Notes7

2022-06-12

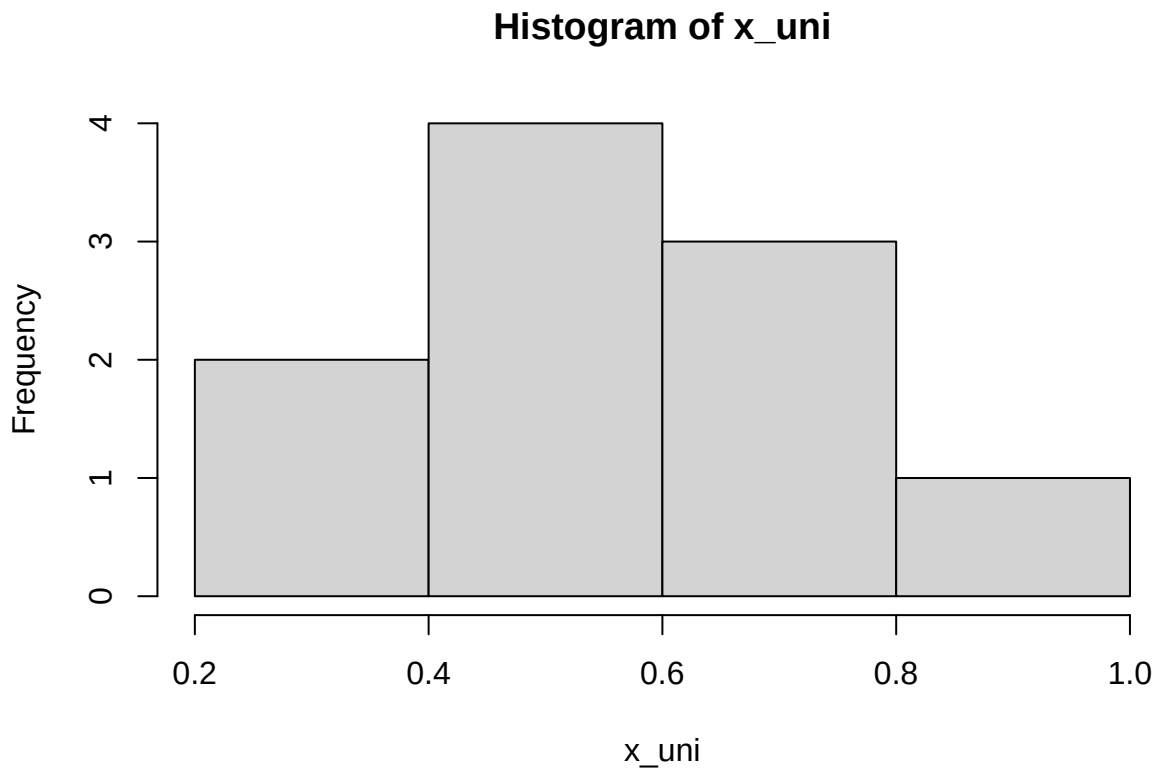
With ANOVA we test whether an added regressor *significantly* increases the explanatory ability of a multivariable model. Now we'll introduce a way to deal with this problem that, differently from anova, can be applied to any kind of regression: **cross-validation** (see at the bottom).

Generating random numbers in R

- `runif()` function creates random numbers from a *uniform distribution* (default: between 0 and 1):

```
x_uni <- runif(n=10)
```

```
hist(x_uni)
```



EVery time you call the `runif()` function you get a different set of random numbers, unless you set a *seed*: this is an integer number that changes every time unless it is set by hand, in this way:

```r
x_uni1 <- runif(n=10)
x_uni1
```

```
##  [1] 0.80035770 0.79133586 0.92519638 0.20154541 0.72858443 0.44368664
##  [7] 0.24415677 0.56179239 0.11617408 0.01055031
```

