Lesson 12

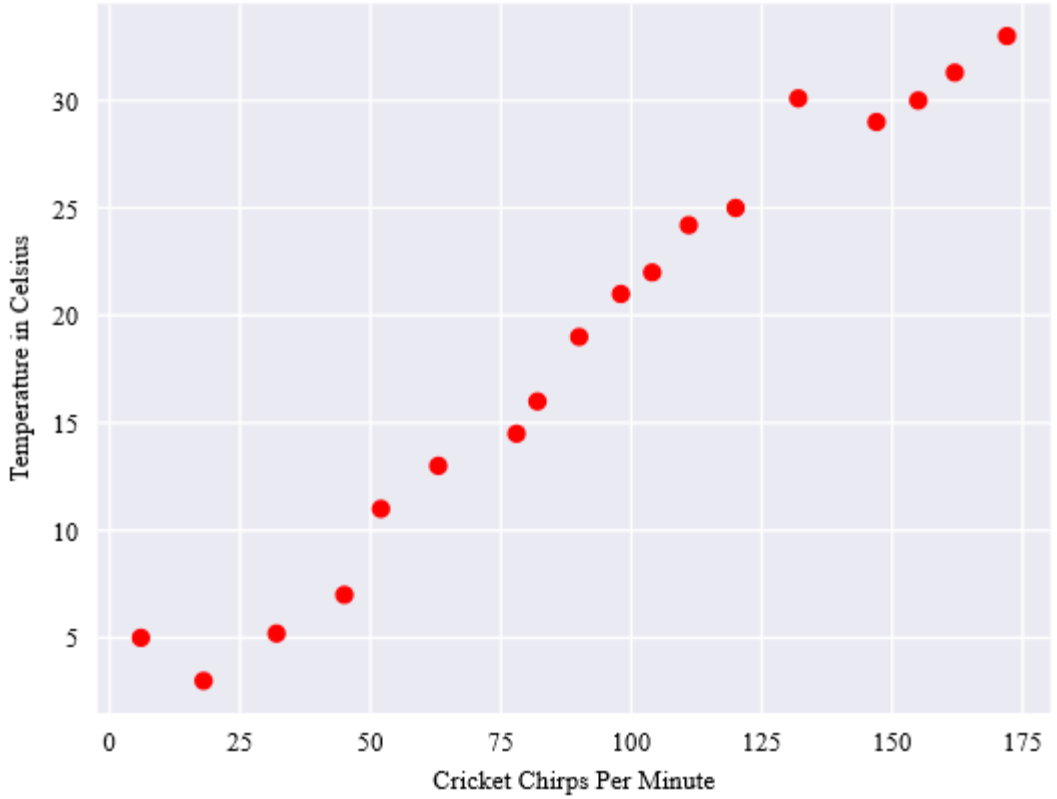
Machine Learning - Regression

Kush Kulshrestha

Introduction

It has long been known that crickets (an insect species) chirp more frequently on hotter days than on cooler days. For decades, professional and amateur scientists have catalogued data on chirps-per-minute and temperature.

