Regression - Overview

Supervised regression methods

Marcus Birkenkrahe

April 19, 2023

README

Overview and case study of regression methods:

- Regression models are everywhere
- Linear vs. nonlinear regression
- Simple vs. multiple linear regression
- Pearson's correlation coefficient ρ
- Case study: Challenger catastrophe 1986

Source: Lantz, ML in practice, 2019 etc.

Regression analysis

- Regression[1] techniques model size and strength of numeric relationships:
  1. Body weight is a function of calorie intake: weight = f(calories)
  2. Income is a function of job experience: income = f(experience)
  3. Poll numbers help to estimate election chances: election = f(polls)
- The dependent (unknown, predicted) variables are: weight, income, election. The independent (known, given, predictor) variables are calories, experience, polls.

[1] The origin of the term "regression" is Galton's discovery that fathers who were extremely short or tall tended to have sons whose heights were closer to the average height, which he called "regression to the mean" (from Latin 're-gredere', grow back)

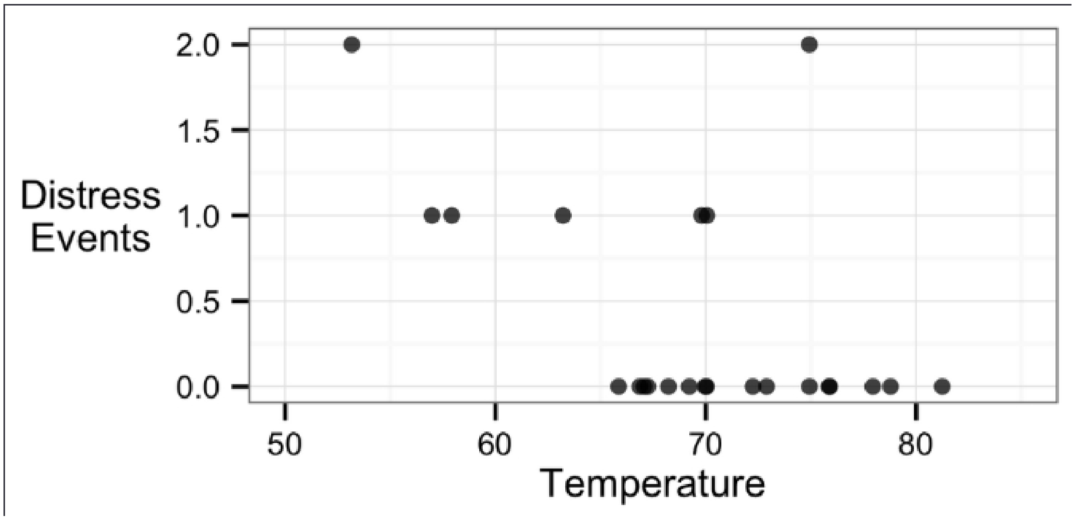


Figure 1: 23 space shuttle O-ring distress events vs. launch temperature

- All ML methods are *regression methods* because they use the data to estimate regression functions, e.g. k-NN uses location data to estimate a distance function, naive Bayes uses event data to estimate a stochastic function, and so on.

Use cases

- Regression analysis is the most common ML method, for example to:
  1. Examine how populations and individuals vary by their measured characteristics in economics, sociology, psychology, ecology.
  2. Quantify causal relationships between an event and its response in clinical drug trials, engineering safety, marketing research.
  3. Identify patterns to forecast future behavior given known criteria like predicting insurance claims, disaster damage, election results.

Understanding regression

- The prediction line follows a linear equation y = a + bx and is determined by intercept and slope.

