Linear regression

By Dominic Waithe

Easy to do, hard to master

28th February 2018

Today’s talk and presentation.

- Talk
    - What is linear regression.
    - How do we do it. Little bit of maths here.
    - Simple Vs multiple linear regression.
    - Error and other things.
- Practical content.
    - Download ‘jupyter notebook’.
    - Work through the notebooks.

Sources:

Introduction

- Simple linear regression is a statistical method that allows us to summarise and study relationships between two continuous (quantitative variables)

    - One variable often denoted (x), is regarded as the predictor, explanatory or independent variable.

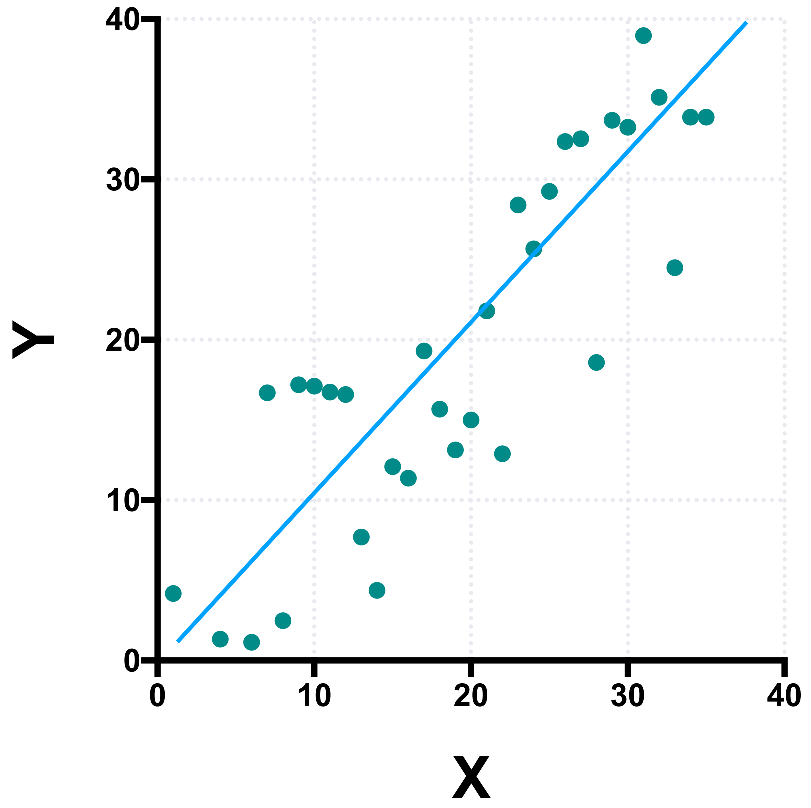
    - The other variable, called (y) is regarded as the response, outcome or dependent variable

- Why do we fit? We fit data so we can make inferences about that data. We can make predictions for data for which we don't know the output variable.

Sources: https://onlinecourses.science.psu.edu/stat501/node/251

Introduction

- Common nomenclature



x_i denotes the independent variable for each datapoint **i**

y_i denotes the dependent variable for each datapoint **i**

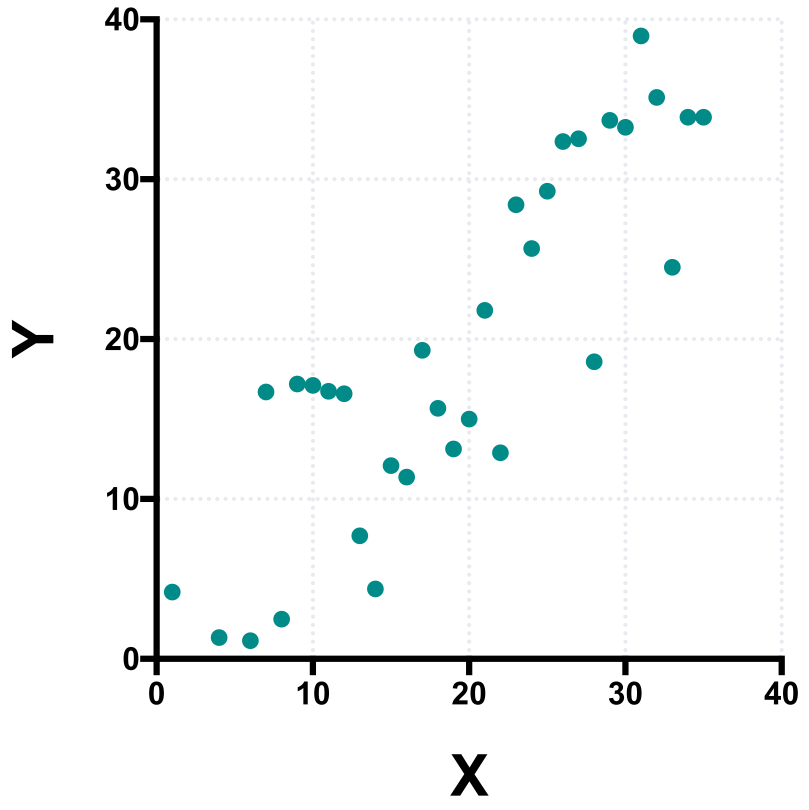
$\hat{y}_i$ is the predicted dependent variable for each datapoint **i**

$\hat{y}_i = b_o + b_1 x_i$ The equation of a straight-line.