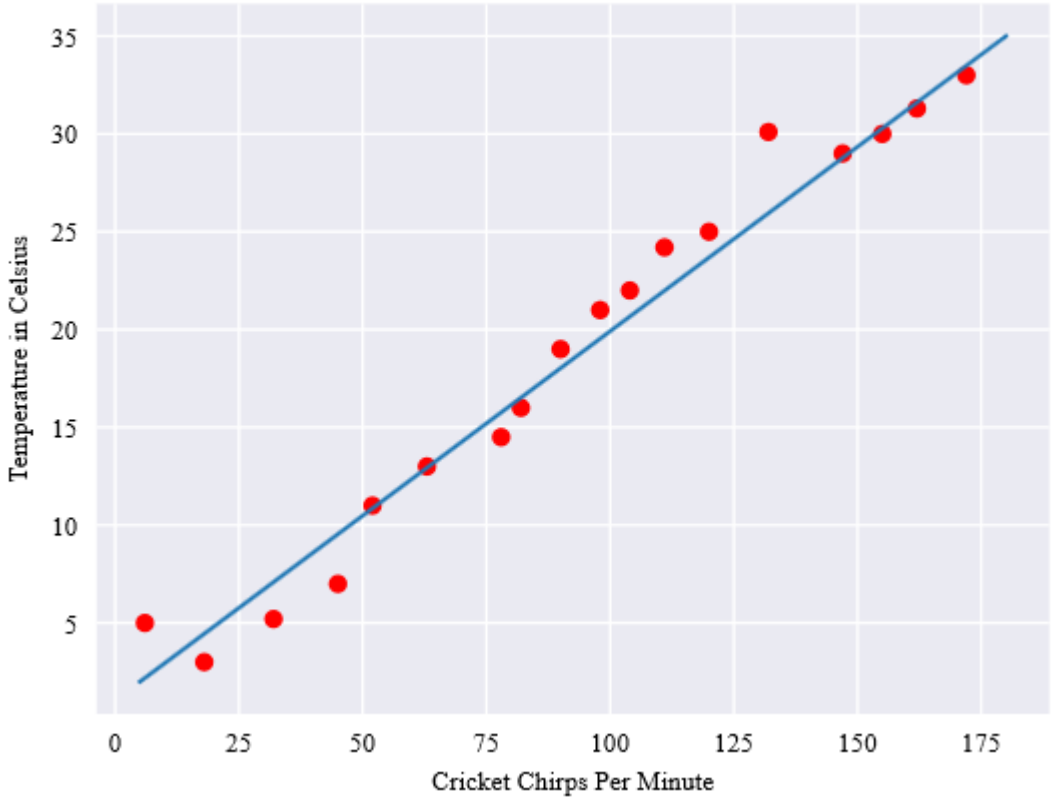
Using this data, you want to explore this relationship.



Introduction

As expected, the plot shows the temperature rising with the number of chirps. Is this relationship between chirps and temperature linear?

Yes, you could draw a single straight line like the following to approximate this relationship: True, the line doesn't pass through every dot, but the line does clearly show the relationship between chirps and temperature.



