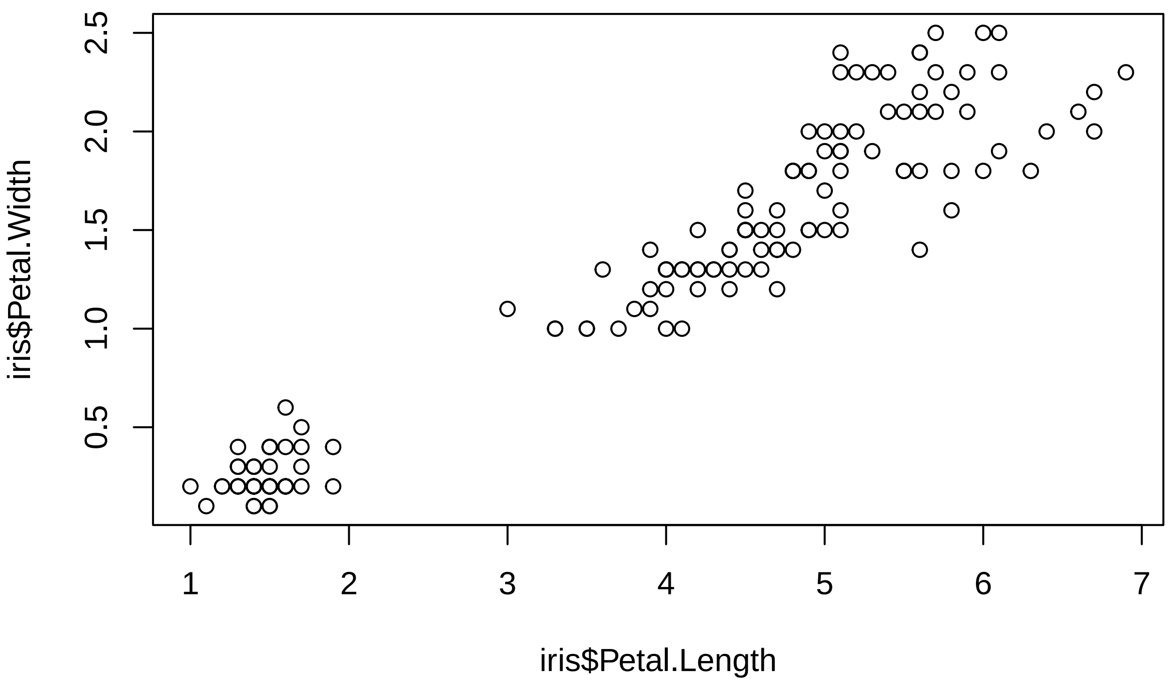
notes4

2022-06-11

Plot function on "iris" dataset

```
plot(x=iris$Petal.Length, y=iris$Petal.Width)
```



Petal.Width depends on Petal.Length, i.e. **Petal.Length** is the *regressor* (independent) variable that the (dependent) variable **Petal.Width** depends on.

Regression Set of statistical techniques to *predict* the value of 1+ *dependent* variables from the values of 1+ *independent* variables.

What kind of dependence is this (from a mathematical point of view)?

Linear regression

Petal.Width (Y) is a *linear function* of Petal.Length (X) plus random noise: $Y = beta_0 + beta_1 X + noise$

- $beta_0$ = width of a theoretical Petal.Length=0 (**intercept**).
- $beta_1$ = how much Petal.Width increases (in cm) for a 1 cm increase of Petal.Length (**slope**).
- *noise* = random noise that makes Petal.Width to vary (variation not explained by Petal.Length). In the plot above, noise is the Petal.Width variation at Petal.Length = 5 cm (i.e. Petal.Width varies between 1.5 and 2). We assume *noise* to be *normally distributed* with mean 0 and variance *sigma* that does not depend on X (**homoscedasticity**).

Mean Squared Error (MSE)

Mean of the square difference between predicted Y value and actual Y value:

```
beta0 <- -0.5
beta1 <- 0.5
predicted_Y <- beta0 + beta1*iris$Petal.Length
actual_Y <- iris$Petal.Width

mse <- mean((predicted_Y - actual_Y)^2)
```

We can use MSE values to compare different linear regressions and evaluate which one better fits data: this will have the lowest MSE (high MSE means bad regression).

