Tutorial: Multivariate Linear Regression using R

Pramudita Satria Palar, Vani Virdyawan, Ferryanto

2022-04-24

Basic Theory

Now that we can create a single-variable linear regression (see the previous tutorial), you also want to be able to create a linear regression for many independent variables. This is relevant to real-world applications where you want to investigate the relationship between many independent variables and one dependent variable. Taking the case of fluid mechanics as an example, we want to know the simultaneous effect of changes in Mach number and angle of attack on the lift coefficient of an airfoil at transonic flight speed. Multivariable linear regression, or multiple linear regression (MLR), draws a linear relationship between many variables and one dependent variable.

$$Y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_k + \varepsilon, \tag{1}$$

where k is the number of independent variables used and ε is the error term. Note that β_0 is the intercept, and β_i is the slope for the i-th variable (for example, β_2 is the slope for the second variable). To simplify the notation, we will write these MLR coefficients as vectors , i.e. $\boldsymbol{\beta}$, yakni $\boldsymbol{\beta} = \{\beta_0, \beta_1, \ldots, \beta_k\}^T$. We will also define $P = k + 1$ as the number of variables plus 1, which indicates the number of coefficients used, or we can also define it as the number of basis functions.

There is a lot of important information that we can get from making a linear regression model, such as (1) looking for the relationship between independent variables and the dependent variable (e.g., if the variable x_1 is changed, how will it affect Y?), (2) looking for which variable that has the greatest effect on Y, and also other important information.

Why do we want to create an MLR model? The figure below illustrates how a linear regression model can help us in interpreting the relationship between two variables, x_1 and x_2, with y. The relationship in the image below is more precisely $Y = 20 + 8x_1 + 15x_2$. From the visualization and the form of the equation, we can see how the dependent variable relates with the independent variables. For example, increasing x_1 by one unit will increase y by 8 units of y. In this tutorial, we will try to build a linear regression model using R.

The function $Y = 20 + 8x_1 + 15x_2$ will be evaluated at $[-1, 1]^2$ (two variables with a lower limit of -1 and an upper limit of 1). Defined in R as follows:

```
test_function <-function(x1,x2)
{
  result = 20+8*x1+15*x2
  return(result)
}
```

as visualized in the following figure:

To be able to create an MLR model, the first step is to collect data points first,