Sources:

Linear Regression

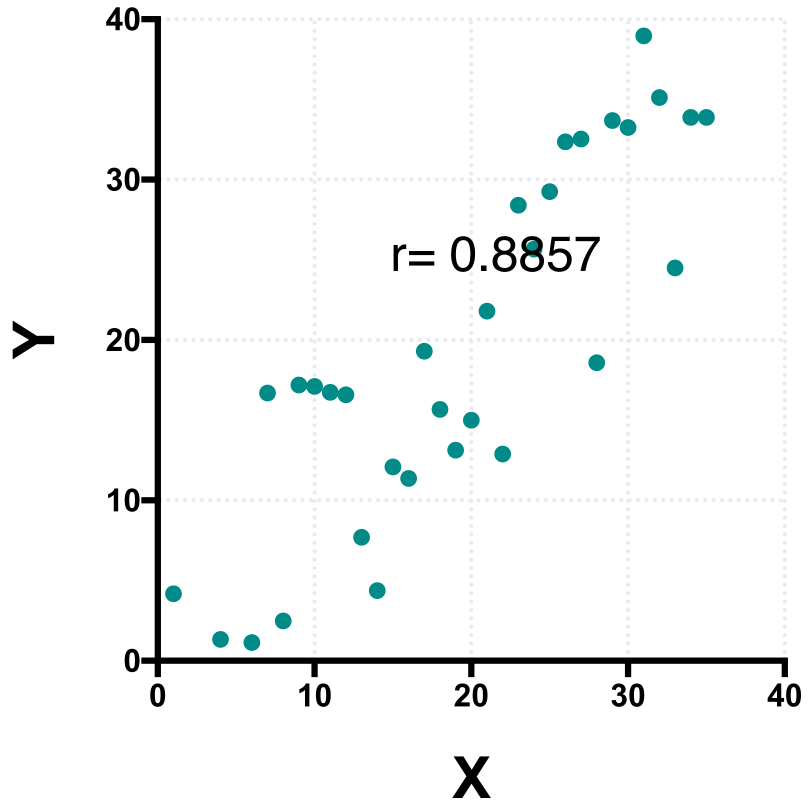
- An example of “simple linear regression”.
- It is called simple when there is only one independent variable (x).
- When their is more than one independent variable then it is called ‘multiple linear regression’.



Sources:

Should I fit a line to my data?

- We should first establish whether the data should be fit or not. A good metric through which to decide to this is through correlation.
- Correlation measures the linear relationship between x and y.



$$r = \frac{1}{n-1}\sum\left(\frac{x-\bar{x}}{s_x}\right)\left(\frac{y-\bar{y}}{s_y}\right)$$

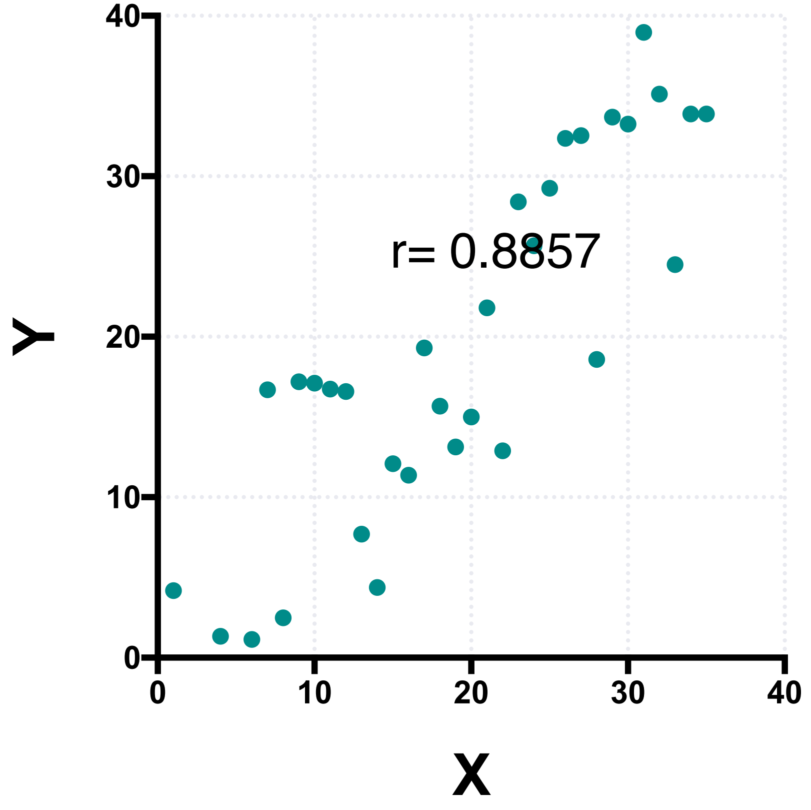
S_x, S_y = standard deviation of x, y $\bar{x}, \bar{y}$, = mean of x, y

- A correlation value of zero, means no relationship between x and y.
- -1 or 1 means a perfect negative or positive relationship (i.e a straight line).