Linear Model Function (lm)

`lm()` finds the best $beta_0$ and $beta_1$ values (i.e., those giving the lowest MSE):

```
lin_reg <- lm(formula = iris$Petal.Width ~ iris$Petal.Length)
lin_reg
```

```
##
## Call:
## lm(formula = iris$Petal.Width ~ iris$Petal.Length)
##
## Coefficients:
##       (Intercept)  iris$Petal.Length
##           -0.3631             0.4158
```

```
class(lin_reg)
```

```
## [1] "lm"
```

```
names(lin_reg)
```

```
##  [1] "coefficients" "residuals"     "effects"       "rank"
##  [5] "fitted.values" "assign"        "qr"            "df.residual"
##  [9] "xlevels"       "call"          "terms"         "model"
```

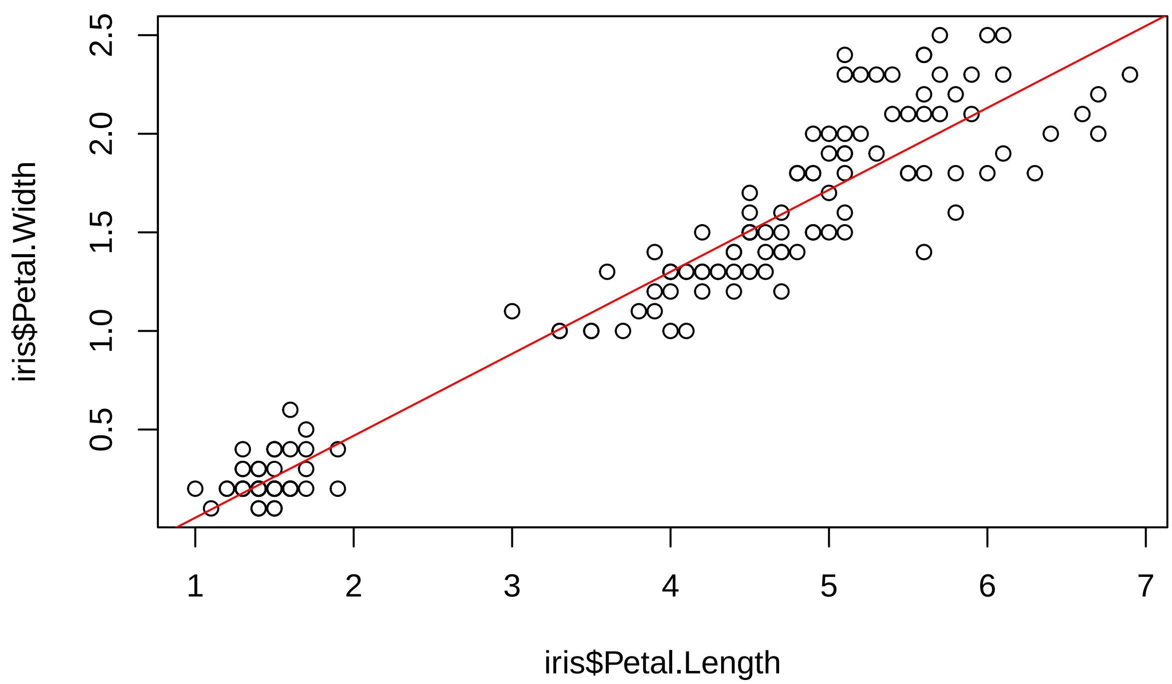
We find $beta_0$ and $beta_1$ values in "coefficients":

```
lin_reg$coefficients
```

```
##       (Intercept) iris$Petal.Length
##        -0.3630755         0.4157554
```

Petal.Width increases by 0.416 cm for every 1 cm increase of Petal.Length.

```
plot(x=iris$Petal.Length, y=iris$Petal.Width)
abline(a=lin_reg$coefficients[1], b=lin_reg$coefficients[2], col="red") #where lin_reg$coefficients[1]
```



This is the best possible *regression line*.

Other than "coefficients" there is the "residual" slot:

```
head(lin_reg$residuals)
```

```
##           1           2           3           4           5           6
## -0.01898206 -0.01898206  0.02259348 -0.06055760 -0.01898206  0.05629131
```

that are the differences between actual and predicted values of Y (Petal.Width), so that MSE is defined as:

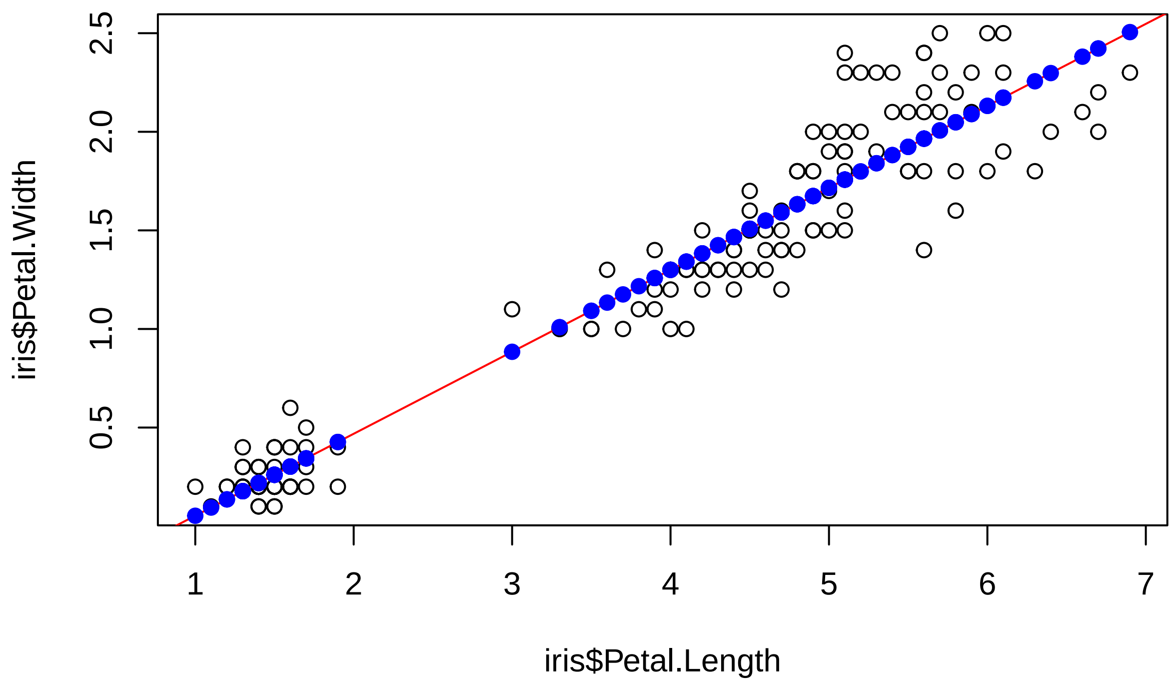
```
mse <- mean(lin_reg$residuals^2)
mse
```

```
## [1] 0.04206731
```

The "fitted.values" slot hosts the predicted values of Y (i.e. those that lie on the regression line, that make the line basically):

```
plot(x=iris$Petal.Length, y=iris$Petal.Width)
abline(a=lin_reg$coefficients[1], b=lin_reg$coefficients[2], col="red")

points(iris$Petal.Length, lin_reg$fitted.values, pch = 19, col = "blue")
```



Statistical significance

To quantify the uncertainty associated to the fitted (predicted) beta values, check the summary of the lm object lin_reg you just created:

```
summ <- summary(lin_reg)
summ
```

```
##
## Call:
## lm(formula = iris$Petal.Width ~ iris$Petal.Length)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
```

```
## -0.56515 -0.12358 -0.01898  0.13288  0.64272
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -0.363076   0.039762  -9.131  4.7e-16 ***
## iris$Petal.Length  0.415755   0.009582  43.387  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2065 on 148 degrees of freedom
## Multiple R-squared:  0.9271, Adjusted R-squared:  0.9266
## F-statistic:  1882 on 1 and 148 DF,  p-value: < 2.2e-16
```

we are interested in the "coefficients" part to quantify uncertainty:

```
sum_coef <- summ$coefficients
sum_coef
```

```
##                     Estimate  Std. Error   t value     Pr(>|t|)
## (Intercept)       -0.3630755 0.039761990 -9.131221 4.699798e-16
## iris$Petal.Length  0.4157554 0.009582436 43.387237 4.675004e-86
```

The second column is what we need: it contains the *standard error* which estimates the uncertainty on the estimated (fitted, predicted) beta values.

```
sum_coef[2,1] #which gives the estimated beta1 value: it stands for sum_coef[#row, #col]
```

```
## [1] 0.4157554
```

```
sum_coef[2,2] #giving the std. error associated to the estimated beta1 value
```

```
## [1] 0.009582436
```

```
sum_coef[2,1] - sum_coef[2, 2]
```

```
## [1] 0.406173
```

```
sum_coef[2,1] + sum_coef[2, 2]
```

```
## [1] 0.4253379
```

The two latter values ($beta_1$ +- $sigma$) are those the estimated $beta_1$ value would range in between if the analysis is repeated.

Then there are the P-values (4th column). P-value is the probability that random fluctuations would produce the estimated beta value that we obtained *if the true beta coefficient was 0*:

```
sum_coef[1, 4]
```

```
## [1] 4.699798e-16
```

```r
sum_coef[2, 4]
```

```
## [1] 4.675004e-86
```

Both $beta_0$ and $beta_1$ have very small P-value, so there's high confidence that the true (actual) coefficient beta is not 0 (i.e. beta=0 would mean that Petal.Width doesn't increase with Petal.Length, meaning no dependence. Such small P-values tell instead that there really is a dependence between Petal.Width and Petal.Length)