Statistical Modeling in R

One big part of statistics is fitting a *model* to data. R has many functions for doing this, but I'll mention only `lm`, which fits a linear model.

Models in R are often specified using *formulas*, that say how one thing is modelled in terms of other things.

For `lm`, we want to specify that some *response* variable is modelled as a linear combination (plus noise) of some *explanatory* variables. This is done using a formula such as

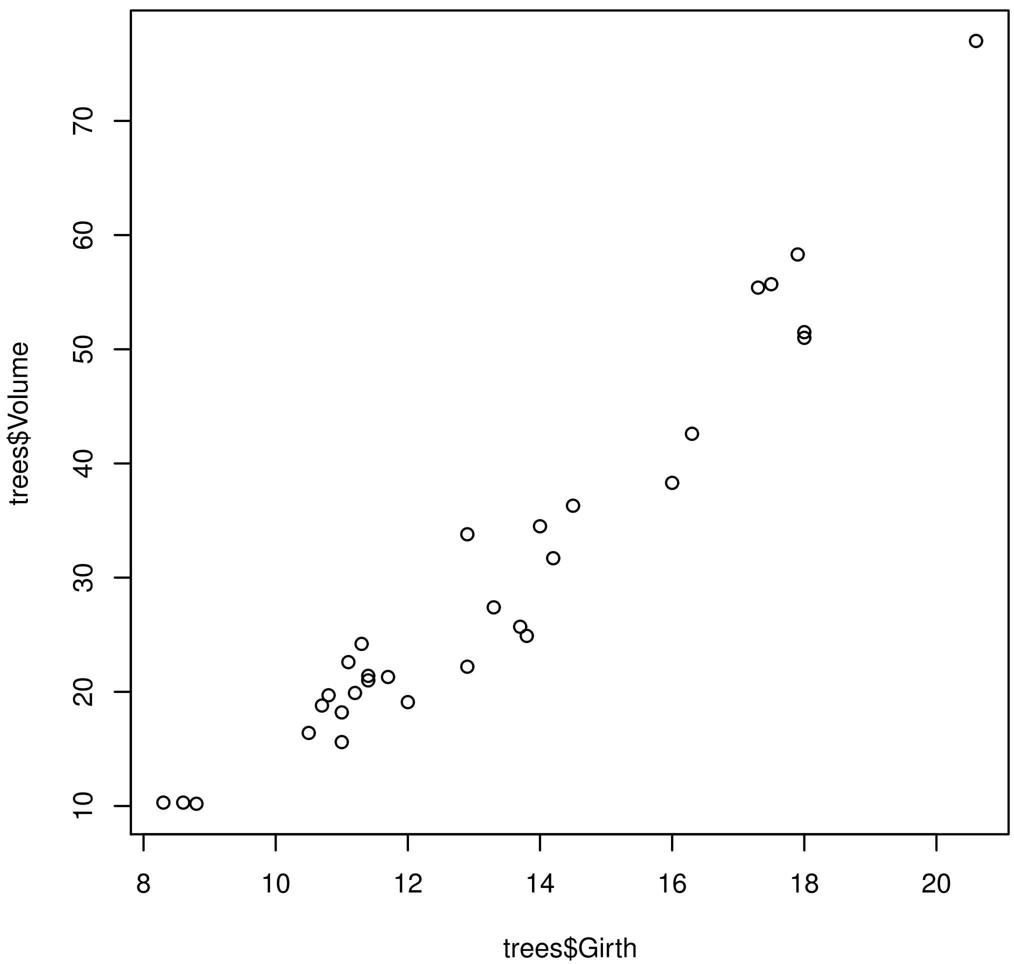
```
growth ~ ave_temp + fertilizer + variety
```

This might express that the amount by which some plant grows is linearly related to the average temperature, the amount of fertilizer used, and a set of indicator variables indicating the variety of the plant.

A Simple Example of a Linear Model

Here, I'll show the results of a very simple linear model, relating the volume of wood in a cherry tree to its girth (diameter of trunk). The data is in the data frame `trees` that comes with R.

Here's a plot of the data:



Fitting the Model with `lm`

We can fit a linear model for volume given girth as follows:

```
> lm (trees$Volume ~ trees$Girth)

Call:
lm(formula = trees$Volume ~ trees$Girth)

Coefficients:
(Intercept)  trees$Girth
    -36.943        5.066
```

The result says that best fit model for the volume is

$$\text{Volume} \;=\; -36.943 \;+\; 5.066\,\text{Girth} \;+\; \text{noise}$$

We can get the same result with an abbreviated formula by saying the data comes from the data frame `trees`:

```
  lm (Volume ~ Girth, data=trees)
```

Using the Result of `lm`

The value returned by `lm` is an object of class `"lm"`, which has special methods for printing and other operations.