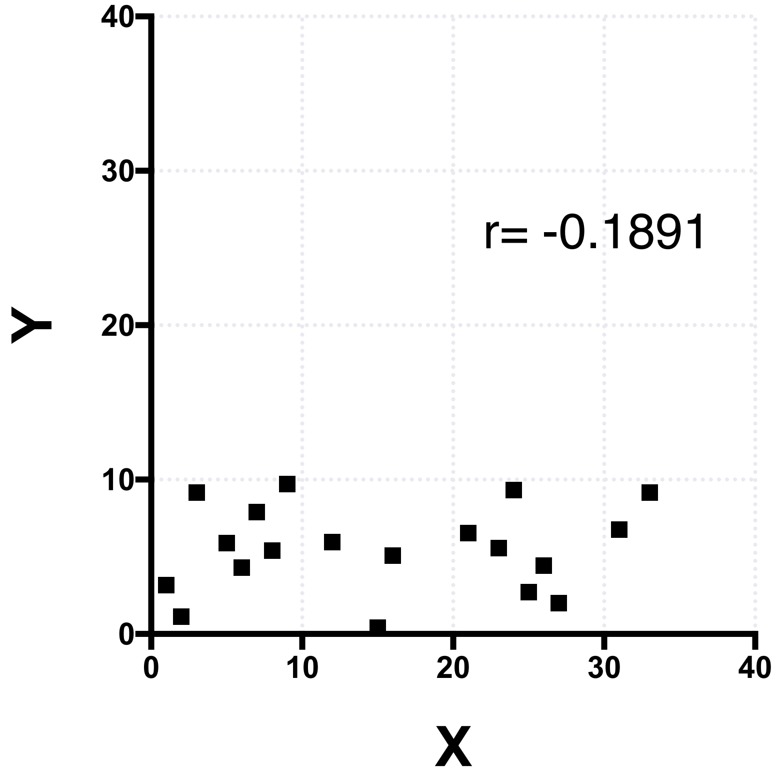
Sources:

Weakly correlated data

- If the relationship is weak, then x won’t tell you about y, they are independent, no need to fit.



Statistical relationship

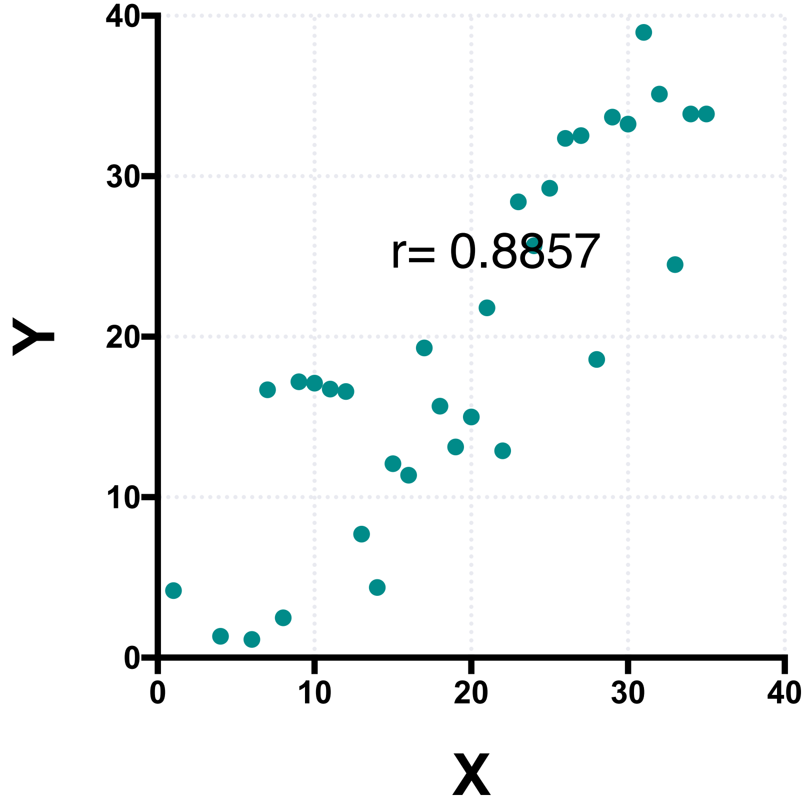


Statistical relationship

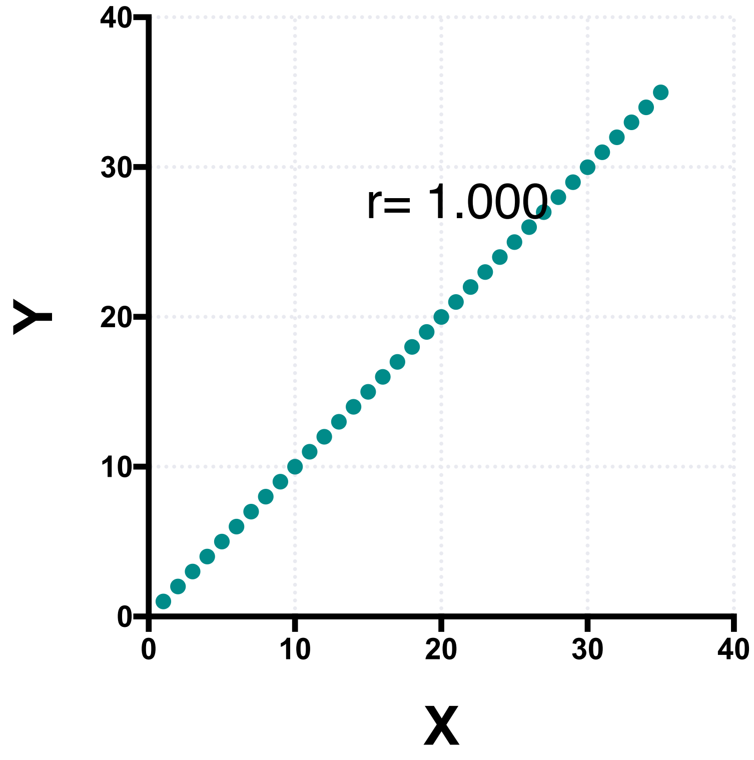
Sources:

Deterministic and Statistical data.

- Statistical relationships (e.g. x: latitude, y:skin cancer mortality, x:height, y:weight), non-perfect relationships.
- Deterministic relationship (e.g. x: x, y: pi*r^2, Ohm’s law, Boyles law), perfect relationships. We don’t need to fit these.



