Big Data Analytics

Session 11

Predicting Algae Bloom

Problem Description

- Harmful algae in rivers
  - A serious ecological problem
    - Strong impact on
      - river life forms and
      - water quality

- Objectives:
  - Monitor and perform an early forecast of algae blooms
    - to improve the quality of rivers
    - chemical monitoring is cheaper and easily automated than biological analysis (microscopic examination)
  - Provide a better understanding of the factors influencing the algae frequencies

Data Collection

- Several water samples were collected in different European rivers at different times during a period of approximately 1 year.
- For each water sample,
  - different chemical properties were measured, as well as
  - the frequency of several harmful algae
- Some related characteristics were stored
  - the season of the year
  - the river size
  - the river speed
- Data was collected in the context of the ERUDIT research Network
  - available in the UCI machine learning repository
  - http://archive.ics.uci.edu/ml/datasets/Coil+1999+Competition+Data

Data Description

- Two main datasets:
  - Training dataset
    - 200 observations
    - 11 predictors
      - Nominal (3): season, size, speed
      - Numerical (8): different chemical parameters measured in the water samples
        - » Maximum pH value, Minimum value of O_2 (Oxygen)
        - » Mean value of Cl, NO_3^-, NH_4^+, PO_4^{3-}, PO_4, chlorophyll
    - 7 responses
      - Seven frequency numbers of different harmful algae found in respective sample
  - Test dataset
    - 140 observations
    - 11 predictors
    - no responses

Goal: to predict the frequency of the seven algae for these 140 water samples

Load the Data into R

- Download the data (in .txt form) to your working directory (getwd()) from http://www.dcc.fc.up.pt/~ltorgo/DataMiningWithR/datasets2.html or https://archive.ics.uci.edu/ml/machine-learning-databases/coil-mld/coil.html
  - Analysis.txt: training data; Eval.txt: test data

http://www.dcc.fc.up.pt/~ltorgo/DataMiningWithR/DataSets/Analysis.txt

```
algae <- read.table('Analysis.txt', header=F, dec='.',
          col.names=c('season','size','speed','mxPH','mnO2','Cl','NO3','NH4','oPO4',
                      'PO4','Chla','a1','a2','a3','a4', 'a5','a6','a7'),
          na.strings=c('XXXXXXX'))
```

#header=F: indicates that the file to be read does not include a first line with variable names

#dec='.': the numbers use '.' to separate decimal places (e.g., 34.2)

#na.strings: unknown values are represented by XXXXXXX

or `library(DMwR)`

or `algae <- read.table('/Users/than/../Analysis.txt', header=F, dec='.', … )`

`/Users/than/../` is the directory where the training data is stored.

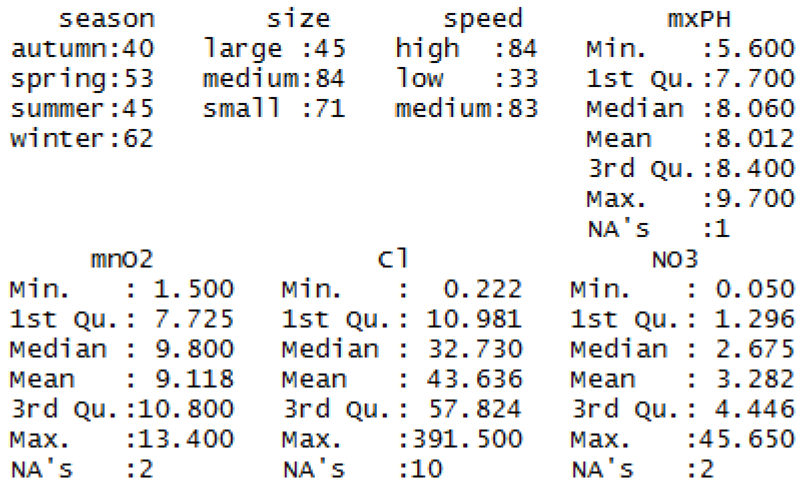
```
> head(algae)
  season size  speed  mxPH mnO2 Cl     NO3    NH4     oPO4    PO4     Chla    a1   a2   a3   a4   a5   a6   a7
1 winter small medium 8.00 9.8  60.800 6.238  578.000 105.000 170.000 50.0    0.0  0.0  0.0  0.0  34.2 8.3  0.0
2 spring small medium 8.35 8.0  57.750 1.288  370.000 428.750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

Descriptive Data Analysis

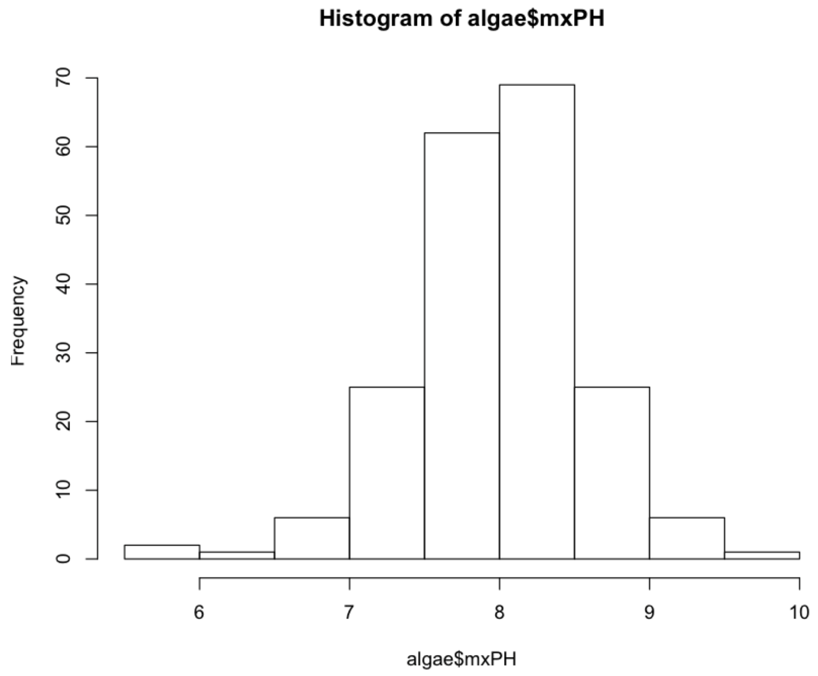
Data Visualisation and Summarisation

Data Visualisation and Summarisation

- Use `summary(algae)`
  - Notice the difference that nominal and numerical variables are presented
    - Nominal: frequency counts
    - Numerical: 5 number summary