$$(x_{i1}, x_{i2}, \ldots, x_{ik}, y_i), \quad i = 1, 2, \ldots, n$$

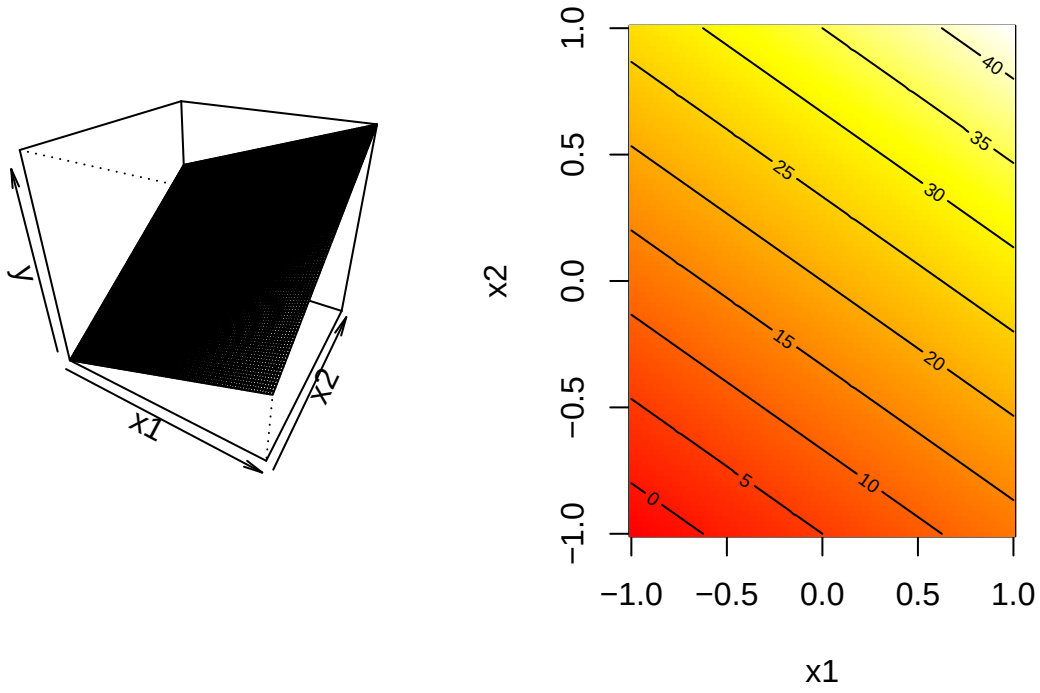

Figure 1: Visualization of the above data

where $n > k$. Each observation is described by the linear model as follows:

$$\begin{aligned} sy_i &= \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_{ik} + \varepsilon_i \\ &= \beta_0 + \sum_{i=1}^{k} \beta_j x_{ij} + \varepsilon_i \quad i = 1, 2, \ldots, n \end{aligned}$$

We can obtain $(x_{i1}, x_{i2}, \ldots, x_{ik}, y_i), \quad i = 1, 2, \ldots, n$ by various means, for example, by physical experiments (tensile test as an example), computer experiments (using computational fluid dynamics), or the results of field observations.

Calculation of coefficients using matrix

MLR problem definition will be easier if we use matrix notation. So, we will first define some matrices that we will use. We first start with a design matrix F of size $n\ddot{O}P$:

$$\mathbf{F} = \begin{bmatrix} 1 & x_{11} & x_{12} & \ldots & x_{1k} \\ 1 & x_{21} & x_{22} & \ldots & x_{2k} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_{n1} & x_{n2} & \ldots & x_{nk} \end{bmatrix}$$

in linear algebra terms, $\mathbf{F}$ is the result from evaluating the basis functions $1, x_1, x_2, \ldots, x_k$ at every point in our data. Next, we collect our observations y in an $n \times 1$ matrix named $\mathbf{y}$:

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \ldots \\ y_n \end{bmatrix}$$

Followed by an $n \times 1$ matrix named $\boldsymbol{\beta}$ (this is the same $\boldsymbol{\beta}$ as which we defined earlier):

$$\boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \ldots \\ \beta_k \end{bmatrix}$$

Similar to the case of simple linear regression, the calculation of the coefficient on the MLR is done by minimizing the residual between the observation and the linear plane (in other words, the linear plane should be as close as possible to the observation).

We can define the relationship between $\boldsymbol{y}$, $\mathbf{F}$, $\boldsymbol{\beta}$, and $\boldsymbol{\epsilon}$ as

$$\boldsymbol{y} = \mathbf{F}\boldsymbol{\beta} + \boldsymbol{\epsilon}, \tag{2}$$

so

$$\boldsymbol{\epsilon} = \boldsymbol{y} - \mathbf{F}\boldsymbol{\beta}. \tag{3}$$

Next, we will redefine the *Residual Sum of Squares (RSS)* which is the function of $\boldsymbol{\beta}$ and it must be minimized. Remember that we want to minimize $\sum_{i=1}^{n} \epsilon_i^2$. By giving the notation $J(\boldsymbol{\beta})$ to define RSS, we can define the objective function as follows

$$J(\boldsymbol{\beta}) = \sum_{i=1}^{n} \epsilon_i^2 = \boldsymbol{\epsilon}^T \boldsymbol{\epsilon} (\boldsymbol{y} - \mathbf{F}\boldsymbol{\beta})^T (\boldsymbol{y} - \mathbf{F}\boldsymbol{\beta}). \tag{4}$$

This objective function must be minimized by finding $\boldsymbol{\beta}$ which minimizes $J(\boldsymbol{\beta})$, or in other words

$$\left. \frac{\partial J(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} \right|_{\hat{\boldsymbol{\beta}}} = 0$$

This problem is an optimization problem with P variables and we can formalize it as

$$\hat{\boldsymbol{\beta}} = \arg\max_{\boldsymbol{\beta}} J(\boldsymbol{\beta}) = \arg\max_{\boldsymbol{\beta}} ||\boldsymbol{y} - \mathbf{F}\boldsymbol{\beta}||^2$$

which requires us to solve the following system of linear equations

$$\mathbf{F}^T\mathbf{F}\boldsymbol{\beta} = \mathbf{F}^T\boldsymbol{y}. \tag{5}$$