Statistical relationship

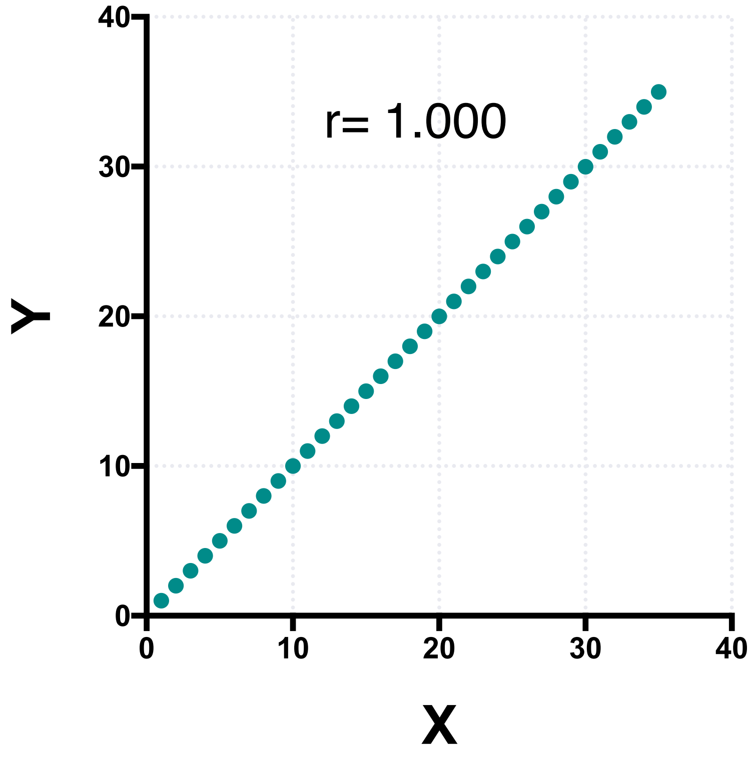


Deterministic relationship

Sources:

Deterministic data.

- With deterministic data we can directly calculate the gradient of the line so there is no need to fit the data. Linear regression is not required.



Deterministic relationship

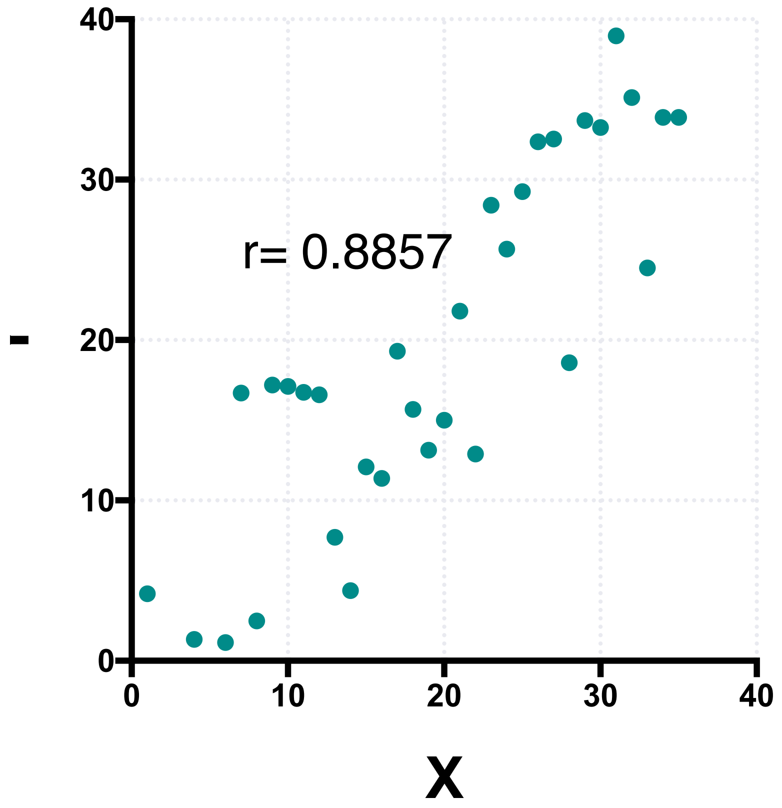
$y_i = b_o + b_1 x_i$

$\mathbf{y_i}$ is the dependent (not predicted) variable for each datapoint **i**

Sources:

Correlation has another purpose

- The square of the correlation coefficient, r^2, is indicative of the relation of x and y in our model.



Statistical relationship

r= 0.8857, r^2 = 0.7845

- r^2, The coefficient of determination, tells us that 78 % of the variability in y can be explained by x, the remainder comes from other sources.

Sources:

So how do we fit?

- Now we know we have suitable data. How do we fit?
- The most common method for linear regression is the method of “linear least squares” which is analytically derived..
- The method is ‘closed form’ and ‘convex’, meaning it can be solved without iteration and there is only one global solution.
- The only constraints are that x and y must be real numbers and also there must be more equations (data pairs) than parameters (b_0, b_1).

Lets take the following equations

$b_0 + 1*b_1 = 6$
$b_0 + 2*b_1 = 5$
$b_0 + 3*b_1 = 7$
$b_0 + 4*b_1 = 10$

$b_0 + b_1x_i = y_i$

The goal is to minimise the sum of error produced by choice of b_0 and b_1 for all the equations
That is when we know we have the best fitting line.

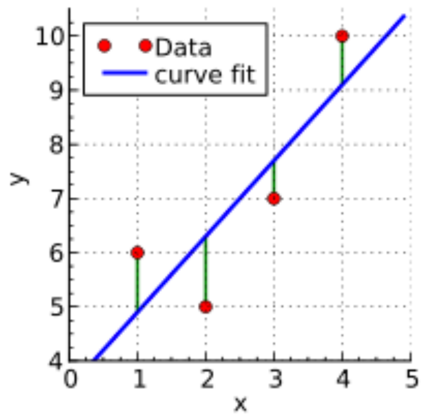
Sources:

So how do we fit?