```
   season        size          speed          mxPH      
 autumn:40   large :45    high  :84    Min.   :5.600  
 spring:53   medium:84    low   :33    1st Qu.:7.700  
 summer:45   small :71    medium:83    Median :8.060  
 winter:62                             Mean   :8.012  
                                       3rd Qu.:8.400  
                                       Max.   :9.700  
                                       NA's   :1      
     mnO2              Cl               NO3        
 Min.   : 1.500   Min.   :  0.222   Min.   : 0.050  
 1st Qu.: 7.725   1st Qu.: 10.981   1st Qu.: 1.296  
 Median : 9.800   Median : 32.730   Median : 2.675  
 Mean   : 9.118   Mean   : 43.636   Mean   : 3.282  
 3rd Qu.:10.800   3rd Qu.: 57.824   3rd Qu.: 4.446  
 Max.   :13.400   Max.   :391.500   Max.   :45.650  
 NA's   :2        NA's   :10        NA's   :2       
```



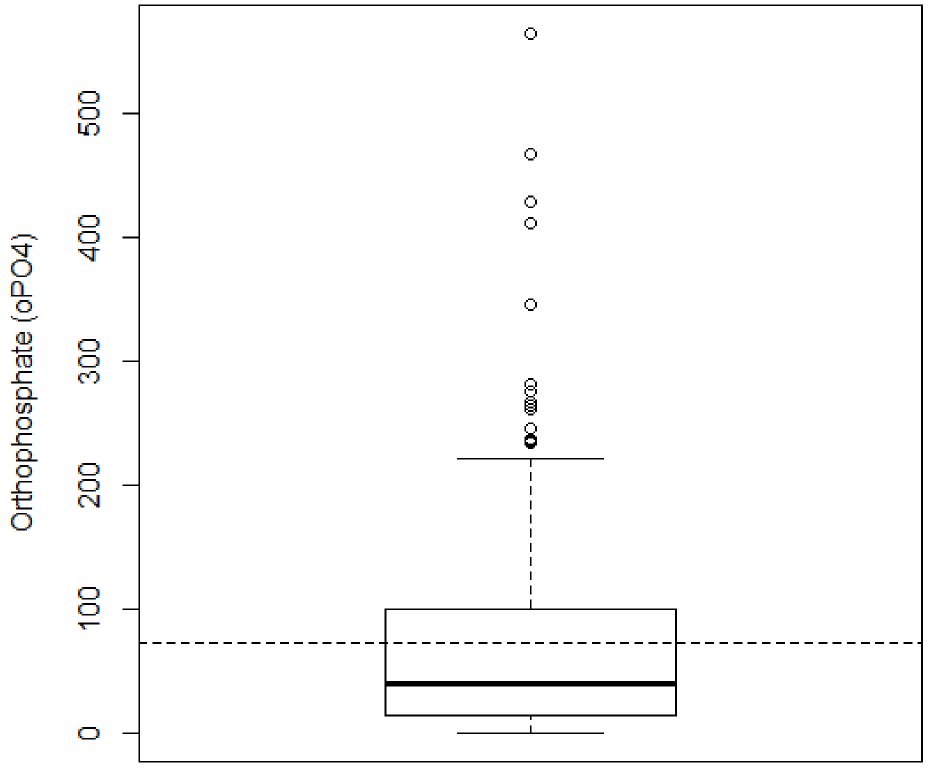
- Use graphs to check the shape of distribution

`> hist(algae$mxPH)` # the shape suggests that mxPH is nearly normal distributed

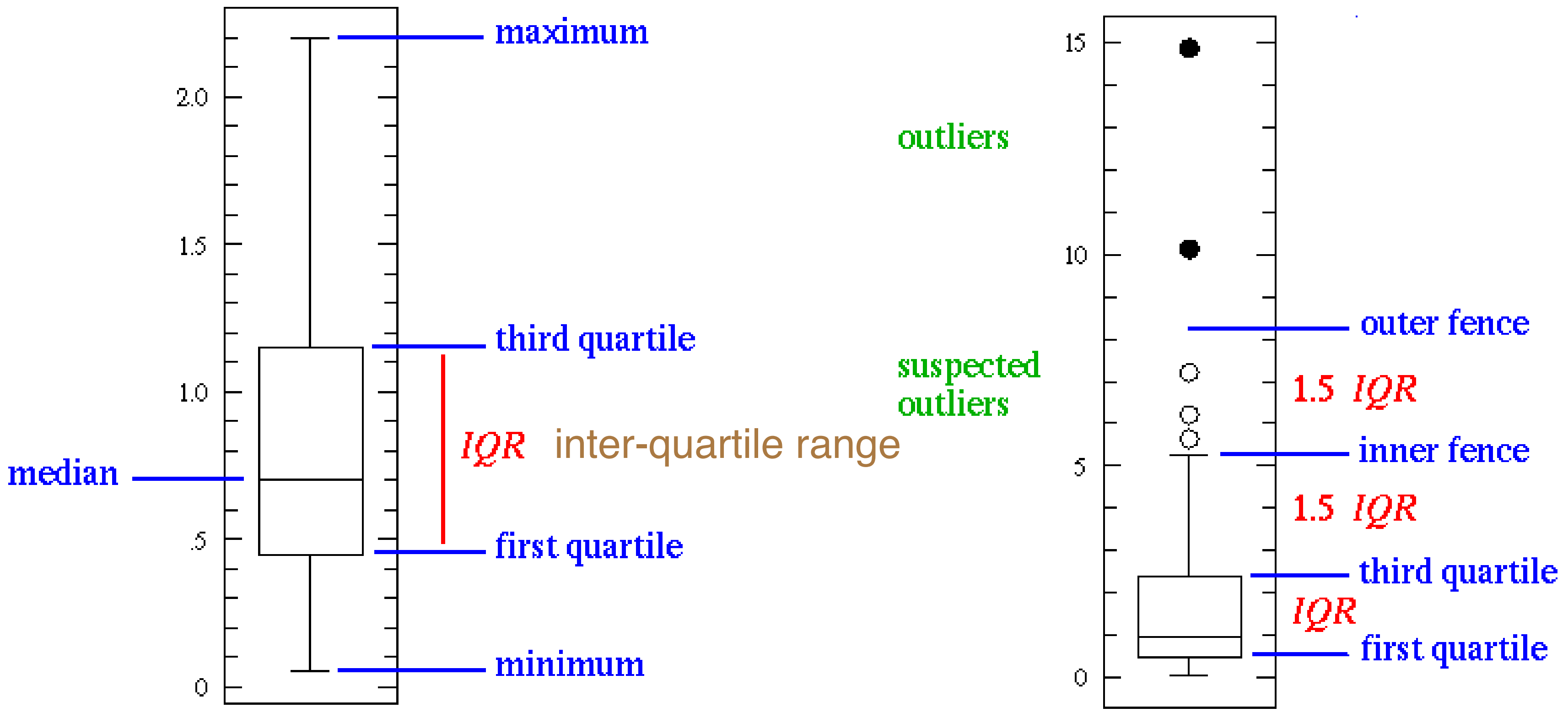
Data Visualisation and Summarisation

- Or boxplot

```
> boxplot(algae$oPO4, ylab='Orthophosphate (oPO4)')
> abline(h=mean(algae$oPO4,na.rm=T),lty=2)
```

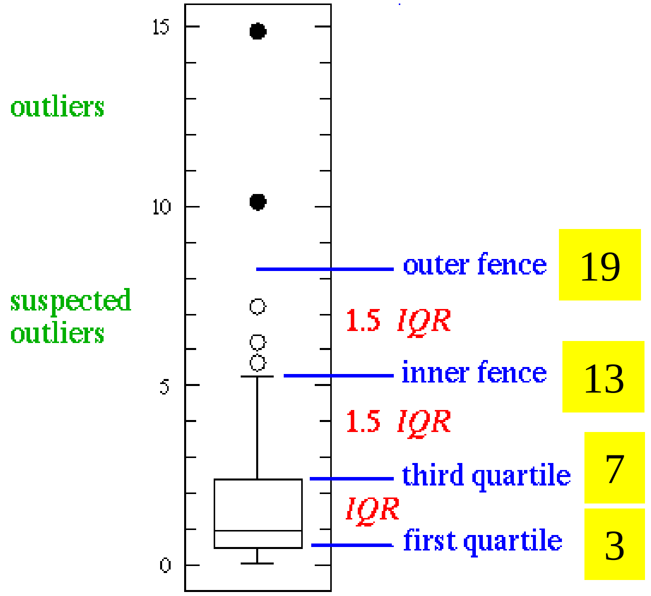


Boxplots and Outliers

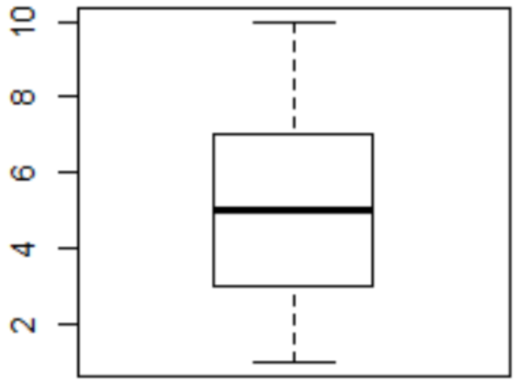


In R, all the suspected outliers and outliers are unfilled circles.

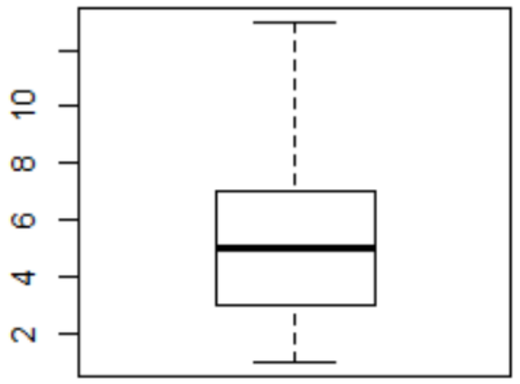
Boxplots and Outliers



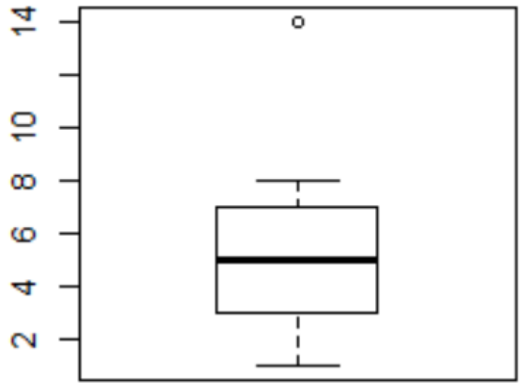
data is 1,2,3,4,5,6,7,8,x
Q1=3, Q2=5, Q3=7
IQR = 7-3 = 4
1.5*IQR = 6
3*IQR = 12
inner fence = Q3+1.5*IQR = 13
outer fence = Q3+3*IQR =19



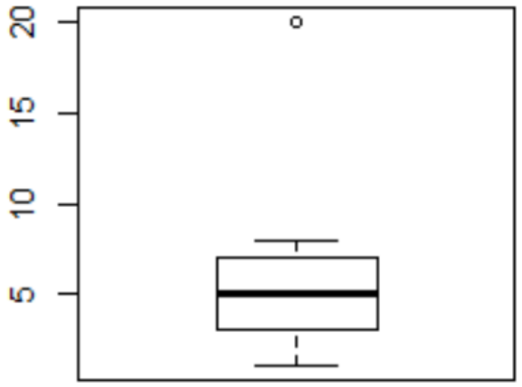
values1 = c(1:8,10)



values2 = c(1:8,13)



values3 = c(1:8,14)

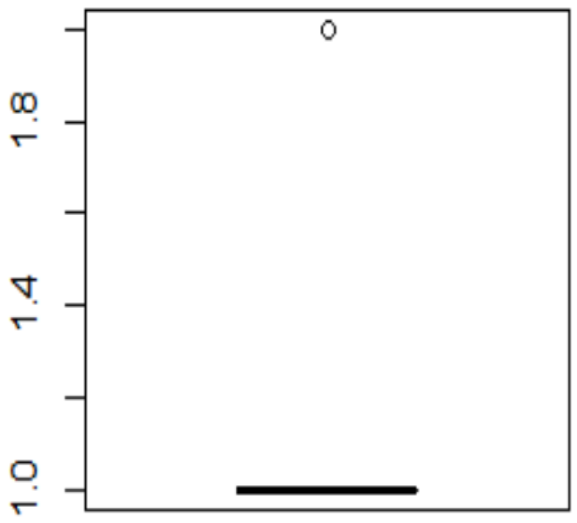


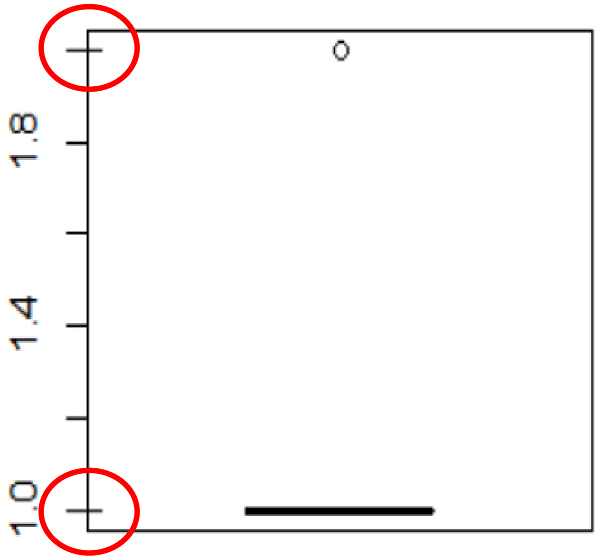
values4 = c(1:8,20)

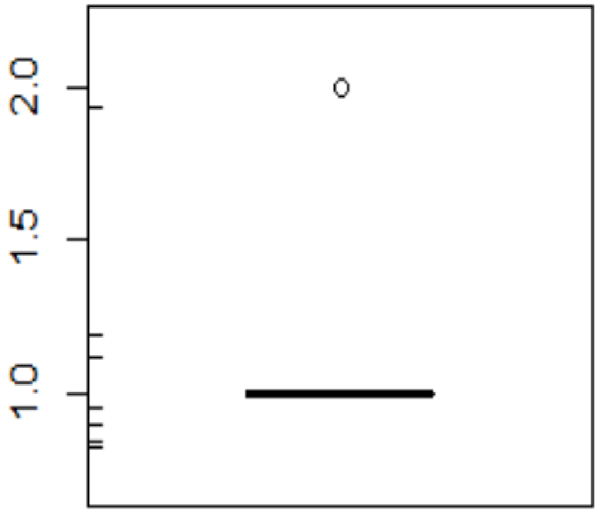
Outliers – A Remark

- Outliers are not necessarily "bad" data-points
- They may well be the most important, most information rich, part of the dataset
- Under no circumstances should they be automatically removed from the dataset
- Outliers may deserve special consideration
  - they may be the key to the phenomenon under study or the result of human blunders

rug and jitter







