPERIMENTAL COMPARISON #####
##
## ** DATASET :: a1
##
## ++ LEARNER :: cv.lm  variant ->  cv.lm.v1
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v1
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v2
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
```

```
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v3
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rf  variant ->  cv.rf.v1
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rf  variant ->  cv.rf.v2
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
```

```
##
##
## ++ LEARNER :: cv.rf  variant ->  cv.rf.v3
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ** DATASET :: a2
##
## ++ LEARNER :: cv.lm  variant ->  cv.lm.v1
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v1
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v2
```

```
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v3
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rf  variant ->  cv.rf.v1
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rf  variant ->  cv.rf.v2
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
```

```
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rf  variant ->  cv.rf.v3
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ** DATASET :: a3
##
## ++ LEARNER :: cv.lm  variant ->  cv.lm.v1
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v1
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
```

```
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v2
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v3
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rf  variant ->  cv.rf.v1
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
```

```
##
## ++ LEARNER :: cv.rf  variant ->  cv.rf.v2
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rf  variant ->  cv.rf.v3
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ** DATASET :: a4
##
## ++ LEARNER :: cv.lm  variant ->  cv.lm.v1
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v1
##
```

```
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v2
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v3
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rf  variant ->  cv.rf.v1
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
```

```
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rf  variant ->  cv.rf.v2
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rf  variant ->  cv.rf.v3
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ** DATASET :: a5
##
## ++ LEARNER :: cv.lm  variant ->  cv.lm.v1
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
```

```
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v1
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v2
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v3
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
```

```
## ++ LEARNER :: cv.rf  variant ->  cv.rf.v1
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rf  variant ->  cv.rf.v2
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rf  variant ->  cv.rf.v3
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ** DATASET :: a6
##
## ++ LEARNER :: cv.lm  variant ->  cv.lm.v1
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
```

```
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v1
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v2
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v3
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
```

```
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rf  variant ->  cv.rf.v1
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rf  variant ->  cv.rf.v2
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rf  variant ->  cv.rf.v3
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
```

```
##
##
## ** DATASET :: a7
##
## ++ LEARNER :: cv.lm  variant ->  cv.lm.v1
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v1
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v2
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rpart  variant ->  cv.rpart.v3
```

```
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rf  variant ->  cv.rf.v1
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rf  variant ->  cv.rf.v2
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
##
##
## ++ LEARNER :: cv.rf  variant ->  cv.rf.v3
##
##  5 x 10 - Fold Cross Validation run with seed =  1234
## Repetition  1
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  2
```

```
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  3
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  4
## Fold:  1  2  3  4  5  6  7  8  9  10
## Repetition  5
## Fold:  1  2  3  4  5  6  7  8  9  10
```

Finding the best scores

```
#Prints the best values that were found by the algorithms
bestScores(res.all)
```

```
## $a1
##        system     score
## nmse cv.rf.v3 0.5467636
##
## $a2
##        system     score
## nmse cv.rf.v3 0.7695782
##
## $a3
##           system score
## nmse cv.rpart.v2     1
##
## $a4
##        system     score
## nmse cv.rf.v1 0.9728596
##
## $a5
##        system     score
## nmse cv.rf.v2 0.7916332
##
## $a6
##        system    score
## nmse cv.rf.v2 0.911758
##
## $a7
##           system    score
## nmse cv.rpart.v3 1.029505
```

```
#compAnalysis(res.all,against='cv.rf.v3', datasets=c('a1','a2','a4','a6'))
```

Predictions of the 7 algae

```
#Storing the best model for algae prediction
bestModelsNames <- sapply(bestScores(res.all),function(x) x['nmse','system'])


#Creating the vectors
learners <- c(rf='randomForest',rpart='rpartXse')
```

```r
#Filtering the names of best Models
funcs <- learners[sapply(strsplit(bestModelsNames,'\\.'), function(x) x[2])]


#Assigned with para settings for each variants
parSetts <- lapply(bestModelsNames, function(x) getVariant(x,res.all)@pars)


#Creating the null list
bestModels <- list()


#Applying the models
for(a in 1:7) {
form <- as.formula(paste(names(clean.algae)[11+a],'~ .'))
bestModels[[a]] <- do.call(funcs[a],
c(list(form,clean.algae[,c(1:11,11+a)]),parSetts[[a]]))
}

#Cleaning the data and filling the NA values in the dataset
clean.test.algae <- knnImputation(test.algae, k = 10, distData = algae[,
1:11])

#Creating the predictor matrix
preds <- matrix(ncol=7,nrow=140)

#Applying the best model for the dataset
 for(i in 1:nrow(clean.test.algae))
 preds[i,] <- sapply(1:7,
 function(x)
 predict(bestModels[[x]],clean.test.algae[i,]))

#Calculating mean
avg.preds <- apply(algae[,12:18],2,mean)

#Calculating Normalized mean squared Error
apply( ((algae.sols-preds)^2), 2,mean) /apply(
(scale(algae.sols,avg.preds,F)^2),2,mean)
```

```
##        a1        a2        a3        a4        a5        a6        a7
## 0.4691854 0.8744285 1.0000000 0.7658754 0.7081748 0.8294282 1.0000000
```