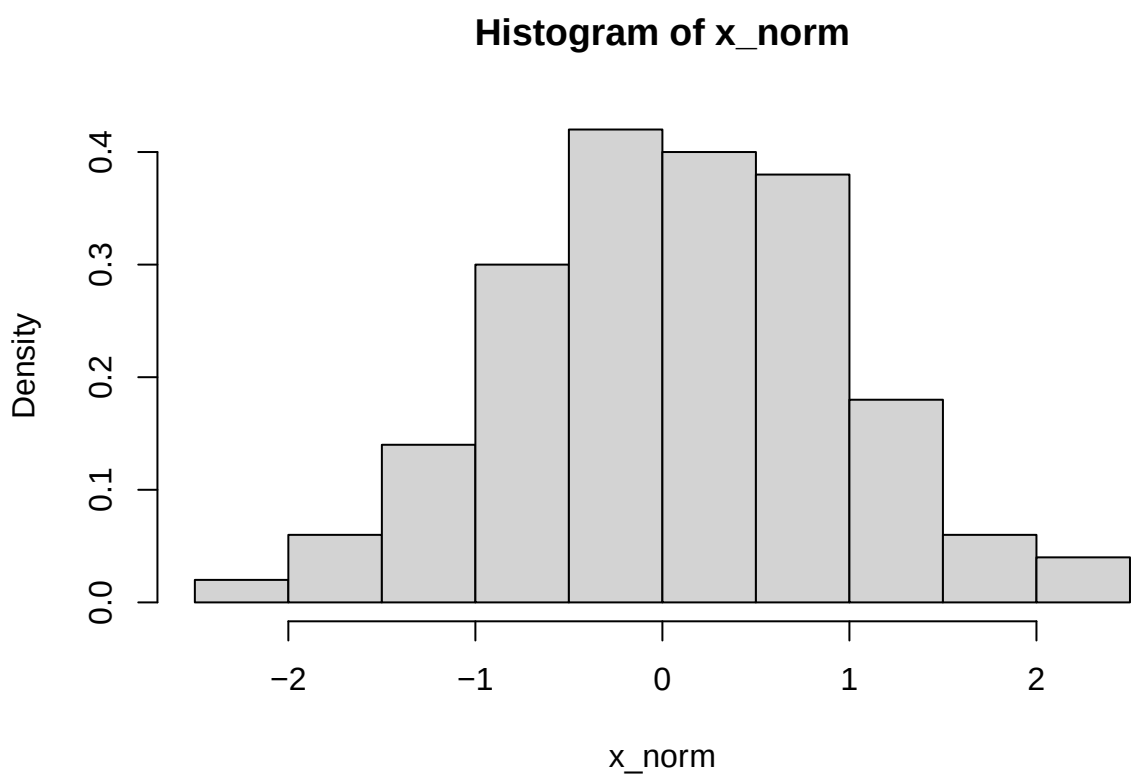
```r
set.seed(1)
x_uni2 <- runif(n=10)
x_uni2
```

```
##  [1] 0.26550866 0.37212390 0.57285336 0.90820779 0.20168193 0.89838968
##  [7] 0.94467527 0.66079779 0.62911404 0.06178627
```

Dovrebbe venire uguale, a me no.

- the `rnorm()` function generates random numbers that follow a *normal distribution* (default: mean 0 and unit Std. Dev.)

```r
x_norm <- rnorm(100)
hist(x_norm, probability = T) #prob set TRUE to see probability of occurrence of each value
```



```
#instead of the number of observations for each value (the Frequency)
```

Linear regression for non-linear functions

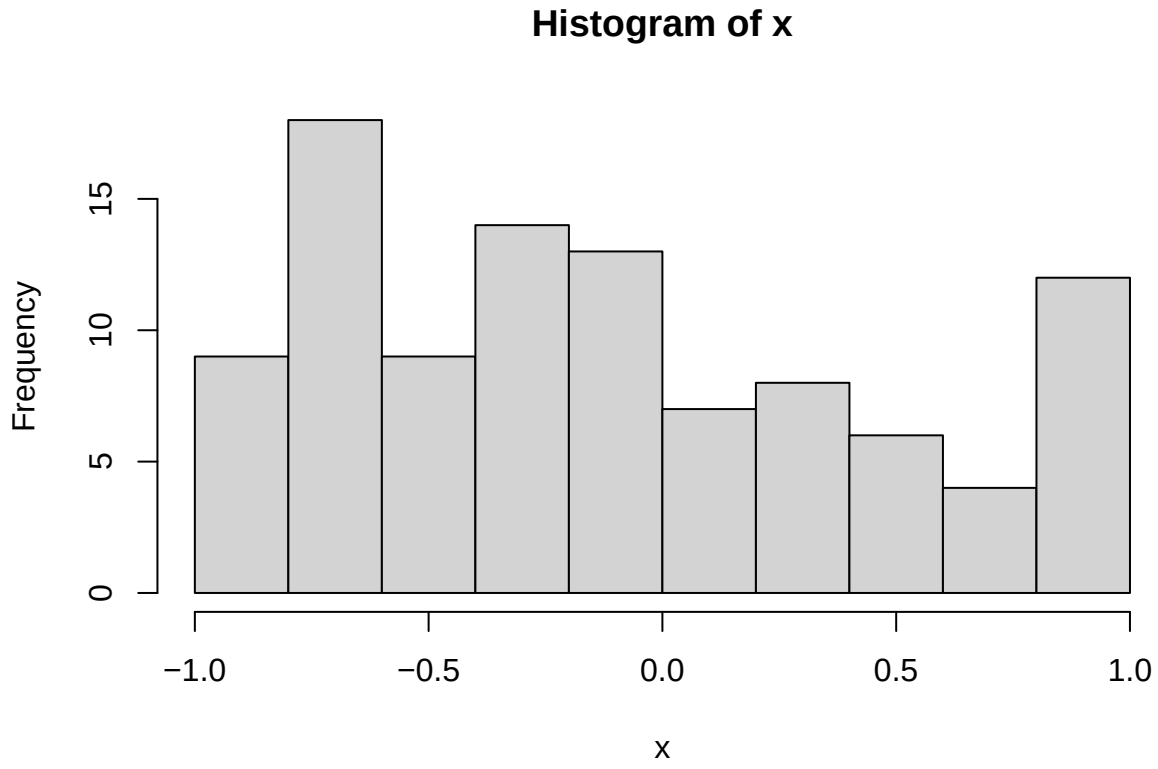
Linear regression is noot only applied when Y depends linearly on X: it is sufficient to have Y depending on a linear combination of *arbitrary functions* of X.