We finally find that the coefficients of MLR can be calculated using

$$\boxed{\hat{\boldsymbol{\beta}} = (\mathbf{F}^T\mathbf{F})^{-1}\mathbf{F}^T\boldsymbol{y}} \tag{6}$$

The above technique has its own name: **Ordinary Least Squares (OLS)**. We can see that the calculation of $\boldsymbol{\beta}$ is carried out by solving a system of linear equations. It should be noted that OLS is used in linear regression models and other models that can be defined linearly, such as polynomial regression.

In addition to coefficients, of course we also want to know other important information such as R^2, confidence interval, and prediction interval. Details of the calculation are not given in this short tutorial, but we will do the calculation using R.

Creating Multiple Linear Regression Model

Let's create an MLR model using R. We will use the data already available in the package `datarium` which you can add to your R by typing `install.packages (datarium)` in the R console (if you are having problems, go to Tools -> Install packages). We will use `marketing` data, which is already in the form of `data. frame`, because it is easy to understand. The `marketing` data has three variables: the amount of money in thousands of dollars used to advertise a product from three advertising mediums. The three variables are `youtube`, `facebook`, and `sales`. The dependent variable in this data is `sales`, which indicates the number of sales as a function of `youtube`, `facebook`, and `sales`.

First, we import the marketing data into the R environment we are using and then use the `head()` function to view the first 6 rows of marketing data:

```
data("marketing", package = "datarium")
head(marketing)
```

```
##   youtube facebook newspaper sales
## 1  276.12    45.36     83.04 26.52
## 2   53.40    47.16     54.12 12.48
## 3   20.64    55.08     83.16 11.16
## 4  181.80    49.56     70.20 22.20
## 5  216.96    12.96     70.08 15.48
## 6   10.44    58.68     90.00  8.64
```

The following plot shows the individual plots between the three independent variables and the dependent variable. It is seen that there is a linear trend that can be drawn from these three plots. However, we need to be very careful when looking at individual plots like this because sales are a function of changing three variables simultaneously. For example, try executing the following code:

```
par(mfrow=c(1,3))
plot(marketing$youtube, marketing$sales, pch=19)
plot(marketing$facebook, marketing$sales, pch=19)
plot(marketing$newspaper, marketing$sales, pch=19)
```