```
> values <- c(1,1,1,1,1,1,2)
> boxplot(values)
```

```
> rug(values, side=2)
```

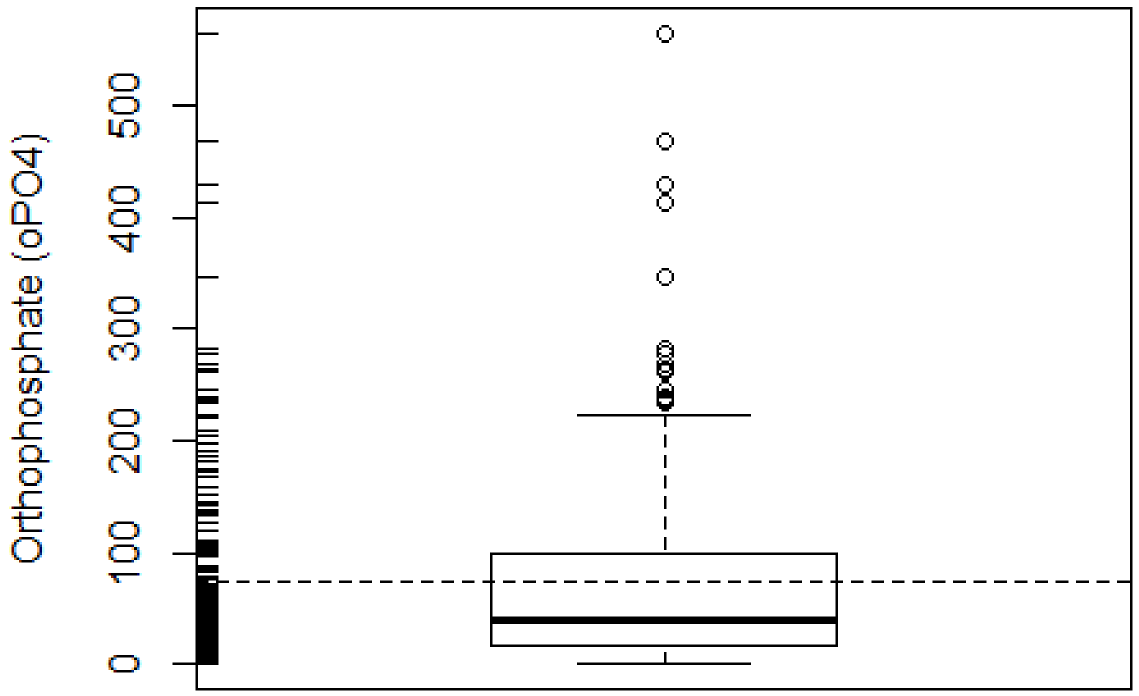
```
> values = c(1,1,1,1,1,1,2)
> boxplot(values, ylim=c(0.7,2.2))
> set.seed(7)
> rug(jitter(values), side=2)
```

```
> set.seed(7)
> jitter(values)
[1] 1.1955637   0.9590982   0.8462791   0.8278995   0.8974998   1.1168042   1.9360249
```

Data Visualisation and Summarisation

- Or boxplot

```
> boxplot(algae$oPO4, ylab='Orthophosphate (oPO4)')
> abline(h=mean(algae$oPO4,na.rm=T),lty=2)
> rug(jitter(algae$oPO4),side=2)  #side=2 - left,  3-up,  4-right, 1-bottom
```



jitter:
Add a small amount of noise to a numeric vector.

rug:
Adds a set of tick marks along the base of a plot.

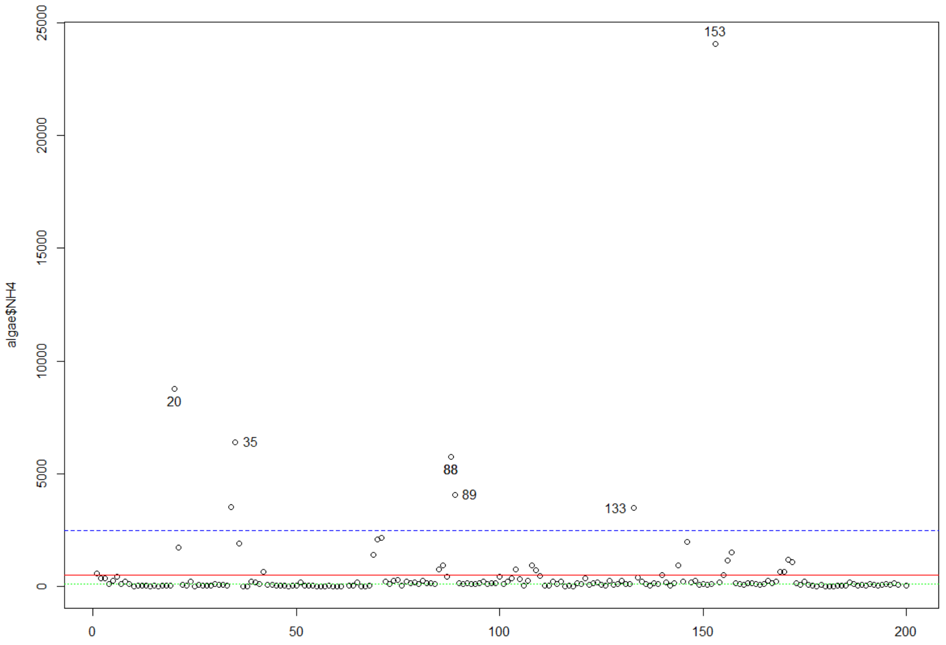
Data Visualisation and Summarisation

- Detect outliers with graphics

```
>plot(algae$NH4,xlab='')
>abline(h=mean(algae$NH4,na.rm=T),lty=1,col="red")
>abline(h=mean(algae$NH4,na.rm=T)+sd(algae$NH4,na.rm=T),lty=2,col="blue")
>abline(h=median(algae$NH4,na.rm=T),lty=3,col="green")
>identify(algae$NH4)
```

# `identify` is interactive: when a user click on the plotted dots with the left mouse, the row number of that observation will be shown. Click right mouse to finish interaction.

`identify` might not work in RStudio, though. Try the original R tool instead.



- Detect outliers without graphics

```
> algae[algae$NH4 >19000,]
      season size   speed mxPH mnO2 Cl      NO3     NH4    oPO4 PO4 Chla a1   a2 a3 a4   a5   a6    a7
NA    <NA>   <NA> <NA>   NA    NA    NA      NA     NA     NA   NA   NA   NA   NA NA  NA  NA   NA    NA
153  autumn medium high 7.3 11.8 44.205 45.65   24064 44    34   53.1 2.2 0   0   1.2 5.9 77.6 0
NA.1 <NA>   <NA> <NA>   NA    NA    NA      NA     NA     NA   NA   NA   NA   NA NA  NA  NA   NA    NA
```

Data Preprocessing

Dealing with Missing Values

Dealing With Unknown Values

- Unknown (missing) values
  - are common in real-world problems
  - may preclude the use of certain statistical learning approaches
    - E.g., randomForest(), mean()

- Solutions
  - Remove the cases with unknowns
  - Fill in the unknown values by exploring the most frequent value
  - Fill in the unknown values by exploring the correlations between variables
  - Fill in the unknown values by exploring the similarity between cases
  - Use tools that are able to handle these values

Removing the Obs. with Unknown Values

- Before removing them, check/count them first