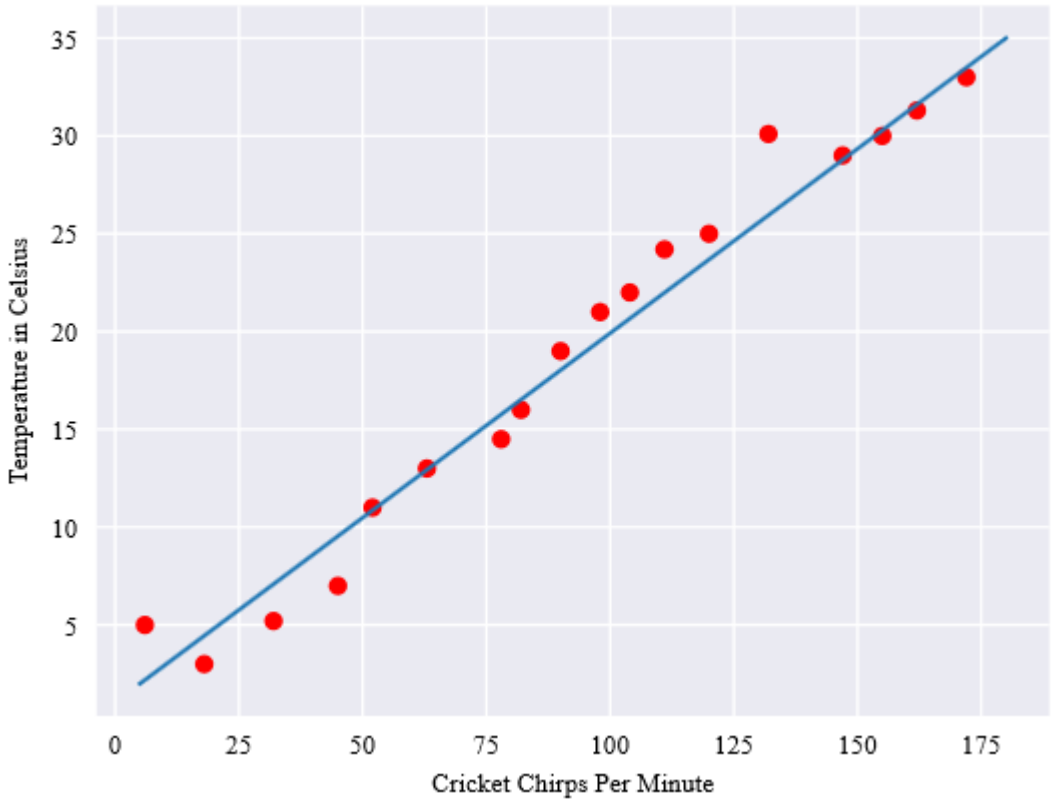
Introduction

Using the equation for a line, you could write down this relationship as follows:

$$y = mx + b$$

where,

- y is the temperature in Celsius—the value we're trying to predict.
- m is the slope of the line.
- x is the number of chirps per minute—the value of our input feature.
- b is the y-intercept



Introduction

$$y = mx + b$$

By convention in machine learning, you'll write the above equation for a model slightly differently:

$$y' = b + w_1 x_1$$

where,

- y’ is the predicted label (a desired output).
- b is the bias (the y-intercept), sometimes referred to as w_0
- w1 is the weight of feature 1. Weight is the same concept as the "slope“ m in the traditional equation of the line.
- X1 is the feature.

To **infer** (predict) the temperature y’ for a new chirps-per-minute value x1, just substitute the x1 value into this model.
Although this model uses only one feature, a more sophisticated model might rely on multiple features, each having a separate weight (w1, w2 , etc.). For example, a model that relies on three features might look as follows:

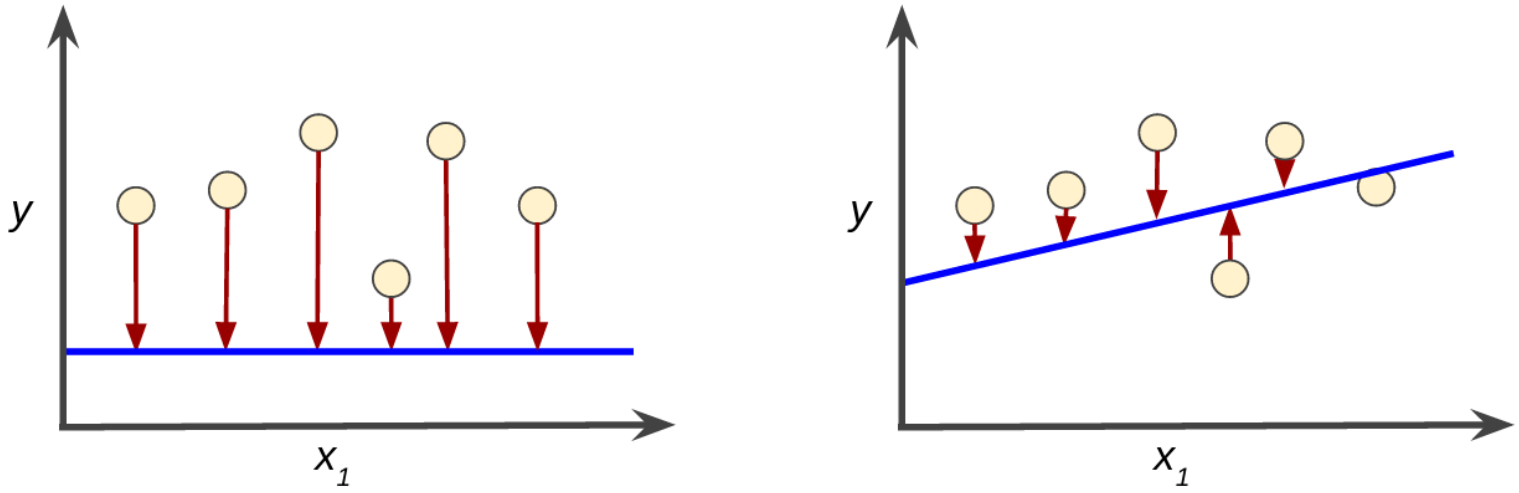
$$y' = b + w_1 x_1 + w_2 x_2 + w_3 x_3$$

What exactly is Training ?