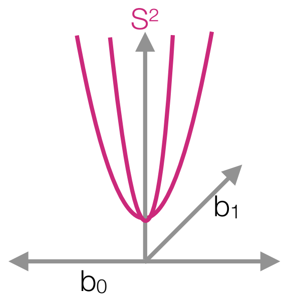
- Whats the intuition for this.

Lets take the following equations. The error for each equation can be defined as:

$b_0 + 1*b_1 = 6$ the error is $6-(b_0 + 1*b_1)$
$b_0 + 2*b_1 = 5$ the error is $5-(b_0 + 2*b_1)$
$b_0 + 3*b_1 = 7$ the error is $7-(b_0 + 3*b_1)$
$b_0 + 4*b_1 = 10$ the error is $10-(b_0 + 4*b_1)$

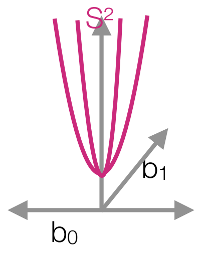


The sum of differences. We want to minimise this, that is the optimum.
$S^2(b_0,b_1) = (6-(b_0 + 1*b_1))^2 + (5-(b_0 + 2*b_1))^2 + (7-(b_0 + 3*b_1))^2 + (10-(b_0 + 4*b_1))^2$



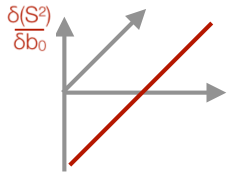
So how do we fit?

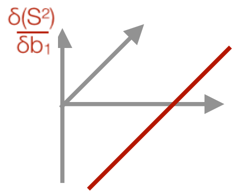
- To find the minima analytically we calculate the partial derivatives of the function with respect to each parameter.



$$\frac{\delta(S^2)}{\delta b_0} = -2\sum_{i=1}^{n}[y_i - (b_0 + b_1.x_i)]$$

$$\frac{\delta(S^2)}{\delta b_1} = -2\sum_{i=1}^{n}[y_i - (b_0 + b_1.x_i)].x_i$$





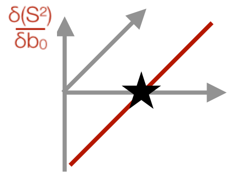
- We calculate the derivatives, because we know that the derivative will equal zero when you are in the global minimum of the cost function. (remember our function is convex, there is only one minima).

Sources:

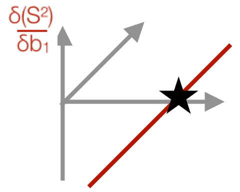
So how do we fit?

- We then evaluate where the derivative is zero. This is the minimum of the sum of differences function. Our optimum solution.

$$\frac{\delta(S^2)}{\delta b_0} = -2\sum_{i=1}^{n}[y_i - (b_0 + b_1.x_i)] = 0$$



$$\frac{\delta(S^2)}{\delta b_1} = -2\sum_{i=1}^{n}[y_i - (b_0 + b_1.x_i)].x_i = 0$$



- When we do this, we have two linear equations

Sources:

Then some linear algebra

$$nb_0 + b_1 \sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i$$

Rearrange $\frac{\delta(S^2)}{\delta b_0}$

$$b_0 \sum_{i=1}^{n} x_i + b_1 \sum_{i=1}^{n} x_i x_i = \sum_{i=1}^{n} x_i y_i$$

Rearrange $\frac{\delta(S^2)}{\delta b_1}$

$$\underbrace{\begin{bmatrix} n & \sum_{i=1}^n x_i \\ \sum_{i=1}^n x_i & \sum_{i=1}^n x_i^2 \end{bmatrix}}_{X'X} \begin{bmatrix} b_0 \\ b_1 \end{bmatrix} = \underbrace{\begin{bmatrix} \sum_{i=1}^n y_i \\ \sum_{i=1}^n x_i\, y_i \end{bmatrix}}_{X'Y},$$

Arrange as a matrix (1)

$$\begin{bmatrix} b_0 \\ b_1 \end{bmatrix} = \frac{1}{n\sum_{i=1}^n x_i^2 - (\sum_{i=1}^n x_i)^2} \begin{bmatrix} \sum_{i=1}^n y_i \sum_{i=1}^n x_i^2 - \sum_{i=1}^n x_i \sum_{i=1}^n x_i\, y_i \\ n\sum_{i=1}^n x_i\, y_i - \sum_{i=1}^n x_i \sum_{i=1}^n y_i \end{bmatrix},$$

invert matrix $\beta = (X'X)^{-1}X'Y$ (2)

$$b_0 = \frac{\sum_{i=1}^n y_i \sum_{i=1}^n x_i^2 - \sum_{i=1}^n x_i \sum_{i=1}^n x_i\, y_i}{n\sum_{i=1}^n x_i^2 - (\sum_{i=1}^n x_i)^2}$$
$$\frac{\bar{y}\left(\sum_{i=1}^n x_i^2\right) - \bar{x}\sum_{i=1}^n x_i\, y_i}{\sum_{i=1}^n x_i^2 - n\bar{x}^2}$$

expanded form (3)

$$b1 = \frac{n\sum_{i=1}^n x_i\, y_i - \sum_{i=1}^n x_i \sum_{i=1}^n y_i}{n\sum_{i=1}^n x_i^2 - (\sum_{i=1}^n x_i)^2}$$
$$\frac{(\sum_{i=1}^n x_i\, y_i) - n\bar{x}\,\bar{y}}{\sum_{i=1}^n x_i^2 - n\bar{x}^2}$$