```
> library(DMwR)
> data(algae)          # load fresh data again before we try different ways of dealing with unknown values
> algae[!complete.cases(algae),]        # check whether each obs is complete or not
    season   size  speed mxPH mnO2    Cl   NO3 NH4    oPO4     PO4  Chla   a1   a2  a3   a4  a5  a6  a7
28  autumn  small   high 6.80 11.1 9.000 0.630  20   4.000      NA  2.70 30.3  1.9 0.0  0.0 2.1 1.4 2.1
38  spring  small   high 8.00   NA 1.450 0.810  10   2.500   3.000  0.30 75.8  0.0 0.0  0.0 0.0 0.0 0.0
48  winter  small    low   NA 12.6 9.000 0.230  10   5.000   6.000  1.10 35.5  0.0 0.0  0.0 0.0 0.0 0.0
55  winter  small   high 6.60 10.8    NA 3.245  10   1.000   6.500    NA 24.3  0.0 0.0  0.0 0.0 0.0 0.0
56  spring  small medium 5.60 11.8    NA 2.220   5   1.000   1.000    NA 82.7  0.0 0.0  0.0 0.0 0.0 0.0
57  autumn  small medium 5.70 10.8    NA 2.550  10   1.000   4.000    NA 16.8  4.6 3.9 11.5 0.0 0.0 0.0
58  spring  small   high 6.60  9.5    NA 1.320  20   1.000   6.000    NA 46.8  0.0 0.0 28.8 0.0 0.0 0.0
59  summer  small   high 6.60 10.8    NA 2.640  10   2.000  11.000    NA 46.9  0.0 0.0 13.4 0.0 0.0 0.0
60  autumn  small medium 6.60 11.3    NA 4.170  10   1.000   6.000    NA 47.1  0.0 0.0  0.0 0.0 1.2 0.0
61  spring  small medium 6.50 10.4    NA 5.970  10   2.000  14.000    NA 66.9  0.0 0.0  0.0 0.0 0.0 0.0
62  summer  small medium 6.40   NA    NA    NA  NA      NA  14.000    NA 19.4  0.0 0.0  2.0 0.0 3.9 1.7
63  autumn  small   high 7.83 11.7 4.083 1.328  18   3.333   6.667    NA 14.4  0.0 0.0  0.0 0.0 0.0 0.0
116 winter medium   high 9.70 10.8 0.222 0.406  10  22.444  10.111    NA 41.0  1.5 0.0  0.0 0.0 0.0 0.0
161 spring  large    low 9.00  5.8    NA 0.900 142 102.000 186.000 68.05  1.7 20.6 1.5  2.2 0.0 0.0 0.0
184 winter  large   high 8.00 10.9 9.055 0.825  40  21.083  56.091    NA 16.8 19.6 4.0  0.0 0.0 0.0 0.0
199 winter  large medium 8.00  7.6    NA    NA  NA      NA      NA    NA  0.0 12.5 3.7  1.0 0.0 0.0 4.9
>
> nrow(algae[!complete.cases(algae),])
[1] 16
>
> algae <- na.omit(algae)
[1] 184
```

Removing the Obs. with Unknown Values

- Probably think twice before removing so many observations

```
> library(DMwR)
> data(algae)                  # load fresh data again before we try different ways of dealing with unknown values
> algae[!complete.cases(algae),]     # check whether each observation is complete or not
    season   size  speed mxPH mnO2    Cl    NO3 NH4    oPO4     PO4  Chla   a1   a2  a3   a4  a5  a6  a7
28  autumn  small   high 6.80 11.1 9.000 0.630  20   4.000      NA  2.70 30.3  1.9 0.0  0.0 2.1 1.4 2.1
38  spring  small   high 8.00   NA 1.450 0.810  10   2.500   3.000  0.30 75.8  0.0 0.0  0.0 0.0 0.0 0.0
48  winter  small    low   NA 12.6 9.000 0.230  10   5.000   6.000  1.10 35.5  0.0 0.0  0.0 0.0 0.0 0.0
55  winter  small   high 6.60 10.8    NA 3.245  10   1.000   6.500    NA 24.3  0.0 0.0  0.0 0.0 0.0 0.0
56  spring  small medium 5.60 11.8    NA 2.220   5   1.000   1.000    NA 82.7  0.0 0.0  0.0 0.0 0.0 0.0
57  autumn  small medium 5.70 10.8    NA 2.550  10   1.000   4.000    NA 16.8  4.6 3.9 11.5 0.0 0.0 0.0
58  spring  small   high 6.60  9.5    NA 1.320  20   1.000   6.000    NA 46.8  0.0 0.0 28.8 0.0 0.0 0.0
59  summer  small   high 6.60 10.8    NA 2.640  10   2.000  11.000    NA 46.9  0.0 0.0 13.4 0.0 0.0 0.0
60  autumn  small medium 6.60 11.3    NA 4.170  10   1.000   6.000    NA 47.1  0.0 0.0  0.0 0.0 1.2 0.0
61  spring  small medium 6.50 10.4    NA 5.970  10   2.000  14.000    NA 66.9  0.0 0.0  0.0 0.0 0.0 0.0
62  summer  small medium 6.40   NA    NA    NA  NA      NA  14.000    NA 19.4  0.0 0.0  2.0 0.0 3.9 1.7
63  autumn  small   high 7.83 11.7 4.083 1.328  18   3.333   6.667    NA 14.4  0.0 0.0  0.0 0.0 0.0 0.0
116 winter medium   high 9.70 10.8 0.222 0.406  10  22.444  10.111    NA 41.0  1.5 0.0  0.0 0.0 0.0 0.0
161 spring  large    low 9.00  5.8    NA 0.900 142 102.000 186.000 68.05  1.7 20.6 1.5  2.2 0.0 0.0 0.0
184 winter  large   high 8.00 10.9 9.055 0.825  40  21.083  56.091    NA 16.8 19.6 4.0  0.0 0.0 0.0 0.0
199 winter  large medium 8.00  7.6    NA    NA  NA      NA      NA    NA  0.0 12.5 3.7  1.0 0.0 0.0 4.9
>
> manyNAs(algae)          #returns the row numbers that have more than 20% of the columns with an NA. In this case, 18*20% = 3.6 columns.
[1]  62 199
> algae <- algae[-c(62,199),]
> algae <- algae[-manyNAs(algae),]      # the last two commands have the same effect
```

Filling with Most Frequent Values

- Several alternatives can be chosen, with different trade-offs between
  - the level of approximation, and
  - the computational complexity of the method

- First alternative (simplest and fastest)
  - Use some statistics of centrality to fill in the unknown values
    - mean, median, mode, etc
      - choose mean if the distribution is nearly normal
      - choose median if not
  - For example,

This method is simple, fast, thus appealing for large dataset. However, it may introduce a large bias in the data.

```
    season    size   speed mxPH mnO2    Cl    NO3  NH4    oPO4     PO4  Chla    a1   a2   a3   a4   a5   a6   a7
48  winter   small     low   NA 12.6 9.000 0.230   10    5.000   6.000  1.10 35.5  0.0  0.0  0.0  0.0  0.0  0.0
```

Recall that the mxPH is nearly normal distributed, we could use its mean value to fill in the hole.

```
>algae[48,'mxPH'] <- mean(algae$mxPH,na.rm=T)
```

#calculate the mean of the mxPH column while ignoring any NA values in this column

Filling by Exploring Correlations

Birkbeck UNIVERSITY OF LONDON

- An alternative to get less biased estimators for unknowns:
  - to explore the relationships between variables

```
> cor(algae[,4:18],use="complete.obs")
```
#disregard obs with NAs

```
> symnum(cor(algae[,4:18],use="complete.obs"))
```
#Symbolically encode a given numeric or logical vector or array

```
     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            .  .                     .
a7
attr(,"legend")
[1]  0 ' ' 0.3 '.' 0.6 ',' 0.8 '+' 0.9 '*' 0.95 'B' 1
```

NH4 and NO3 are positively correlated (0.72)

PO4 and oPO4 are highly correlated (above 0.9)

According to the domain expert, this was expected because the value of the total PO4 includes the value of oPO4

We will find the form of the linear correlation between these variables.

How to Find Linear Relationship

- Find linear relationship between PO_4 and oPO_4

```
> data(algae)
> algae <- algae[-manyNAs(algae),]
> lm(PO4 ~ oPO4, data=algae)

Call:
lm(formula = PO4 ~ oPO4, data = algae)

Coefficients:
(Intercept)          oPO4
     42.897         1.293
```

The linear model we have obtained is $PO_4 = 42.897 + 1.293 \times oPO_4$

- With this formula, we can fill in the unknown values of these unknowns, provided they are not both unknown.
  - Remove the observations with both unknown (sample 62, 199)
  - We have a single observation with an unknown value on PO_4 (sample 28)

Use the Linear Model to Predict

- Use $PO_4 = 42.897 + 1.293 \times oPO_4$ to predict the unknown PO4 at sample 28