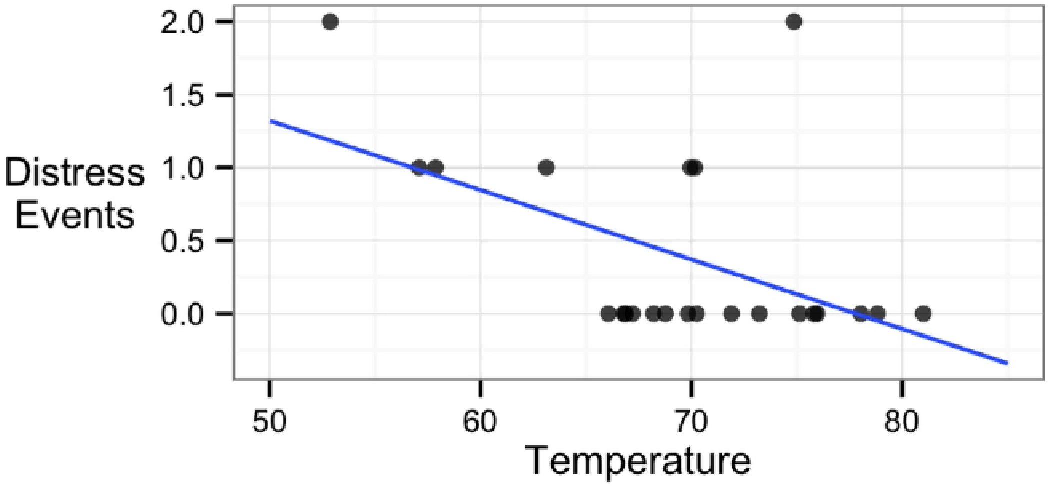


Figure 2: OLS estimate modeling distress events vs. launch temperature

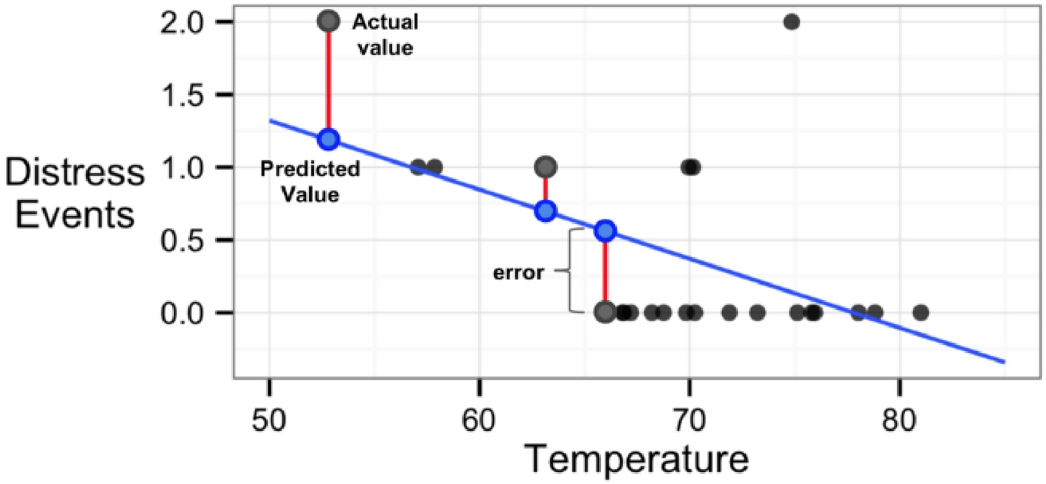


Figure 3: Predictions differ from actuals by a residual (error)

- The shown OLS (Ordinary Least Square) regression is based on minimizing this equation for the error e between actual and predicted y values:

$$\sum (y_i - \hat{y}_i)^2 = \sum e_i^2$$

Figure 4: OLS error

- Using calculus, you can show that the value of b that results in the minimum squared error is:

$$b = \frac{\sum (x_i - \overline{x})(y_i - \overline{y})}{\sum (x_i - \overline{x})^2}$$

Figure 5: Slope solution for minimum squared error

- This value is easy to calculate as a function of covariance and variance of x and y values:

$$\boxed{b = \frac{\mathrm{Cov}(x, y)}{\mathrm{Var}(x)}}$$

Figure 6: Slope as a function of covariance and variance

Example: Challenger O-rings I

- For example, for the Challenger distress vs. launch temperature data[2]:

[2] The data refer to the January 28, 1986 destruction of the US space shuttle Challenger when a rocket booster failed due to the failure of rubber O-rings responsible for sealing

```
launch <- read.csv("../data/challenger.csv", header=TRUE)
str(launch)
```

```
'data.frame': 23 obs. of  4 variables:
 $ distress_ct         : int  0 1 0 0 0 0 0 0 1 1 ...
 $ temperature         : int  66 70 69 68 67 72 73 70 57 63 ...
 $ field_check_pressure: int  50 50 50 50 50 50 100 100 200 200 ...
 $ flight_num          : int  1 2 3 4 5 6 7 8 9 10 ...
```

- We can compute b with R stats functions `cov` and `var` and a = y-bx:

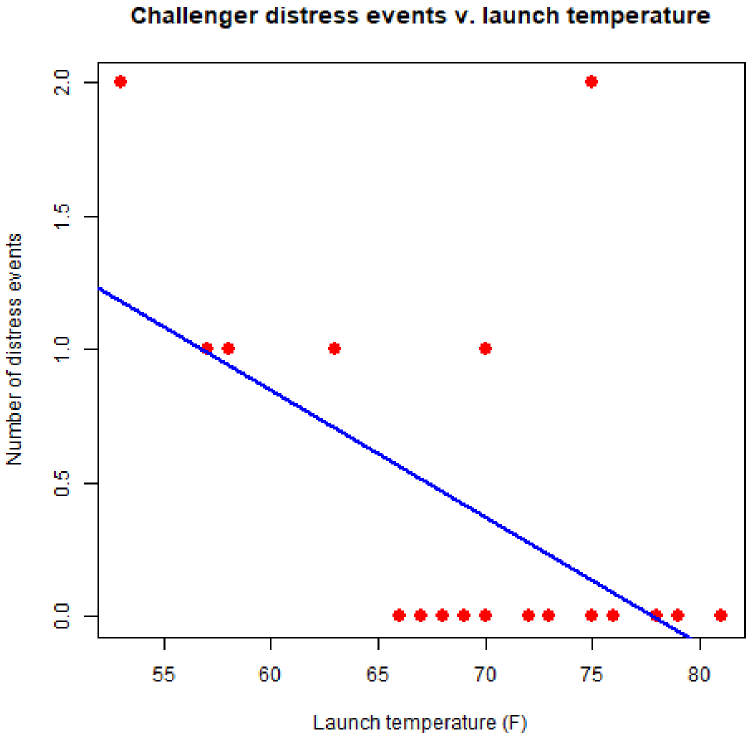
```
b <- cov(x=launch$temperature, y=launch$distress_ct) /
  var(launch$temperature)
b
a <- mean(launch$distress_ct) - b * mean(launch$temperature)
a
```

```
[1] -0.04753968
[1] 3.698413
```

- Plot the manually computed regression model over the data:

```
b <- cov(x=launch$temperature, y=launch$distress_ct) /
  var(launch$temperature)
b
a <- mean(launch$distress_ct) - b * mean(launch$temperature)
a
plot(launch$distress_ct ~ launch$temperature,   ## formula argument
     main="Challenger distress events v. launch temperature",
     xlab="Launch temperature (F)",
     ylab="Number of distress events",
     pch=16,cex=1.5,col="red")
abline(a,b,
       col="blue",lwd=2)
```

the rocket joints, which had never been tested below 40 degrees Fahrenheit (Dalal et al, 1989).