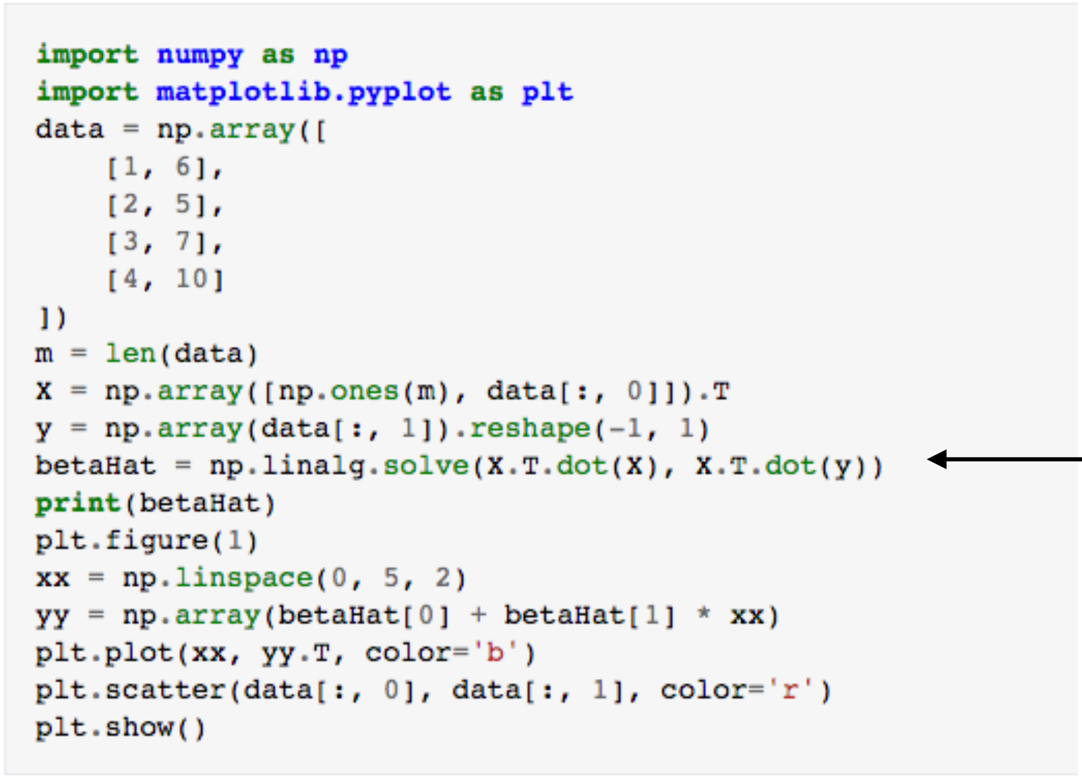
expanded form (4)

Finally we have the solutions. These are neat, but require some processing.

Sources: https://en.wikipedia.org/wiki/Linear_least_squares_(mathematics)

Don’t worry python code is simple.

Some python code for simple linear regression.



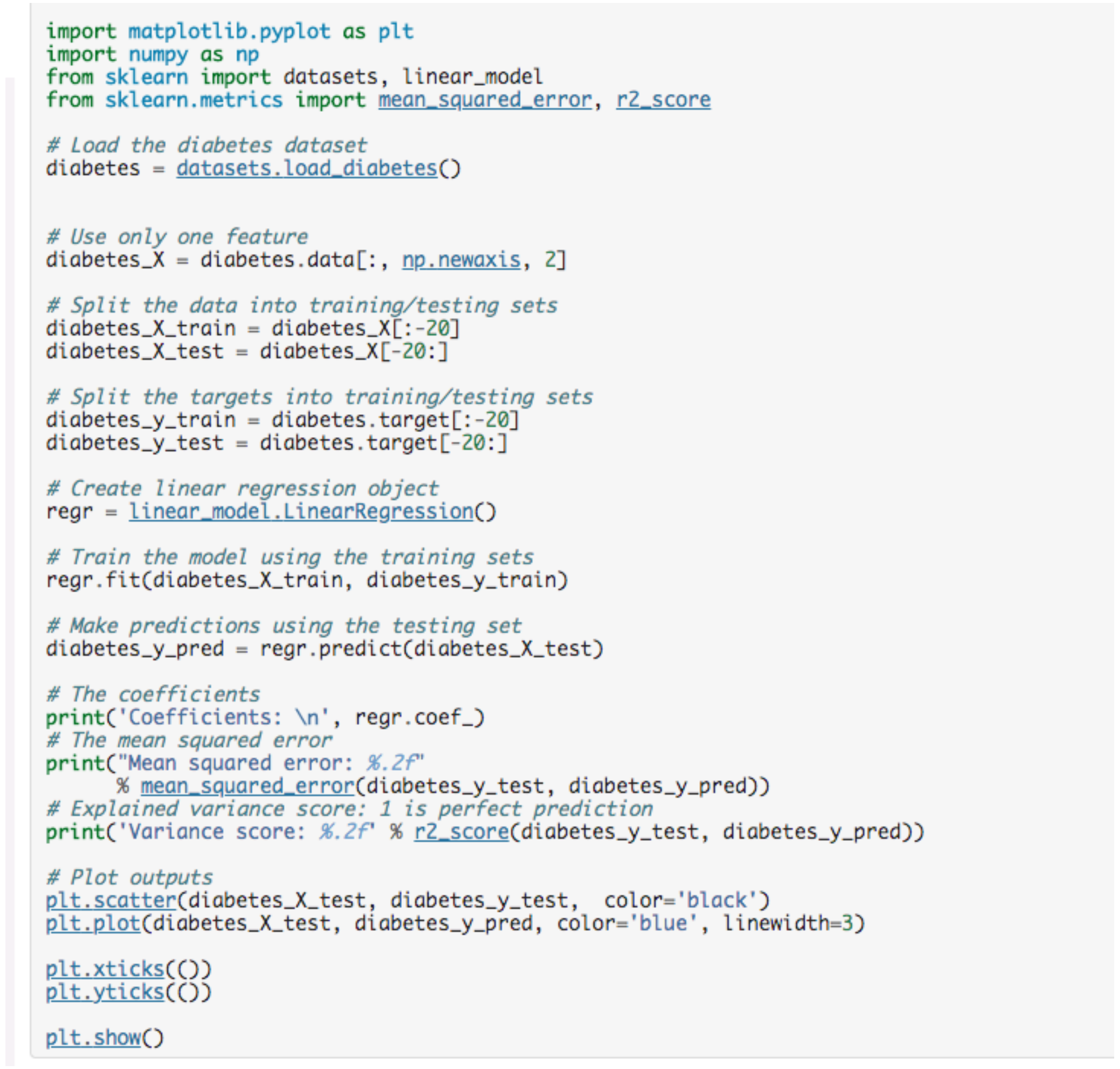
```python
import numpy as np
import matplotlib.pyplot as plt
data = np.array([
    [1, 6],
    [2, 5],
    [3, 7],
    [4, 10]
])
m = len(data)
X = np.array([np.ones(m), data[:, 0]]).T
y = np.array(data[:, 1]).reshape(-1, 1)
betaHat = np.linalg.solve(X.T.dot(X), X.T.dot(y))
print(betaHat)
plt.figure(1)
xx = np.linspace(0, 5, 2)
yy = np.array(betaHat[0] + betaHat[1] * xx)
plt.plot(xx, yy.T, color='b')
plt.scatter(data[:, 0], data[:, 1], color='r')
plt.show()
```