Training a model simply means learning (determining) good values for all the weights and the bias from labelled examples. In supervised learning, a machine learning algorithm builds a model by examining many examples and attempting to find a model that minimizes loss.

Loss is the penalty for a bad prediction. That is, **loss** is a number indicating how bad the model's prediction was on a single example. If the model's prediction is perfect, the loss is zero; otherwise, the loss is greater. The goal of training a model is to find a set of weights and biases that have *low* loss, on average, across all examples.

Example, Figure 3 shows a high loss model on the left and a low loss model on the right.



Linear Regression

Linear regression is very good to answer the following questions:

- Is there a relationship between 2 variables?
- How strong is the relationship?
- Which variable contributes the most?
- How accurately can we estimate the effect of each variable?
- How accurately can we predict the target?
- Is the relationship linear?
- Is there an interaction effect?

Let's assume we only have one variable and one target. Then, linear regression is expressed as:

$$Y = \beta_0 + \beta_1 X$$

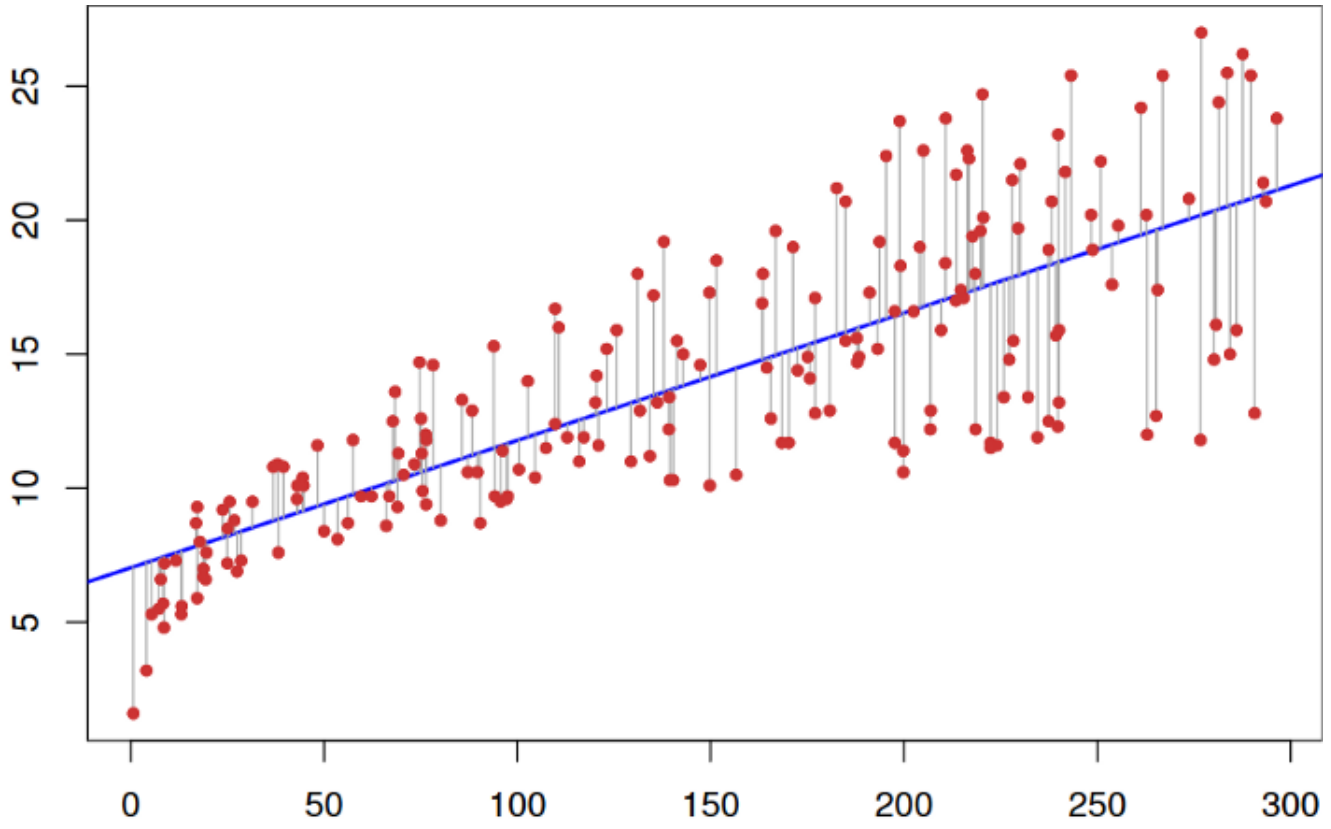
In the equation above, the *betas* are the coefficients. These coefficients are what we need in order to make predictions with our model.