We can save the result, and then get the regression coefficients with `coef`.

```
> m <- lm (Volume ~ Girth, data=trees)
> coef(m)
(Intercept)       Girth
 -36.943459    5.065856
```

We could use these coefficients to predict the volume for a new tree, with girth of 11.6:

```
> coef(m) %*% c(1,11.6)   # %*% will compute the dot product
         [,1]
[1,] 21.82048
```

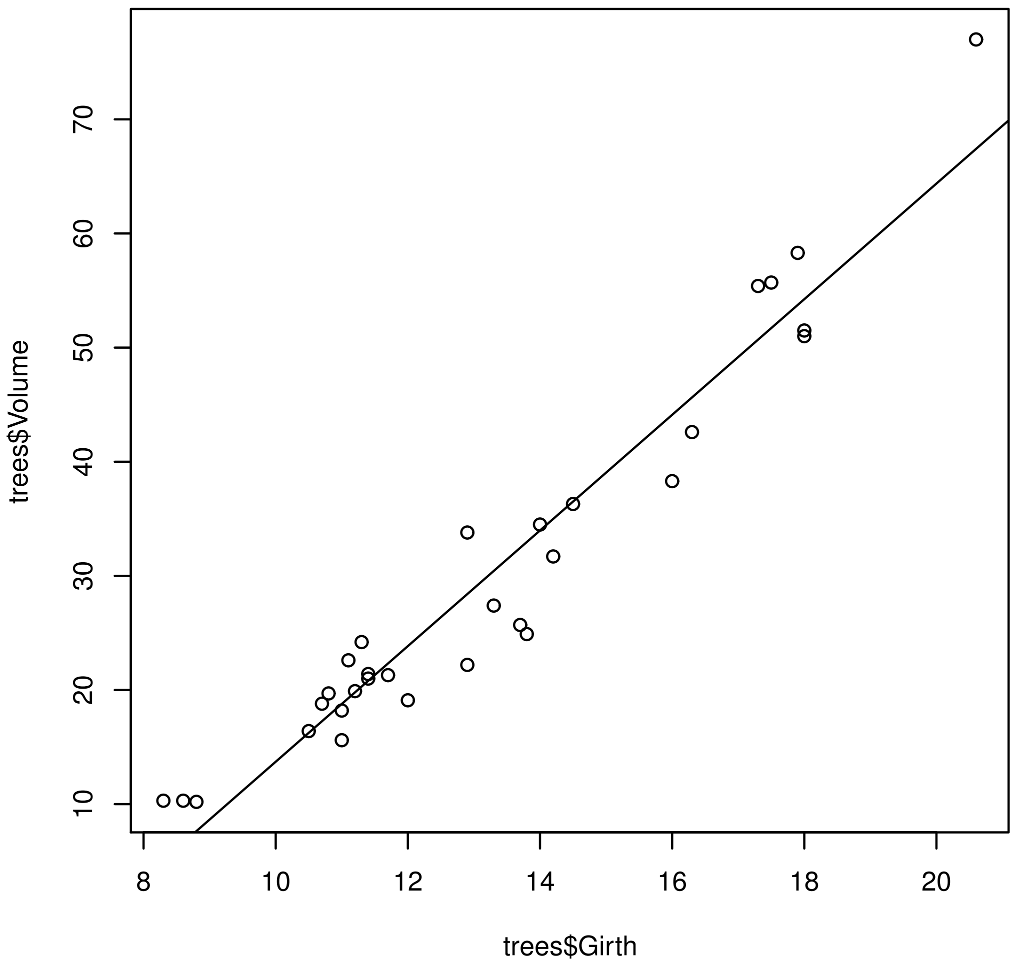
Getting More Details on the Model Fitted

We can also ask for more statistical details with `summary`:

```
> summary(m)

Call:
lm(formula = Volume ~ Girth, data = trees)

Residuals:
   Min     1Q Median     3Q    Max
-8.065 -3.107  0.152  3.495  9.587

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -36.9435     3.3651  -10.98 7.62e-12 ***
Girth         5.0659     0.2474   20.48  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 4.252 on 29 degrees of freedom
Multiple R-squared: 0.9353,    Adjusted R-squared: 0.9331
F-statistic: 419.4 on 1 and 29 DF,  p-value: < 2.2e-16
```

Plotting the Regression Line

We can also plot the regression line from the fitted model on top of a scatterplot of the data, using `abline(m)`:



The plot shows some indication that the relationship is actually curved.

Some Useful Functions of Vectors

The `unique` function returns a vector of unique values:

```
> colours <- c("red","blue","red","red","green","blue")
> unique(colours)
[1] "red"   "blue"  "green"
```

The `sort` function sorts a vector in increasing order (or decreasing order if you use `decreasing=TRUE`):

```
> ages <- c(4,9,12,2,4,9,10)
> sort(ages)
[1]  2  4  4  9  9 10 12
> sort(unique(ages),decreasing=TRUE)
[1] 12 10  9  4  2
```

The `which.min` and `which.max` functions give the index of the smallest and largest elements in a vector (first occurrence if they occur more than once):

```
> which.min(ages)
[1] 4
> which.max(ages)
[1] 3
```