For instance, suppose that Y depends on X as a third-degree polynomial instead of a line:

$Y = X + 0.5X^2 + 5X^3$

Let's initialize the values of X (100 random numbers following a uniform distribution between -1 and +1):

```
x <- runif(100, min = -1, max = 1)
hist(x)
```




```
#sort x by ascending order:
x <- x[order(x)]
```

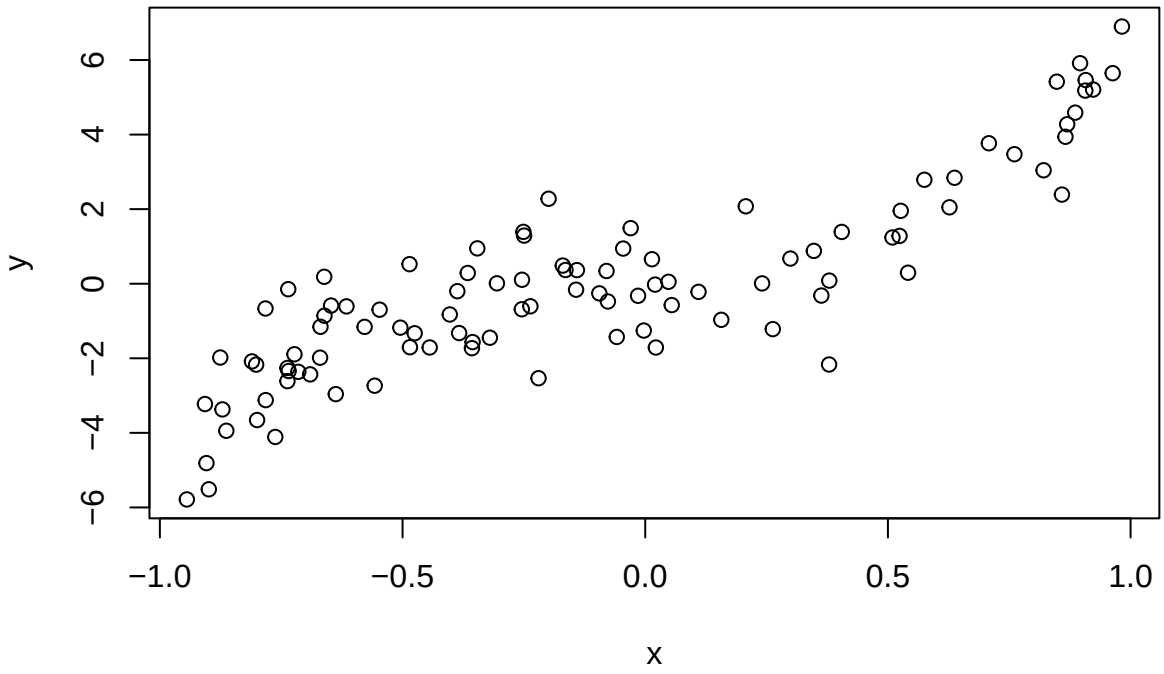
Now we generate the Y:

```
y <- x + 0.5*x^2 + 5*x^3

#then we add gaussian noise to y:
```

```r
y <- y + rnorm(100, mean=0, sd=1)

#and we did so to respect linear regression hypotheses: noise is normally distributed
#and its variance, i.e. sd = 1, does not depend on x
```

We can plot Y and X:

```r
plot(x, y)  #both x and y are numeric variables so we don't use ~
```