invert matrix

$$\beta = (X'X)^{-1}X'Y$$

Sources: https://en.wikipedia.org/wiki/Linear_least_squares_(mathematics)

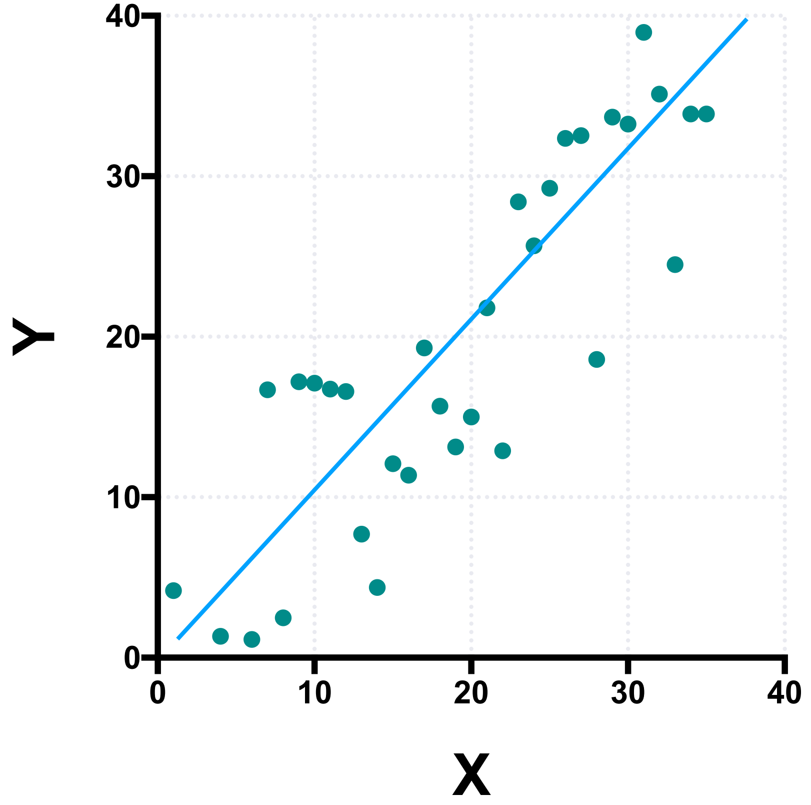
Scikit-learn regression example (no math)



```python
import matplotlib.pyplot as plt
import numpy as np
from sklearn import datasets, linear_model
from sklearn.metrics import mean_squared_error, r2_score

# Load the diabetes dataset
diabetes = datasets.load_diabetes()


# Use only one feature
diabetes_X = diabetes.data[:, np.newaxis, 2]

# Split the data into training/testing sets
diabetes_X_train = diabetes_X[:-20]
diabetes_X_test = diabetes_X[-20:]

# Split the targets into training/testing sets
diabetes_y_train = diabetes.target[:-20]
diabetes_y_test = diabetes.target[-20:]

# Create linear regression object
regr = linear_model.LinearRegression()

# Train the model using the training sets
regr.fit(diabetes_X_train, diabetes_y_train)

# Make predictions using the testing set
diabetes_y_pred = regr.predict(diabetes_X_test)

# The coefficients
print('Coefficients: \n', regr.coef_)
# The mean squared error
print("Mean squared error: %.2f"
      % mean_squared_error(diabetes_y_test, diabetes_y_pred))
# Explained variance score: 1 is perfect prediction
print('Variance score: %.2f' % r2_score(diabetes_y_test, diabetes_y_pred))

# Plot outputs
plt.scatter(diabetes_X_test, diabetes_y_test,  color='black')
plt.plot(diabetes_X_test, diabetes_y_pred, color='blue', linewidth=3)

plt.xticks(())
plt.yticks(())

plt.show()
```

Sources: http://scikit-learn.org/stable/auto_examples/linear_model/plot_ols.html

What we get. Coefficients



We get b0 and b1. We have to make the line.

```
xx = np.linspace(0,40,2)
yy = np.array(b0+ b1* xx)
plot(xx,yy, color='b')
```

$\hat{y}_i = b_o + b_1 x_i$ The equation of a straight-line.