To find the parameters, we need to minimize the **least squares** or the **sum of squared errors**.

Why do we use squared errors?

Linear Regression

In the below above, the red dots are the true data and the blue line is linear model. The grey lines illustrate the errors between the predicted and the true values. The blue line is thus the one that minimizes the sum of the squared length of the grey lines.



Linear Regression

After some math heavy lifting, you can finally estimate the coefficients with the following equations:

$$\beta_1 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

$$\beta_0 = \bar{y} - \beta_1 \bar{x}$$

where *x bar* and *y bar* represent the mean.

Correlation coefficient

β_1 can also be written as:

$$\beta_1 = r\frac{s_y}{s_x}$$

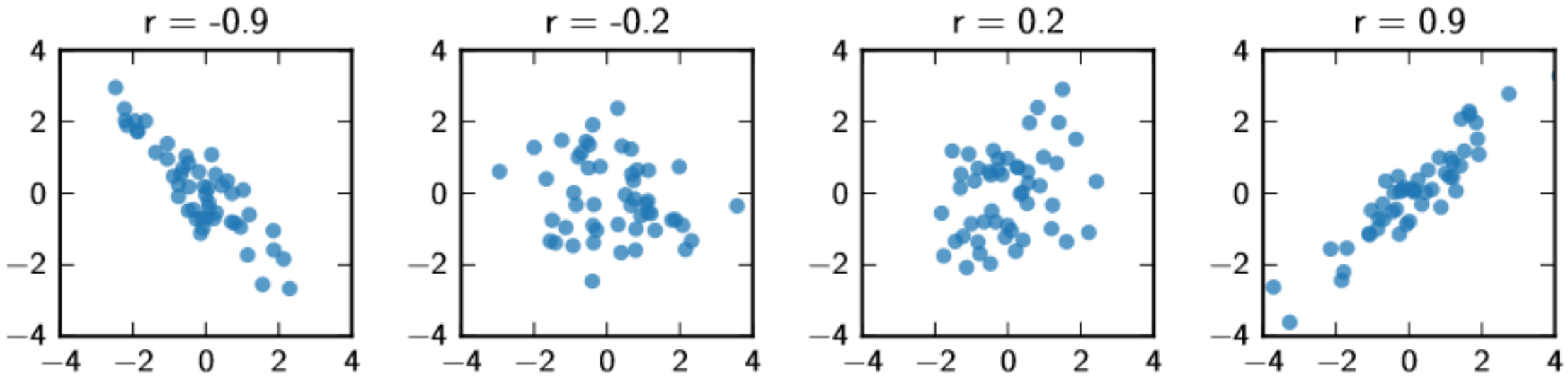
where,
Sy and Sx are standard deviations of x and y values respectively, and r is the **correlation coefficient** defined as,

$$r = \frac{1}{n-1}\sum_{i=1}^{n}\left(\frac{x_i - \bar{x}}{s_x}\right)\left(\frac{y_i - \bar{y}}{s_y}\right).$$

By examining the second equation for the estimated slope β1, we see that since sample standard deviations Sx and Sy are positive quantities, the correlation coefficient (r), which is always between−1 and 1, measures how much x is related to y and whether the trend is positive or negative. Figure 3.2 illustrates different correlation strengths.