Linear regression

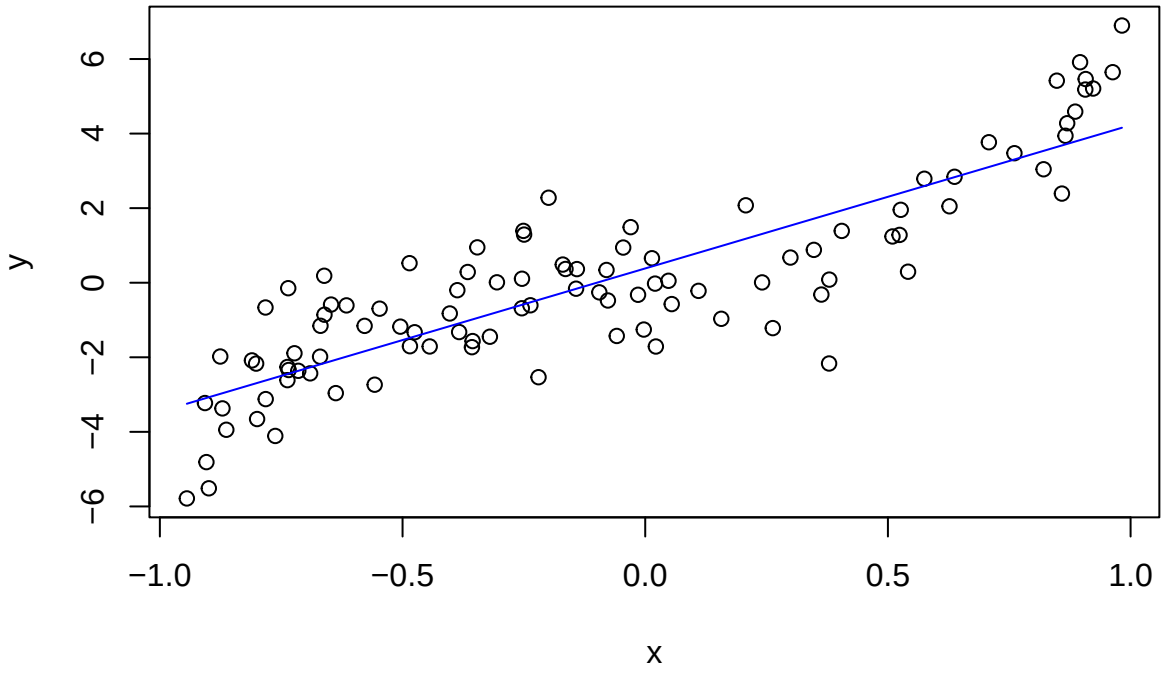
Now we fit data into a linear regression model: from the graph above it's clear there are non-linearities, but linear regression can still be fine.

```r
lm_1 <- lm(y ~ x)
summary(lm_1)$coefficients
```

```
##              Estimate Std. Error   t value     Pr(>|t|)
## (Intercept) 0.3828073  0.1338329  2.860337 5.172322e-03
## x           3.8420508  0.2304865 16.669311 2.305481e-30
```

Let'see the regression curve on the plot:

```
plot(x, y)
lines(x, lm_1$fitted.values, col="blue")
```




```
#in the lines() function, x and lm_1$fitted.values represent X and Y coordinates
#according to the linear regression model
```

This model explains a fraction of the variance of Y equal to:

```
summary(lm_1)$r.squared
```

```
## [1] 0.7392687
```

Polynomial regression

Now X is considered as a quadratic term (X^2), let's define it:

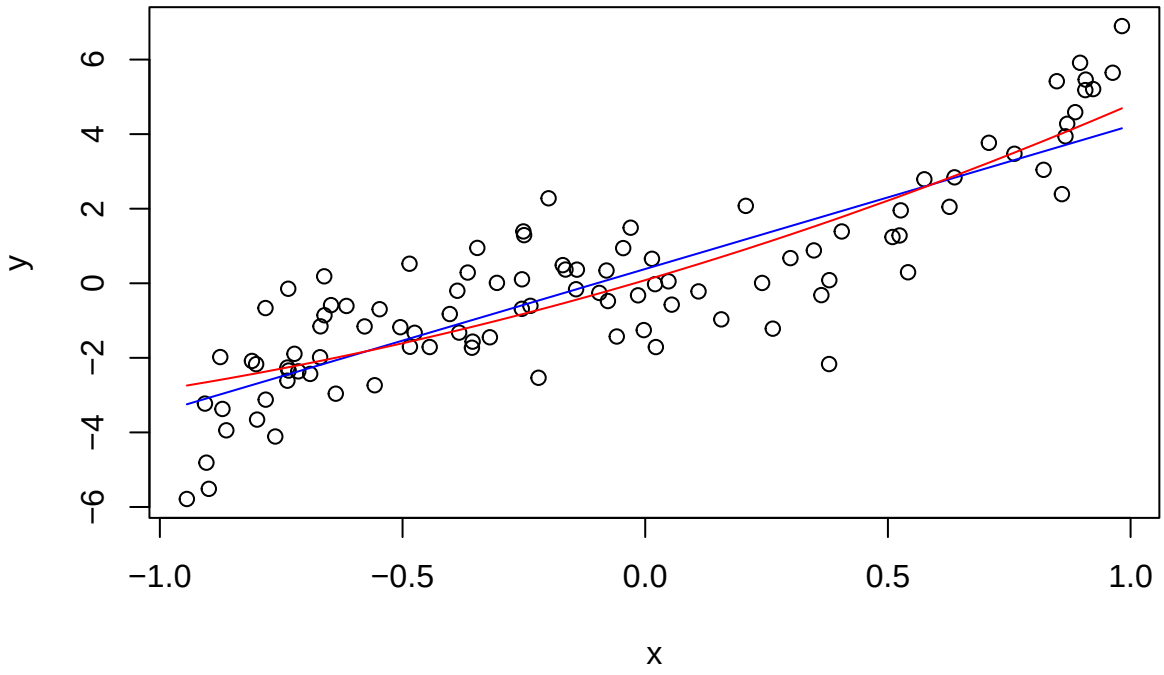
```
x2 <- x^2  #you're initializing x2 starting from the previously defined x!
#(so you're adding the power of 2 to x)
```