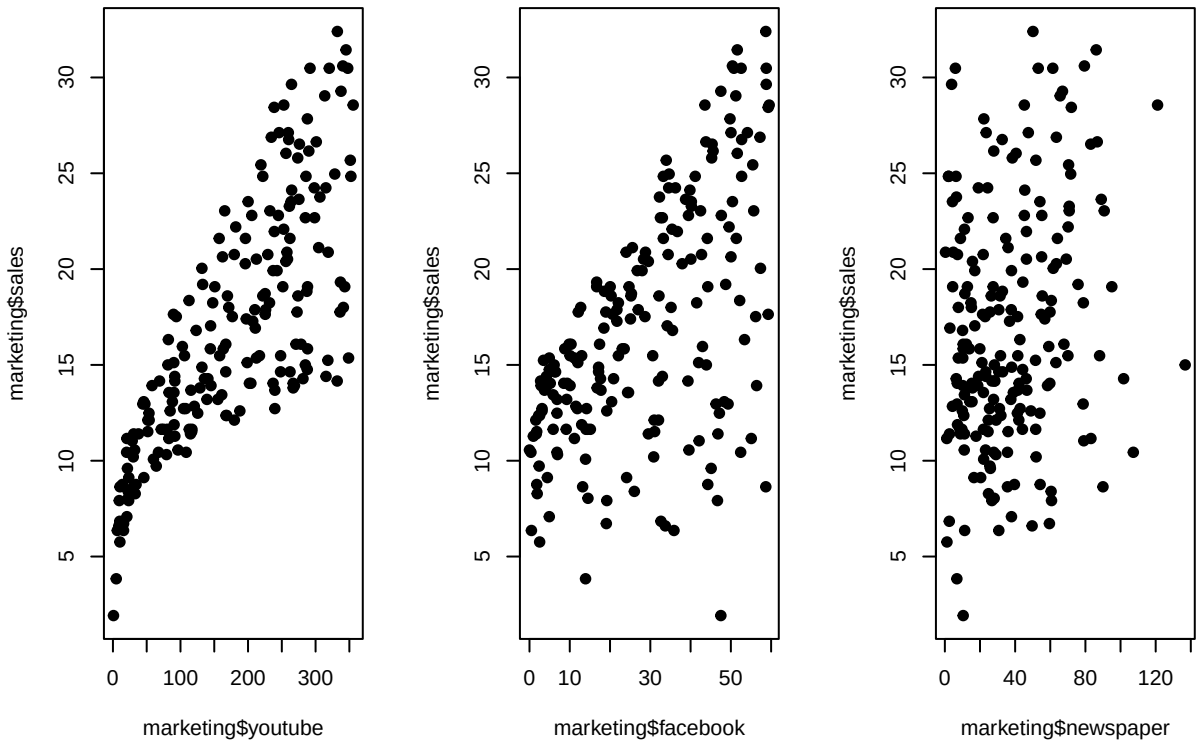

Figure 2: Visualization of all individual independent variables in the marketing data

Our goal is to find an MLR model in the following form:

$$\hat{\text{sales}} = \beta_0 + \beta_1 \times \text{youtube} + \beta_2 \times \text{facebook} + \beta_3 \times \text{newspaper}$$

Here we denote `youtube`, `facebook`, and `newspaper` as the first, second, and third variables.

Let's start with the calculations manually:

```
n <- nrow(marketing) # Jumlah data
dm <- data.matrix(marketing) # Transformasi data frame menjadi matrix
F <- cbind(matrix(1,n,1),dm[,1:3])  # Membuat matriks F
ym <- dm[,4] # Membuat vektor y
coeff <- solve(t(F)%*%F) %*% t(F) %*% ym  # Menghitung koefisien
print(coeff)
```

```
##                   [,1]
##            3.526667243
## youtube    0.045764645
## facebook   0.188530017
## newspaper -0.001037493
```

The calculated coefficient says that the MLR model obtained is

$$\hat{\text{sales}} = 3.525 + 0.045 \times \text{youtube} + 0.188 \times \text{facebook} - 0.001 \times \text{newspaper}$$

The MLR model above says that advertising through Facebook has the most influence on sales, while that through newspapers has almost no influence at all. But, of course, we need to do a deeper analysis to find out other important information, such as the standard error of the coefficient. We will use the function `lm()` for this purpose.

Creating an MLR model with `lm()`

The modeling of the MLR with R follows the same way as when we performed a simple linear regression, namely with the `lm()` function. The main difference is when we define the linear model that will be created, because MLR involves more than one variable. Therefore, we will name the model we created with `lm()` as `MLRmod`.

For the `marketing` data, the simplest syntax for creating an MLR model is as follows:

```
MLRmod = lm(sales~youtube+facebook+newspaper, marketing)
```

We can also write the code as follows to make it easier to read the code.

```
MLRmod = lm(data = marketing, formula = sales~youtube+facebook+newspaper)
```

Let's print the result of this regression model. The coefficient results you see are exactly the same as when we did it manually:

```
print(MLRmod)
```

```
##
## Call:
## lm(formula = sales ~ youtube + facebook + newspaper, data = marketing)
##
## Coefficients:
## (Intercept)      youtube     facebook    newspaper
##    3.526667     0.045765     0.188530    -0.001037
```

We can print all the important information with the `summary()` function:

```r
summary(MLRmod)
```

```
##
## Call:
## lm(formula = sales ~ youtube + facebook + newspaper, data = marketing)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -10.5932  -1.0690   0.2902   1.4272   3.3951
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.526667   0.374290   9.422   <2e-16 ***
## youtube      0.045765   0.001395  32.809   <2e-16 ***
## facebook     0.188530   0.008611  21.893   <2e-16 ***
## newspaper   -0.001037   0.005871  -0.177     0.86
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.023 on 196 degrees of freedom
## Multiple R-squared:  0.8972, Adjusted R-squared:  0.8956
## F-statistic: 570.3 on 3 and 196 DF,  p-value: < 2.2e-16
```