Correlation coefficient

Figure below illustrates different correlation strengths.



An illustration of correlation strength. Each plot shows data with a particular correlation coefficient r. Values farther than 0 (outside) indicate a stronger relationship than values closer to 0 (inside). Negative values (left) indicate an inverse relationship, while positive values (right)indicate a direct relationship.

Coefficient of Determination

The square of the correlation coefficient, r^2 will always be positive and is called the **coefficient of determination**.

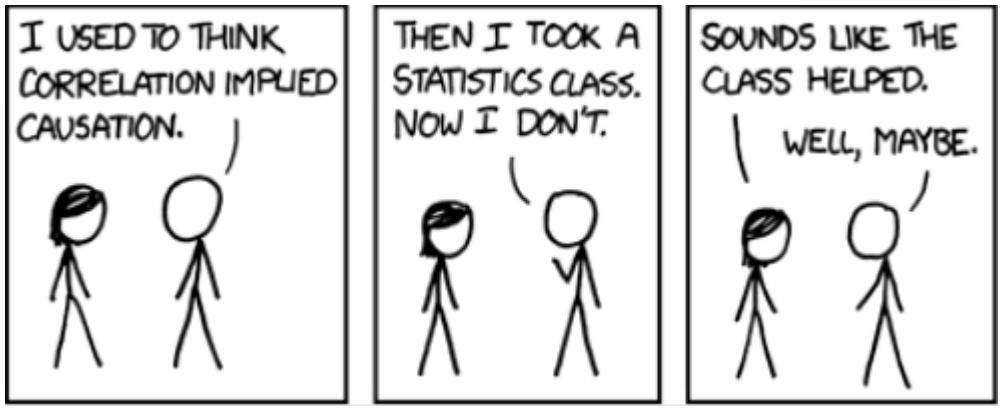
This also is equal to the proportion of the total variability that's explained by a linear model.

As an extremely crucial remark, correlation does not imply causation!

Correlation and Causation

Just because there's a strong correlation between two variables, there isn't necessarily a causal relationship between them.

For example, drowning deaths and ice-cream sales are strongly correlated, but that's because both are affected by the season (summer vs. winter). In general, there are several possible cases, as illustrated below:



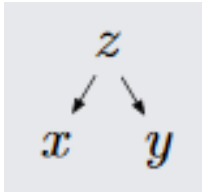
1) **Causal Link**: Even if there is a causal link between x and y, correlation alone cannot tell us whether y causes x or x causes y.

$$x \longrightarrow y$$

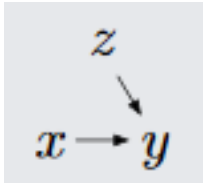
$$x \longleftarrow y$$

Correlation and Causation

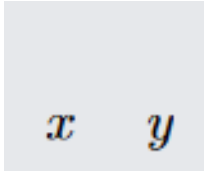
2) **Hidden Cause**: A hidden variable z causes both x and y, creating the correlation.



3) **Confounding Factor**: A hidden variable z and x both affect y, so the results also depend on the value of z.



4) **Coincidence**: The correlation just happened by chance (e.g. the strong correlation between sun cycles and number of Republicans in Congress)



Multiple Linear Regression

This is the case when instead of being single x value, we have a vector of x values (x1, x2, ……, xn) for every data point i.