Correlations

- Variable relationships are usually expressed in terms of their *correlation* or their tendency to grow or fall together.
- Without qualification, correlation refers to the *Pearson correlation coefficient* ρ_{x,y} of two vectors x and y where σ denotes the standard deviation (a measure of spread of x and y that is outlier-resilient):

$$\boxed{\rho_{x,y} = \mathrm{Corr}(x,y) = \frac{\mathrm{Cov}(x,y)}{\sigma_x \sigma_y}}$$

Figure 7: Definition of Pearson's correlation coefficient

- Correlation ranges between -1 and +1 with 0 indicating absence of a linear relationship:

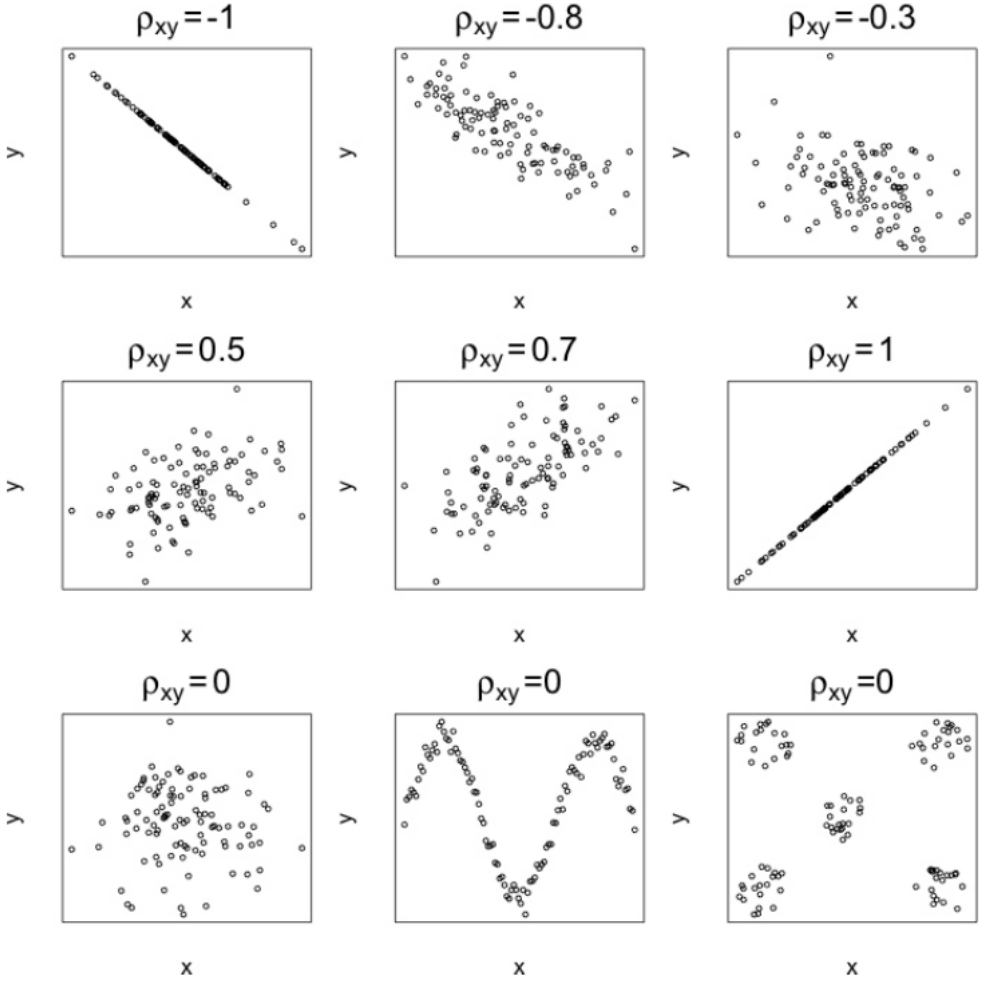


Figure 8: Examples of Pearson coefficients for different datasets

- There are some subtleties here: note how "no linear relationship" in the graphs of the last row reveals nothing about the observable patterns!