Now we use a multivariable linear regression: other than the regressor used before (X), we'll add also this newly defined regressor (X^2):

```
lm_2 <- lm(y ~ x + x2)
summary(lm_2)$coefficients
```

```
##               Estimate Std. Error   t value     Pr(>|t|)
## (Intercept) 0.08356315  0.1991719  0.419553 6.757398e-01
## x           3.82576879  0.2271640 16.841437 1.491532e-30
## x2          0.88214761  0.4401500  2.004198 4.783710e-02
```

```
plot(x, y)

lines(x, lm_1$fitted.values, col = "blue")  #same line as before, to be compared with:
lines(x, lm_2$fitted.values, col = "red")
```



No longer linearity as we added a quadratic term. The aim is to see whether the addition of a quadratic term X^2 does improve the prediction. This is not the case, as:

- the two lines are almost superimposed
- the P-value associated to the quadratic term is non-significant (1.068995e-02)
- the fraction of variance explained by this bivariable model is only slightly higher than that of the univariable model, in fact:

```
summary(lm_2)$r.squared
```

```
## [1] 0.7496364
```

vs 0.7798912 of lm_1.

Remember: R^2 cannot decrease when we add a parameter.

To add even more regressors, there is a faster way instead of doing:

`x2 <- x^2` → `lm_2 <- lm(y ~ x + x2)`, and then again `x3 <- x^3` → `lm_3 <- lm(y ~ x + x2 + x3)`

that is this trick:

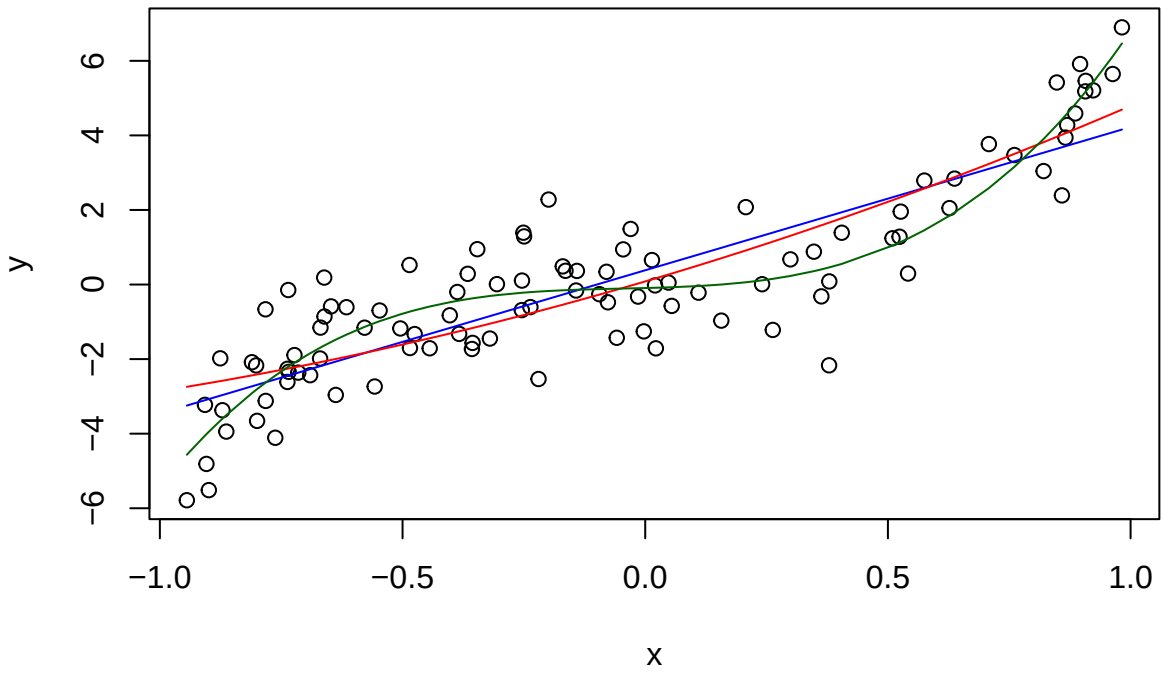
```
lm_3 <- lm(y ~ poly(x, degree = 3, raw = TRUE))
```