The way to read the summary results above is the same as when we create a one-variable linear regression model. Let's look at the very small p-values of `youtube` and `facebook`. We can see that the null hypothesis $\mathcal{H}_\prime : \beta_1 = 0$ and $\mathcal{H}_\prime : \beta_2 = 0$ can be rejected for these two variables (even with a small level of significance). In other words, `youtube` and `facebook` can be said to have a linear influence on `sales`! Sales through newspapers (`newspapers`) give a p-value of 0.86, where this figure is very far from even the 5% significance level (meaning: fail to reject $\mathcal{H}_\prime : \beta_3 = 0$). We can also safely say that advertising through newspapers does not linearly affect sales. Therefore, analysts can recommend to their clients that advertising is better done through Youtube and Facebook because they have big and clear impact on sales. We also know that $\mathcal{H}_\prime : \beta_0 = 0$ can be rejected (although it is meaningless in our current contex). It can also be seen that the value of R^2 is quite high, and this indicates that the linear regression model made is quite accurate.

Predictions from multiple linear regression model

The method of calculating confidence and prediction interval using R is the same for both single or multivariable linear regression models, i.e., using the `predict()` function. Let's take a look at a simple example first, that is, if we want to know the predictions from our MLR model for the following combination: Youtube = 175, Facebook = 28, and newspaper = 60. The questions that you want to answer are: (1) what are the prediction results, and also (2) the confidence and prediction interval of the prediction.

```r
xpred = data.frame(youtube=175, facebook=28, newspaper=60)
ypred = predict(MLRmod, xpred)
print(ypred)
```

```
##        1
## 16.75207
```

What if you want to make multiple predictions at once? One way is to create a vector or list, e.g. if you want to make predictions at three different points:

```r
xpredm = data.frame(youtube=c(175,100,60), facebook=c(28,40,32), newspaper=c(60,54,12))
ypredm = predict(MLRmod, xpredm)
print(ypredm)
```

```
##        1        2        3
## 16.75207 15.58831 12.29306
```

Calculating the confidence intervals of the coefficients

Calculation of the confidence intervals of the intercept and the slopes can be done using the `confint()` function as in the following example:

```
confint(MLRmod,level=0.99) # Confidence level = 99%
```

```
##                   0.5 %     99.5 %
## (Intercept)  2.55308486 4.50024963
## youtube      0.04213632 0.04939297
## facebook     0.16613095 0.21092909
## newspaper   -0.01630884 0.01423386
```

If we want to show only for one coefficient, for `Youtube` as an example, you need to add the `parm` argument in `confint()`. As an example:

```
confint(MLRmod,level=0.99,parm="youtube") # Anda juga dapat mengetikkan "parm=2"
```

```
##              0.5 %     99.5 %
## youtube 0.04213632 0.04939297
```

Calculating predictions with confidence and prediction interval

Calculating confidence intervals and prediction intervals can be done by adding `interval="confidence"` or `interval="prediction"` inside `predict()`, as in the following example:

```
xpredmci = data.frame(youtube=c(175,100,60), facebook=c(28,40,32), newspaper=c(60,54,12))
ypredmci = predict(MLRmod, xpredmci,interval="confidence")
print(ypredmci)
```

```
##        fit      lwr      upr
## 1 16.75207 16.36170 17.14244
## 2 15.58831 15.15964 16.01698
## 3 12.29306 11.76690 12.81922
```

```
xpredmpi = data.frame(youtube=c(175,100,60), facebook=c(28,40,32), newspaper=c(60,54,12))
ypredmpi = predict(MLRmod, xpredmci,interval="prediction")
print(ypredmpi)
```

```
##        fit       lwr      upr
## 1 16.75207 12.744137 20.76001
## 2 15.58831 11.576463 19.60015
## 3 12.29306  8.269627 16.31649
```

Closing

This is the end of our multivariate linear regression tutorial. We hope this tutorial helps you to understand better what an MLR is and also how to create an MLR model in R. Of course, this tutorial does not cover all aspects of multivariate linear regression (for example, we are not showing plots of residuals as an example) but hopefully it will provide you with an initial overview of MLR.