```
> algae[28,'PO4'] <- 42.897 + 1.293 * algae[28,'oPO4']
> algae[28,]
   season  size speed mxPH mnO2 Cl  NO3 NH4 oPO4    PO4 Chla   a1   a2 a3 a4   a5   a6   a7
28 autumn small  high  6.8 11.1  9 0.63  20    4 48.069  2.7 30.3 1.9  0  0 2.1 1.4 2.1
```

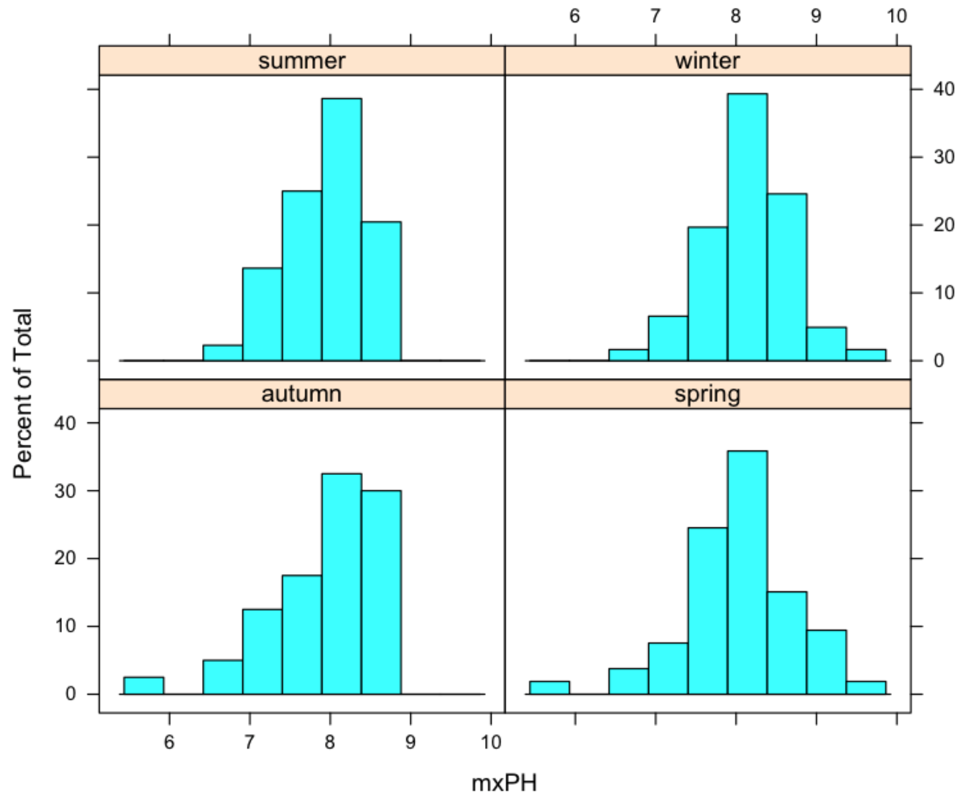
- This can be generalised to fill all missing PO_4 values (if any)

```
> data(algae)
> algae <- algae[-manyNAs(algae),]    # delete both unknowns
> fillPO4 <- function(oP) {
     if (is.na(oP))
          return(NA)          #if oPO4's value not available
     else
          return(42.897 + 1.293 * oP)  #else return the result derived by linear model
  }
> algae[is.na(algae$PO4),'PO4'] <-
        sapply(algae[is.na(algae$PO4),'oPO4'],fillPO4)
#This function is applied to all samples with unknown value on the variable PO4
```

Filling by Exploring Correlations

- For other observations with unknown values, we can explore the correlations between the variables and the nominal variables of this problem.
  - E.g., mxPH and season

```
> histogram(~ mxPH | season, data=algae)
```

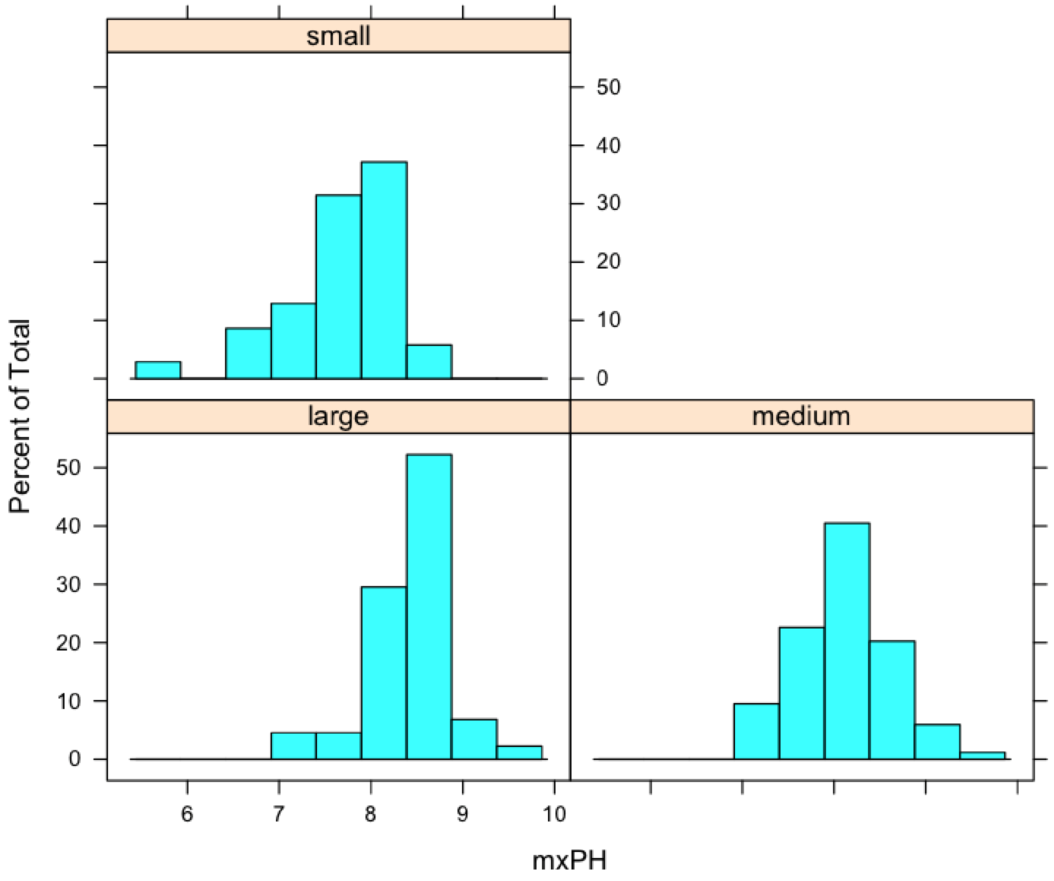


The values of mxPH are not seriously influenced by the season of the year when the samples were collected.

Filling by Exploring Correlations

- If we try the same using the size of the river

```
> histogram(~ mxPH | size, data=algae)
```



What tendency can you observe?

Filling by Exploring Similar Cases

- Another alternative is to use the similarities between the rows to fill in the unknown values
  - If two water samples are similar, and one of them has an unknown value
    - It's very probable that this value is similar to the value of the other sample
  - How to define distance? Which distance can you think of?
    - Euclidean distance!
  - Approach:
    - Find ten most similar cases of any water sample with some unknown value
    - Use their values to fill in the unknown
      - The median of the values of the ten nearest neighbours

```
> algae <- knnImputation(algae,k=10,meth='median')
```

      - The weighted average of the values of the neighbours
        - » The further a neighbour is, the less weight it has (usual weight: 1/d)

```
> algae <- knnImputation(algae,k=10) #in DMwR package
```

using this function will replace all missing values in the df

Obtaining Prediction Models

Multiple Linear Regression

Regression Trees

Multiple Linear Regression

- The implementation of linear regression in R is not able to use datasets with unknown values
  - Use the knn-preprocessed technique to fill in the unknowns.

```
> data(algae)
> algae <- algae[-manyNAs(algae), ]
> clean.algae <- knnImputation(algae, k = 10)
```

- Multiple linear regression

```
> lm.a1 <- lm(a1 ~ .,data=clean.algae[,1:12]) # here consider a1 with other 11 predictors
……
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
……
```

Nominal variables are encoded by dummy variables

Erh, where are seasonautumn, sizelarge and speedhigh?

these are the base case (n-1 variables)

Measures Explained

```
Coefficients:
              Estimate Std. Error t value Pr(>|t|)← p value
(Intercept)  42.942055  24.010879   1.788  0.07537 .
……
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
```

- t test: to see whether each coefficient is statistically significant (hypothesis H0: $\beta_i=0$)
- Pr(>|t|): a value 0.0001 means that we are 99.99% confident that the coefficient is not null
  - **Large value → insignificant factor**, small value → significant factor (notice those with *'s by R)

Measures Explained

- R^2 coefficients (multiple and adjusted)
  - Degree of fit of the model
    - pve: proportion variance explained (the smaller, the lack of fit)
    - The adjusted coefficient is more demanding, as it takes into account the number of parameters in the model

```
Multiple R-squared:  0.3731,  Adjusted R-squared:  0.3215
```

- F-statistics and p-value
  - To test H0: $\beta_1 = \beta_2 = \ldots = \beta_m = 0$
    (the target variable doe not depend on any of the predictors)
    - p-value: 0.0001 means that we are 99.99% confident that the null hypothesis is not true.
      - If p value is too high (>0.1), it makes no sense to look at the t-test on individual coefficients

```
F-statistic: 7.223 on 15 and 182 DF,  p-value: 2.444e-12
```

Simply the Linear Model

- Some predictors have a small significance, we could eliminate them from the model

```
> anova(lm.a1)  # ANOVA: analysis of variance
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
```

It will give us the reduction in the residual sum of squares when adding each variable in turn

`season` contributes the least to the reduction of the fitting error of the model

Update the Model

- Remove season from the model

```
> lm2.a1 <- update(lm.a1, . ~ . - season)
> summary(lm2.a1)
```

keep all x, minus season

```
Call:
lm(formula = a1 ~ size + speed + mxPH + mnO2 + Cl + NO3 + NH4 +
    oPO4 + PO4 + Chla, data = clean.algae[, 1:12])

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) 44.9532874 23.2378377   1.934  0.05458 .
……
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
……
Multiple R-squared:  0.3682,      Adjusted R-squared:  0.3272
F-statistic: 8.984 on 12 and 185 DF,  p-value: 1.762e-13
```

This fit has improved a bit, but still not too impressive

Previously, `adjusted R-squared:0.3215`

Further AVONA Analysis

- Comparison between the two 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
```

→ so much smaller than the first model

The second model is better, as it has a smaller sum of squares

However, with Pr(>F)=0.6971, it means that only with around 30% confidence we can say the two models are different

→ In other words, the difference between the two models are not significant

→ The second model is simpler

because there is one less feature (the season column)

Automatic Model Simplification

- The `step` function will show you how to simplify the linear model step by step

```
> final.lm=step(lm.a1)
Start:  AIC=1152.03          #AIC stands for Akaike Information Criterion
a1 ~ season + size + speed + mxPH + mnO2 + Cl + NO3 + NH4 + oPO4 + PO4 + Chla

#omit several steps in the middle, the last step is:

Step:  AIC=1140.38           # step function use AIC to perform model search
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
```

AIC offers an estimate of the relative information lost when a given model is used to represent the process that generated.