Example: Challenger O-rings II

- We compute the correlation between launch temperature and number of O-ring distress events first manually using the formula and then using R's `cor` function:

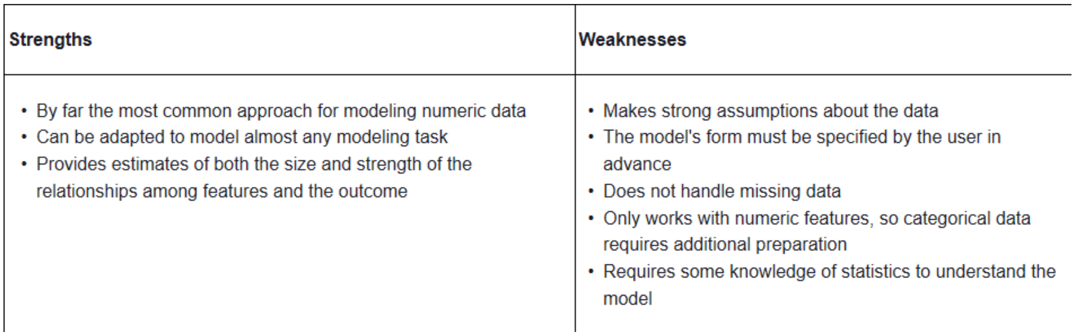
```
r <- cov(launch$temperature, launch$distress_ct) /
  (sd(launch$temperature) * sd(launch$distress_ct))
r
cor(launch$temperature, launch$distress_ct)
```

```
[1] -0.5111264
[1] -0.5111264
```

- The value of `r` suggests that increases in temperature are related to decreases in the number of distressed O-rings.
- The value of `r` suggests a moderately strong negative linear correlation.

Multiple linear regression

- Most real world problems present more than one independent variable, leading to multiple linear regression.



Strengths	Weaknesses
• By far the most common approach for modeling numeric data • Can be adapted to model almost any modeling task • Provides estimates of both the size and strength of the relationships among features and the outcome	• Makes strong assumptions about the data • The model's form must be specified by the user in advance • Does not handle missing data • Only works with numeric features, so categorical data requires additional preparation • Requires some knowledge of statistics to understand the model

Figure 9: Strengths and weaknesses of multiple linear regression analysis

- The modified modeling equation for multiple independent variables with regression coefficients β_{i}
- We can represent the setup of a multiple regression task:

$$y = \beta_0 x_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_i x_i + \varepsilon$$

Figure 10:

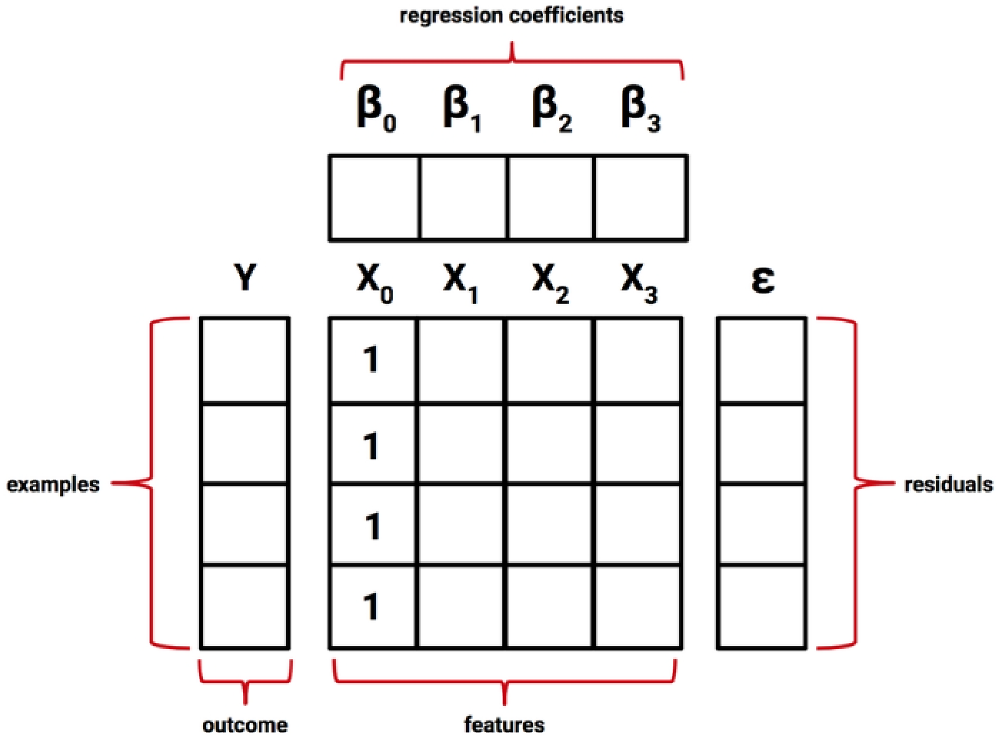


Figure 11: Multiple regression finds the β values that relate the X values to Y while minimizing ϵ