`lm(y ~ poly(x, degree = 3, raw = TRUE))` is equivalent to: `x3 <- x^3` → `lm_3 <- lm(y ~ x + x2 + x3)`

```
summary(lm_3)$coefficients
```

```
##                                     Estimate Std. Error    t value     Pr(>|t|)
## (Intercept)                      -0.09359462  0.1611586 -0.5807611 5.627624e-01
## poly(x, degree = 3, raw = TRUE)1  0.32895503  0.5035273  0.6533012 5.151238e-01
## poly(x, degree = 3, raw = TRUE)2  0.79138588  0.3524536  2.2453619 2.703919e-02
## poly(x, degree = 3, raw = TRUE)3  5.77701501  0.7757574  7.4469348 4.126934e-11
```

```
plot(x, y)
lines(x, lm_1$fitted.values, col = "blue")
lines(x, lm_2$fitted.values, col = "red")
lines(x, lm_3$fitted.values, col = "dark green")
```



[Missing part: very artificial situation. The three variables (x, x2, x3) are strongly correlated to each other and therefore do not provide sufficiently independent information to allow disentagling their contribution].

The variance explained by lm_3 is much higher than that explained by linear and quadratic models:

```
summary(lm_1)$r.squared
```

```
## [1] 0.7392687
```

```
summary(lm_2)$r.squared
```

```
## [1] 0.7496364
```

```
summary(lm_3)$r.squared
```

```
## [1] 0.8413085
```

And the increase in R^2 is significant both to the linear model (lm_1) and the quadratic model (lm_2):

```
anova(lm_3, lm_1)
```

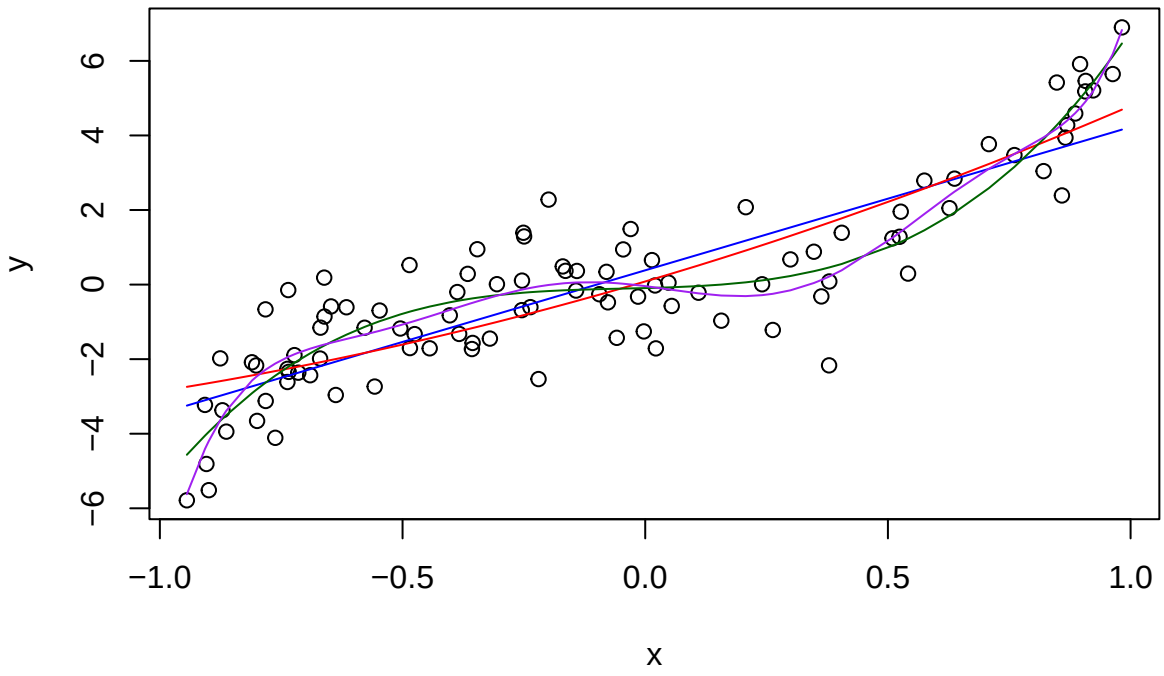
```
## Analysis of Variance Table
##
## Model 1: y ~ poly(x, degree = 3, raw = TRUE)
## Model 2: y ~ x
##   Res.Df    RSS Df Sum of Sq      F   Pr(>F)
## 1     96 102.88
## 2     98 169.03 -2    -66.15 30.864 4.46e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
anova(lm_3, lm_2)
```

```
## Analysis of Variance Table
##
## Model 1: y ~ poly(x, degree = 3, raw = TRUE)
## Model 2: y ~ x + x2
##   Res.Df    RSS Df Sum of Sq      F    Pr(>F)
## 1     96 102.88
## 2     97 162.31 -1   -59.429 55.457 4.127e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

By adding more and more variables (e.g., let's suppose to have a `lm(y ~ poly(x, degree = 8, raw = TRUE))` situation), the higher-terms variables will just *fit the noise*, so that the resulting curve tends to follow the randomic points of the dataset (***overfitting*** → the 8-th degree polynomial model lm_8 describes not only the true dependence of Y on X but also the accidental deviations of the given dataset):