It tells nothing about the absolute quality of a model, only the quality relative to other models. Thus, if all the candidate models fit poorly, AIC will not give any warning of that.

Analyse the Final 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
```

In multiple regression, the proportion of variance explained (PVE) is equal to (adjusted) R^2.

The PVE is still not very interesting (0.3324).

A sign that linearity assumption of this model is inadequate for the domain.

Obtaining Prediction Models

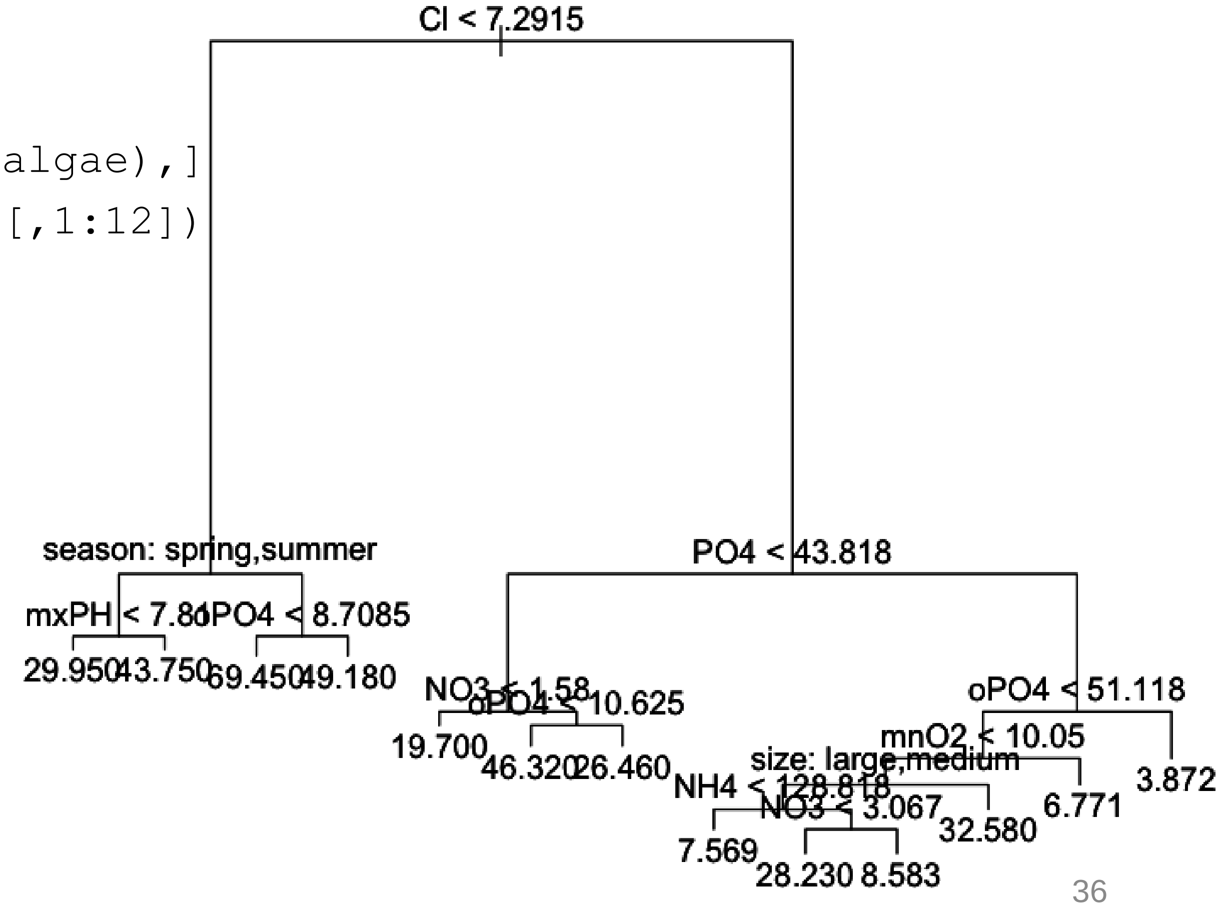
Multiple Linear Regression

Regression Trees

Build a Regression Tree

- It can be done in the same way as building a classification tree

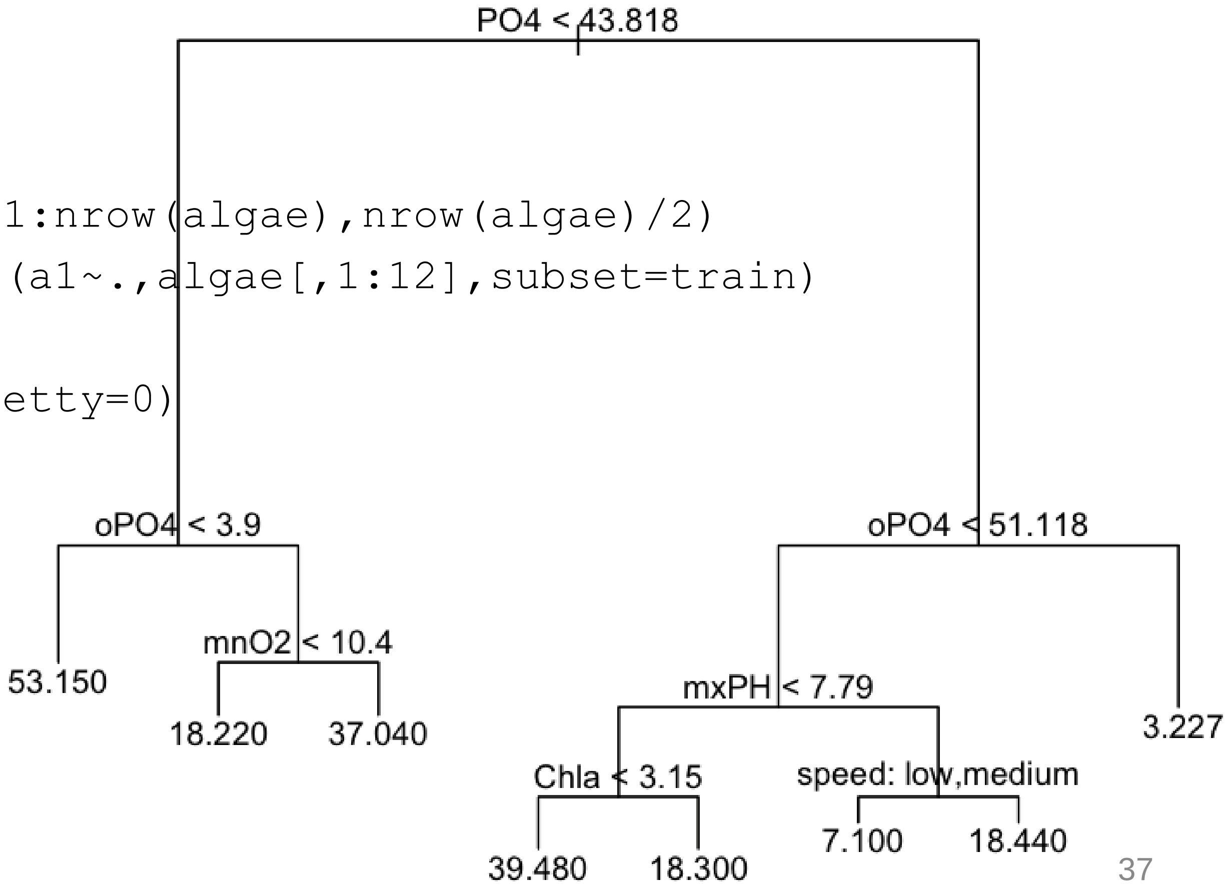
```
> library(tree)
> data(algae)
> algae<-algae[-manyNAs(algae),]
> rt.a1<-tree(a1~.,algae[,1:12])
> text(rt.a1,pretty=0)
```



Build the Tree using Train Part

- Now we randomly sample a train set and build the regression tree based on the set

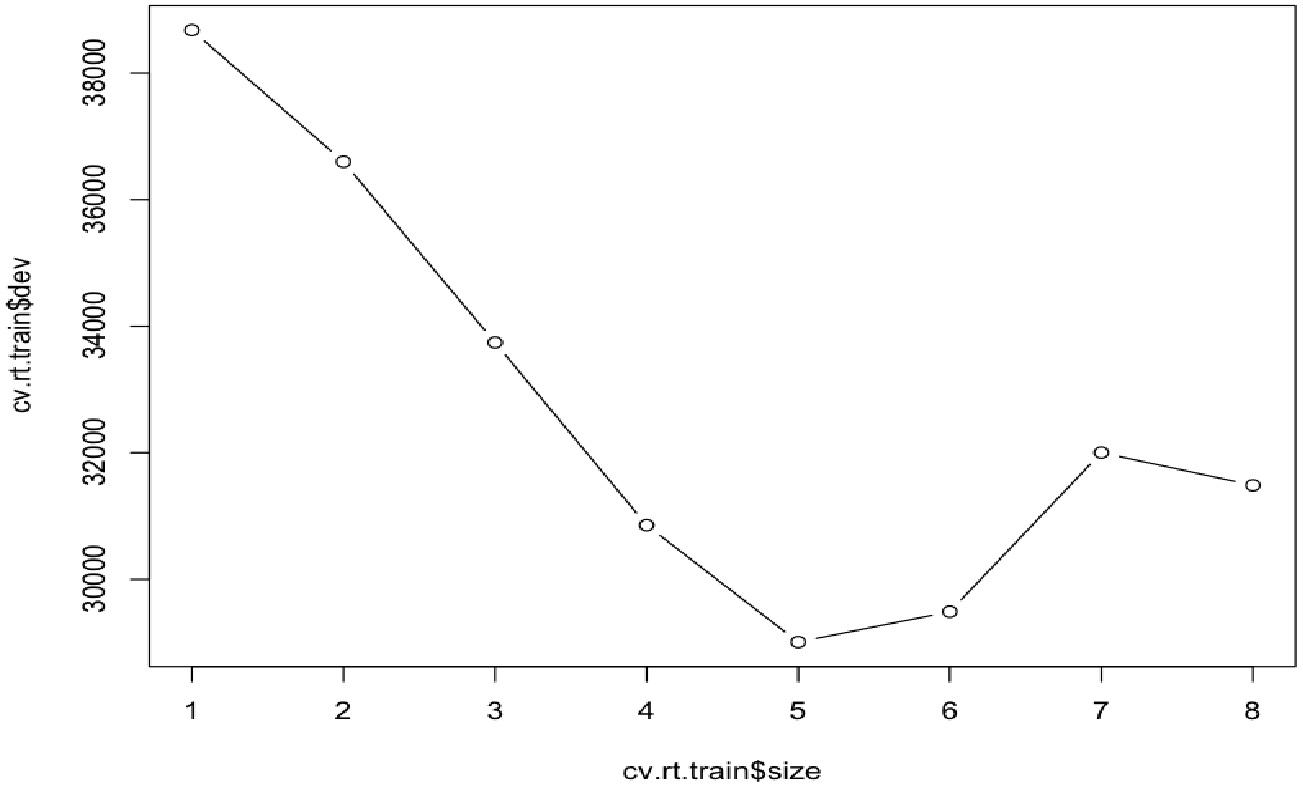
```
> nrow(algae)
[1] 198
> set.seed(2)
> train.a1 <- sample(1:nrow(algae),nrow(algae)/2)
> rt.a1.train <- tree(a1~.,algae[,1:12],subset=train)
> plot(rt.a1.train)
> text(rt.a1.train,pretty=0)
```



Use CV to Check Whether to Prune

- Cross validation is used to see whether the tree `rt.a1.train` needs to be pruned