So, we have n data points (just like before), each with p different predictor variables or features. We'll then try to predict y for each data point as a linear function of the different x variables:

$$y = \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p.$$

Even though it's still linear, this representation is very versatile; here are just a few of thethings we can represent with it:

- **Multiple dependent variables**: for example, suppose we're trying to predict medical outcome as a function of several variables such as age, genetic susceptibility, and clinical diagnosis. Then we might say that for each patient, x1= age, x2= genetics, x3=diagnosis, and y= outcome.
- **Nonlinearities**: Suppose we want to predict a quadratic function y=ax^2 + bx + c, then for each data point we might say x1= 1 , x2=x , and x3=x^2. This can easily be extended to any nonlinear function we want.

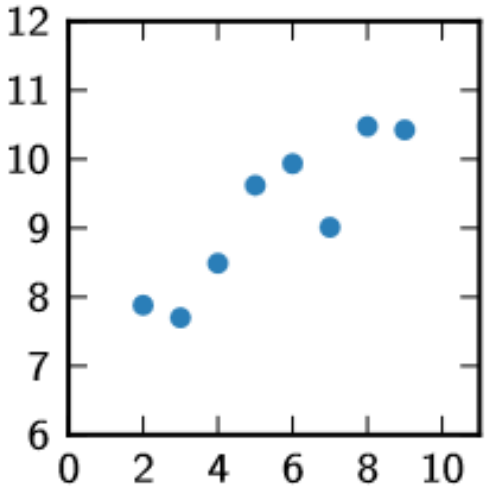
One may ask: why not just use multiple linear regression and fit an extremely high-degree polynomial to our data? While the model then would be much richer, one runs the risk of overfitting

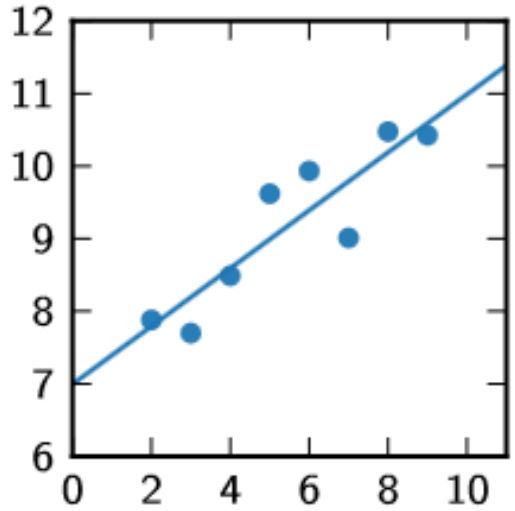
Multiple Linear Regression

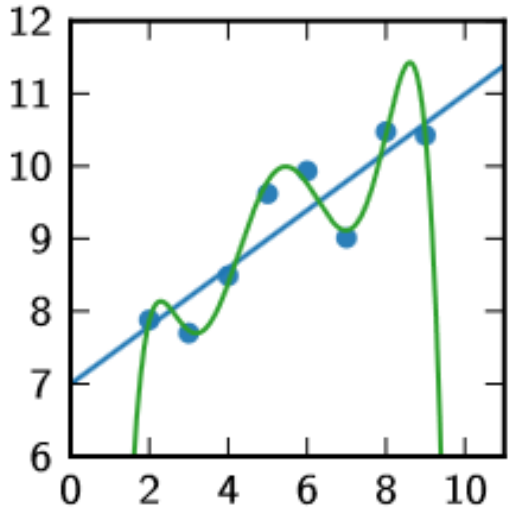
Using too many features or too complex of a model can often lead to overfitting.

Suppose we want to fit a model to the points in Figure 1. If we fit a linear model, it might look like Figure 2. But, the fit isn't perfect. What if we use our newly acquired multiple regression powers to fit a 6th order polynomial to these points? The result is shown in Figure 3

While our errors are definitely smaller than they were with the linear model, the new model is far too complex, and will likely go wrong for values too far outside the range.







Application – Linear Regression in Scikit Learn

Please check out the jupyter notebook

Application – Linear Regression in Statsmodels

Please check out the jupyter notebook