- In matrix notation, the formula above is Y = β X + ϵ, with the matrix X of independent variables, and the best estimate of the vector β is given by:

$$\hat{\beta} = \left(\mathrm{X}^{\mathrm{T}}\mathrm{X}\right)^{-1} \mathrm{X}^{\mathrm{T}}\mathrm{Y}$$

Figure 12: Linear regression coefficients for independent X and dependent Y

- We can use R's built-in matrix operations to create a function `reg` that takes x and y and returns a vector of β coefficient estimates:

```
reg <- function(y,x) {
  x <- as.matrix(x)          # turn x into matrix
  x <- cbind(Intercept = 1, x)  # add intercept
  b <- solve(t(x) %*% x) %*% t(x) %*% y  # compute coefficients
  colnames(b) <- "estimate"  # name coefficient vector
  print(b)
}
```

- Earlier we manually computed a (= β_{0}) = 3.7 and b = -0.048 from averages for the simple linear regression case. With `reg`:

```
reg <- function(y,x) {
  x <- as.matrix(x)          # turn x into matrix
  x <- cbind(Intercept = 1, x)  # add intercept
  b <- solve(t(x) %*% x) %*% t(x) %*% y  # compute coefficients
  colnames(b) <- "estimate"  # name coefficient vector
  print(b)
}
reg( y = launch$distress_ct,   # independent variable
     x = launch[2])          # dependent variable


               estimate
Intercept    3.69841270
temperature -0.04753968

str(launch)
```

```
'data.frame': 23 obs. of  4 variables:
 $ distress_ct         : int  0 1 0 0 0 0 0 0 1 1 ...
 $ temperature         : int  66 70 69 68 67 72 73 70 57 63 ...
 $ field_check_pressure: int  50 50 50 50 50 50 100 100 200 200 ...
 $ flight_num          : int  1 2 3 4 5 6 7 8 9 10 ...
```

- If we add the other independent variables:

```
reg <- function(y,x) {
  x <- as.matrix(x)          # turn x into matrix
  x <- cbind(Intercept = 1, x)  # add intercept
  b <- solve(t(x) %*% x) %*% t(x) %*% y  # compute coefficients
  colnames(b) <- "estimate"  # name coefficient vector
  print(b)
}
reg( y = launch$distress_ct,   # independent variable
     x = launch[2:4])          # dependent variables
```

```
                        estimate
Intercept            3.527093383
temperature         -0.051385940
field_check_pressure  0.001757009
flight_num           0.014292843
```

Summary

- Linear regression, by far the most common modeling approach for numeric data, involves fitting straight lines to data.

- We looked at a historical, simple, Ordinary-Least-Squares (OLS) linear regression case (Challenger Space Shuttle disaster).

- Correlation is usually measured with the Pearson correlation coefficient ρ, the tendency of two variables to grow and fall together. ρ=0 can hide non-linear patterns.

- For multiple linear regression, the β_{i} coefficients can be found using a matrix operation.

Glossary of code

COMMAND	MEANING
`cov(x,y)`	Statistical covariance
`sd(x,y)`	Standard deviation
`var(x,y)`	Variance
`cor(x,y)`	Correlation coefficient

Further study

DataCamp courses (remember you have access until July):

- Introduction to regression in R (DataCamp)
- Supervised learning: regression (DataCamp)

References

- Dalal et al (1989). Risk analysis of the Space Shuttle. In: J. Am. Stat. Ass. 84:945-957.
- Lantz (2019). Machine Learning with R (3e). Packt.