```
> cv.rt.train <- cv.tree(rt.a1.train)
> plot(cv.rt.train$size,cv.rt.train$dev,type='b')
```

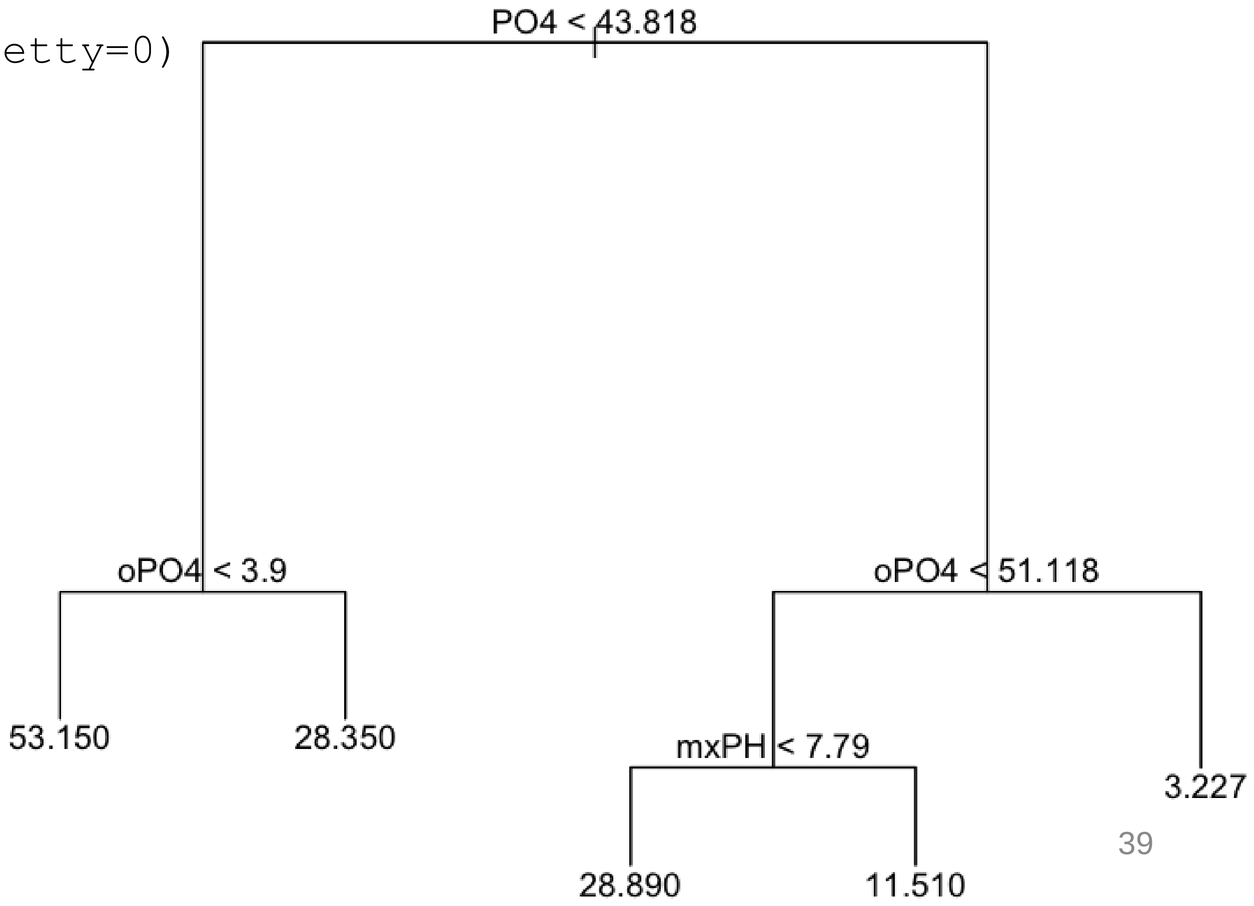


The best tree (the one with the minimum MSE) is of the size 5

Prune the Tree

- Prune the tree to be of size 5:

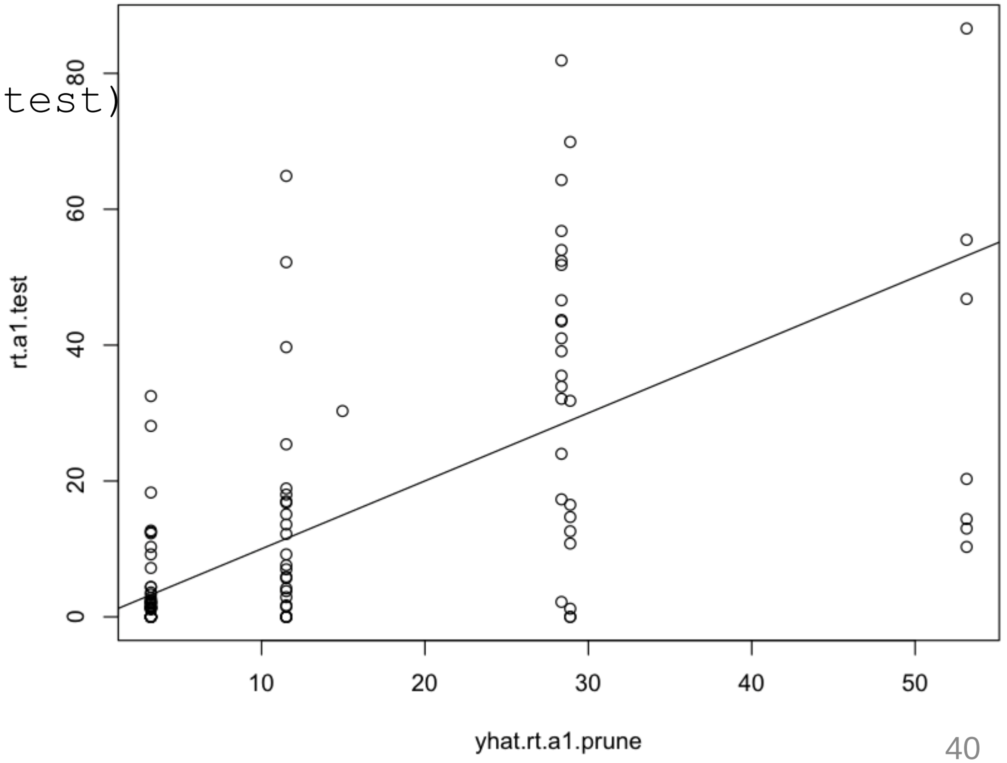
```
> prune.rt.a1 <- prune.tree(rt.a1.train,best=5)
> plot(prune.rt.a1)
> text(prune.rt.a1,pretty=0)
```



Performance Evaluation – Regression Tree

- We use the test part to evaluate the performance

```
> rt.a1.test <- algae[-train,"a1"]
> yhat.rt.a1.prune <- predict(prune.rt.a1,newdata=algae[-train,1:12])
> mean((yhat.rt.a1.prune-rt.a1.test)^2)
[1] 297.0548
> plot(yhat.rt.a1.prune,rt.a1.test)
> abline(0,1)
```



Using Bagging

- Since the bagging/randomForest method requires no missing values, we start from the dataset `clean.algae`

```
> set.seed(20)
> bag.train <- sample(1:nrow(clean.algae),99)
> bag.a1.train <- randomForest(a1~.,clean.algae[1:12],
                                                    subset=bag.train,mtry=11,importance=T)
> bag.a1.train
Call:
 randomForest(formula = a1 ~ ., data = clean.algae[, 1:12], mtry =11,
                  importance = T, subset = bag.train)
                  Type of random forest: regression
                        Number of trees: 500
No. of variables tried at each split: 11