```
lm_8 <- lm(y ~ poly(x, degree = 8, raw = TRUE))

plot(x, y)

lines(x, lm_1$fitted.values, col = "blue")
lines(x, lm_2$fitted.values, col = "red")
lines(x, lm_3$fitted.values, col = "dark green")
lines(x, lm_8$fitted.values, col = "purple")
```



For what above-stated, the results of the ANOVA test tell the 8th-order model is not significantly better than the 3rd-model. This means that at a certain point, while you're adding more and more variables X, R^2 will still increase:

```
summary(lm_8)$r.squared
```

```
## [1] 0.851748
```

... but the prediction won't really improve, and this is demonstrated with the ANOVA test:

```
anova(lm_8, lm_3)
```

```
## Analysis of Variance Table
##
## Model 1: y ~ poly(x, degree = 8, raw = TRUE)
## Model 2: y ~ poly(x, degree = 3, raw = TRUE)
```

```
##   Res.Df     RSS Df Sum of Sq      F Pr(>F)
## 1     91  96.108
## 2     96 102.876 -5   -6.7677 1.2816 0.2787
```

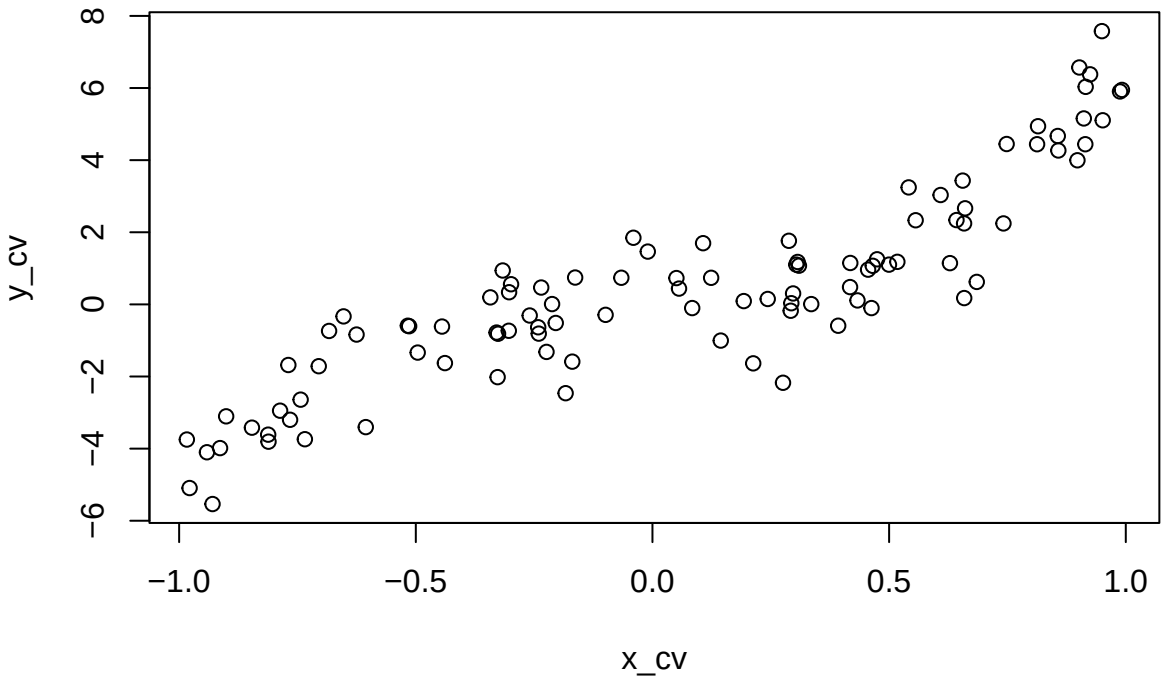
Cross-validation

Cross-validation does the same job as ANOVA but in modern regression techniques (to any kind of regression actually). The idea cross-validation is based on is pretty straightforward:

By adding parameters the prediction will improve (R^2 increases) only on the data on which the model is fit, so if the output is just due to overfitting of that particular dataset, the full model won't fit better than reduced ones if we apply them to a different dataset.

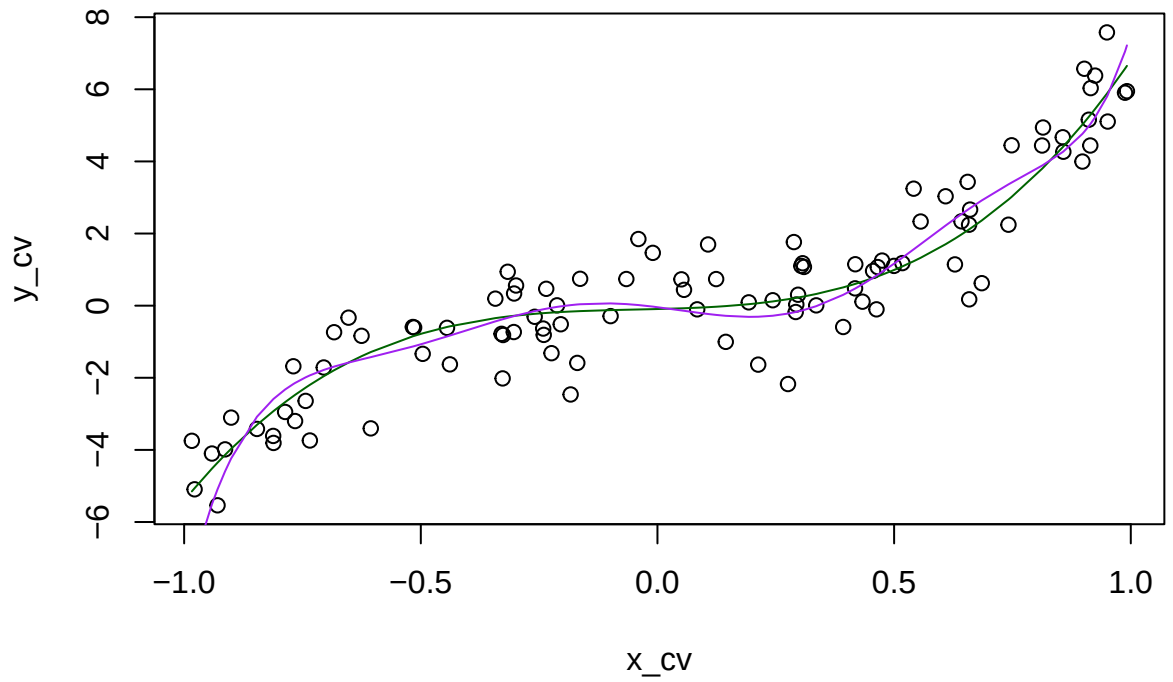
To demonstrate this, we start creating the new dataset to be tested later:

```
x_cv <- runif(100, min = -1, max = 1)
# x_cv will be different from x as we have not reset the random seed

x_cv <- x_cv[order(x_cv)]

y_cv <-  x_cv + 0.5 * x_cv^2 + 5 * x_cv^3
y_cv <- y_cv + rnorm(100)
# also the noise will be different, for the same reason

plot(x_cv, y_cv)
```



The full model and regressed model we are going to consider (to see if the two have the same predictory power on the newly defined dataset, or one is superior to the other) are lm_3 and lm_8.

```
y_cv_pred_3 <- predict(lm_3, newdata = data.frame(x = x_cv))
y_cv_pred_8 <- predict(lm_8, newdata = data.frame(x = x_cv))

plot(x_cv, y_cv)

lines(x_cv, y_cv_pred_3, col = "darkgreen")
lines(x_cv, y_cv_pred_8, col = "purple")
```



Now we want to see the R^2 values, so we define a function computing the R^2 for any set of actual ("y") and predicted ("y_pred") data:

```
r2 <- function(y, y_pred) {
  1 - sum((y - y_pred)^2)/sum((y - mean(y))^2)
}
```

If we use this function on thr original data (lm_3, lm_8), `r2(y, lm_3$fitted.values)` corresponds to `summary(lm_3)$r.squared`:

```
r2(y, lm_3$fitted.values)
```

```
## [1] 0.8413085
```

```
summary(lm_3)$r.squared
```

```
## [1] 0.8413085
```

Now let's try to compute R^2 on the new data:

```
r2_cv_3 <- r2(y_cv, y_cv_pred_3)
r2_cv_8 <- r2(y_cv, y_cv_pred_8)

r2_cv_3
```

```
## [1] 0.8734797
```

```
r2_cv_8
```

```
## [1] 0.8403712
```

→ the 8-th model performs worse than the 3rd-model (IN MY CASE IT IS NOT TRUE, IDK WHY BUT SHOULD BE THE REVERSE)