          Mean of squared residuals: 271.161
                    % Var explained: 42.9
```

This PVE is larger than the linear model when compared to the adjusted R-squared for the linear model, which is around 30%

Performance Evaluation - Bagging

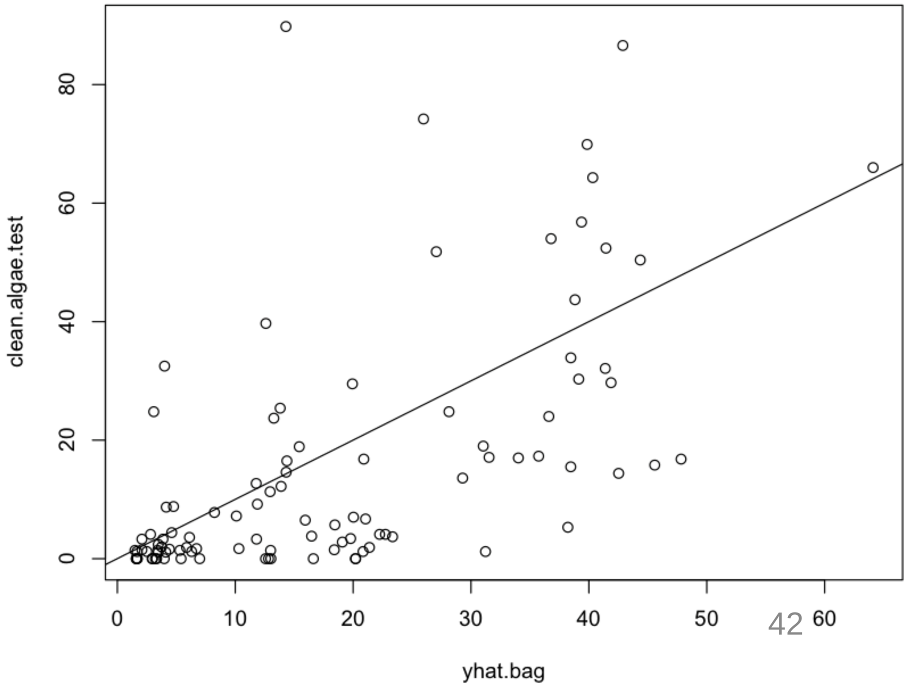
- How well does this bagged model perform on the test set?

```
> yhat.bag <- predict(bag.a1.train,newdata=clean.algae[-bag.train,1:12])
> clean.algae.test <- clean.algae[-bag.train,"a1"]
> mean((yhat.bag-clean.algae.test)^2)
[1] 279.9227
> plot(yhat.bag,clean.algae.test)
> abline(0,1)
```

This MSE is smaller than the best tree

This looks better than in the regression tree →

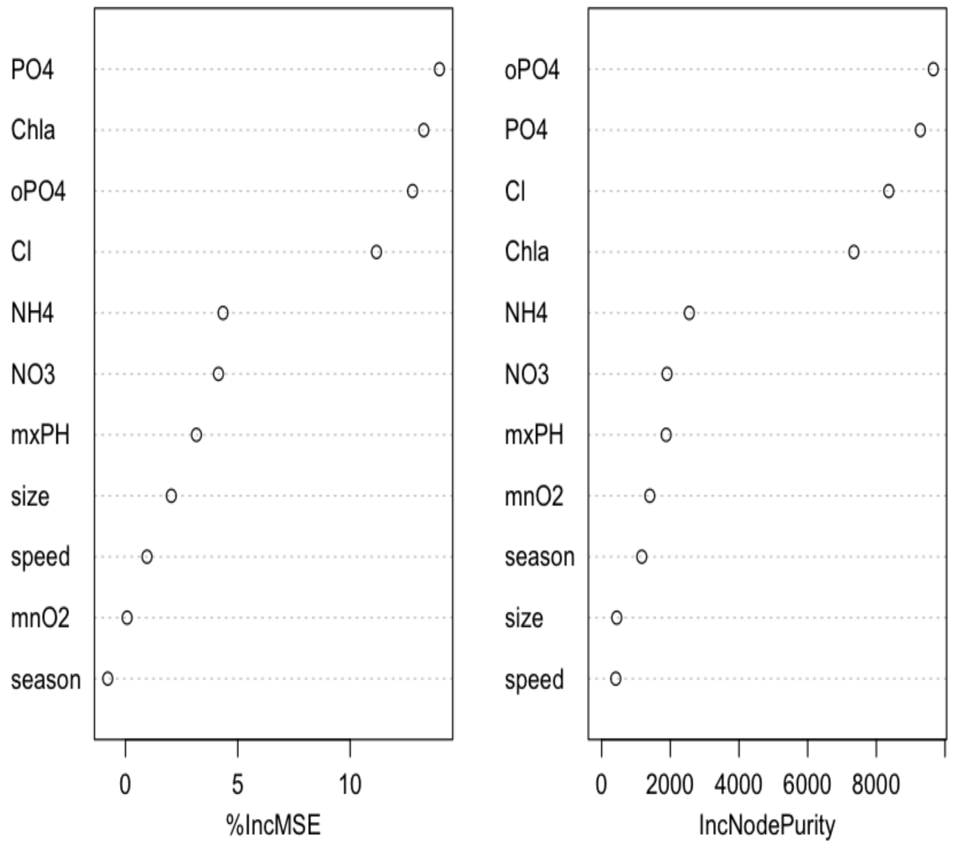
You may play with the number of trees in the bagging at home (ntree=i)



Which Predictors are Important?

```
> importance(bag.a1.train)
           %IncMSE IncNodePurity
season -0.78908700     1168.1408
size    2.04011486      439.9201
speed   0.95764449      410.7687
mxPH    3.15721118     1878.4553
mnO2    0.07241614     1402.3103
Cl     11.17164253     8361.5771
NO3     4.13772152     1904.3340
NH4     4.34200936     2552.3389
oPO4   12.77819198     9659.2113
PO4    13.97331416     9281.1592
Chla   13.27516985     7345.5980

> varImpPlot(bag.a1.train)
```



Using Random Forest

- Choose a smaller mtry value, usually p/3 when building a random forest for regression trees
  - mtry = 11/3 ≈ 3 or 4

```
> set.seed(20)
> rf.a1.train.4 <- randomForest(a1~.,clean.algae[,1:12], subset=bag.train,
                                        mtry=4,importance=T)
> yhat.rf <- predict(rf.a1.train.4,newdata=clean.algae[-bag.train,1:12])
> mean((yhat.rf-clean.algae.test)^2)
[1] 273.3071
```

mtry = 3 is slightly better

You may find the best mtry at home

```
> set.seed(20)
> rf.a1.train.3 <- randomForest(a1~.,clean.algae[,1:12],subset=bag.train,
                                        mtry=3,importance=T)
> yhat.rf <- predict(rf.a1.train.3,newdata=clean.algae[-bag.train,1:12])
> mean((yhat.rf-clean.algae.test)^2)
[1] 272.3034
```

The PVE of rf.a1.train.3 is 49.6% (use summary()), still not very fit

Probably try nonlinear models (polynomials, etc), something for you to try at home, too

Prediction for New Test Set

Prediction for the Algae

- We are given 140 test samples, whose algae levels are unknown.

- We will choose the best models to obtain these predictions.
  - To obtain unbiased estimates of MSE for a set of models
    - By means of a cross-validation experimental process
    - For simplicity, we only predict `a1`

- For `a1`, we have already shown that the randomForest model `rf.a1.train.3` is the best model
  - Use `rf.a1.train.3` to make the prediction

Unknowns in the Test Data

- There are unknowns in the test data

- We could use `knnImputation()` as in the training dataset
  - Use other test cases to fill in the unknowns → not ideal

  - Use training data to find the neighbours instead
    - use `knnImputation()`, but with an extra argument

```
> clean.test.algae <- knnImputation(test.algae,k=10,distData=algae[,1:11])
```

# The `distData` argument allows you to supply an extra set of data (i.e., the training dataset) where the ten nearest neighbours are to be found for each case with unknowns in the `test.algae` dataset.

Make the Prediction

- Finally,…

```
> preds <- rep(0,140)
> preds <- predict(rf.a1.train.3, newdata=clean.test.algae, mtry=3, importance=T)
> preds
        1         2         3         4         5         6         7         8         9 
 7.266943 10.458083 13.387457 13.542400 27.145823 33.591357 35.073133 37.611920 38.065740 
       10        11        12        13        14        15        16        17        18 
36.190503 10.706703 15.288940 40.884010 38.163287 37.630820 26.044157 10.487700 20.337720 
       19        20        21        22        23        24        25        26        27 
40.361043 54.538080  6.965607  4.724927  4.981443 11.896803  6.452217  5.023043 24.228200 
       28        29        30        31        32        33        34        35        36 
43.114077 27.373763 23.633090 26.444843 20.911110 32.294507 38.157100 55.714590 35.624243 
       37        38        39        40        41        42        43        44        45 
35.052197 51.597240 33.467427 39.437900 37.612970 16.618960 10.317370  9.975300 10.411817 
       46        47        48        49        50        51        52        53        54 
 3.337300 10.015007  5.438577 17.838527 31.355300 11.017717  3.678907  5.509753  3.913507 
       55        56        57        58        59        60        61        62        63 
 4.779007 12.729870 13.189073 11.902373 17.185123 14.290100  6.853557 21.050563 16.727573 
       64        65        66        67        68        69        70        71        72 
 8.964617 33.982597 27.070277 18.403937 40.085983 43.577550  4.610323  6.584670  4.338993 
 ......
```

Birkbeck
UNIVERSITY OF LONDON