Sources: https://en.wikipedia.org/wiki/Linear_least_squares_(mathematics)

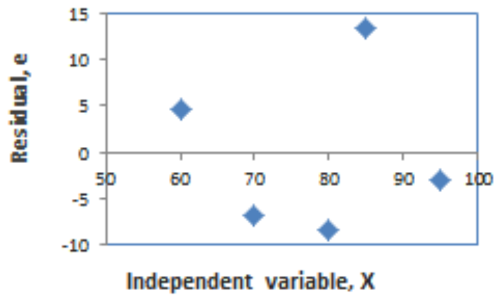
What we get. Residuals

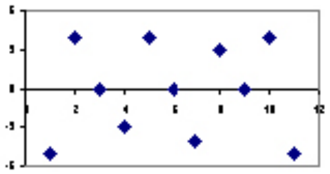
- Residuals measure the error between our data and our predicted line.

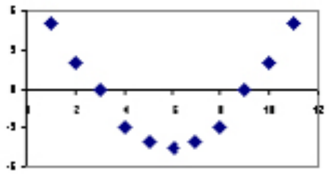
Residual = Observed value - Predicted value

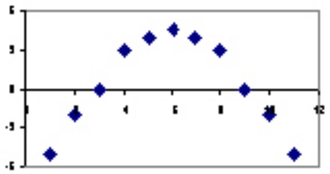
$$e = y - \hat{y}$$

- A residual plot, shows this error for each datapoint x.




Random pattern


Non-random: U-shaped

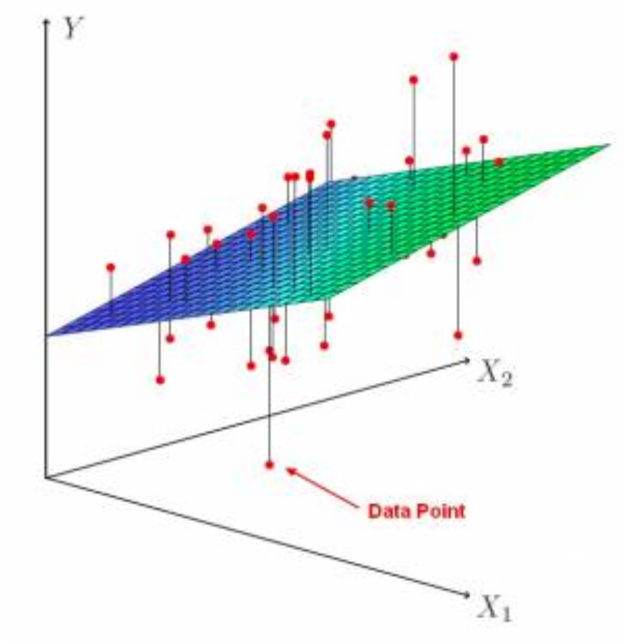

Non-random: Inverted U

If your data, doesn't look randomly distributed, rethink your model.

Sources: http://stattrek.com/regression/residual-analysis.aspx?Tutorial=AP

simple linear regression to multiple linear regression

- Simple linear regression involves one independent variable
- Multiple linear regression can involve > 1 independent variables (predictors)



$$\hat{Y} = B.X$$

where:

- $\hat{Y}$ is a [N x 1] vector representing the predicted score where N is the sample size
- X is a [N x k] matrix representing the predictors variables where k is the number of predictors
- B is a [k x 1] vector representing the regression coefficients (β)
- the regression constant is assumed to be zero.

$$\beta = (X'X)^{-1}X'Y$$

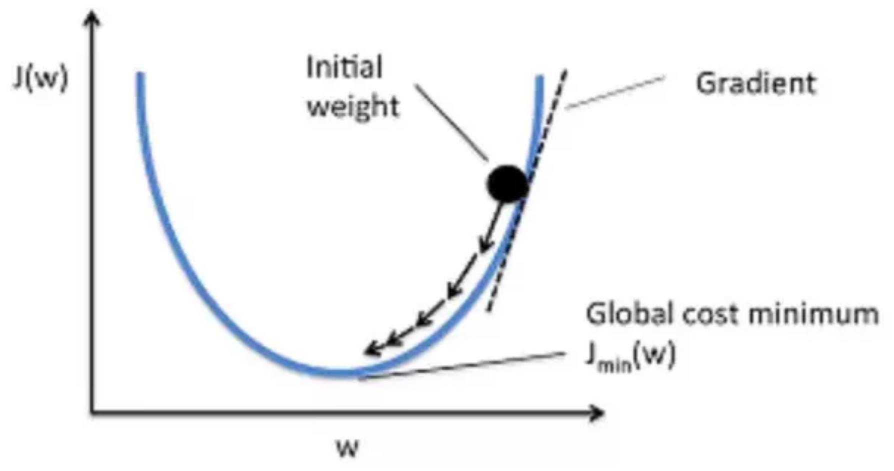
e.g. two independent variables.

- Good news is we can solve with the same approach, just add more x columns (independent variables).

Sources: https://gerardnico.com/data_mining/multiple_regression

Iterative methods for solving linear regression

- So far we have just looked at analytical methods. Most often if an analytical method exists for your function, you use it for fitting.
- The alternative is an iterative method. Here you try and find the minimum by calculating the rate of change when we change parameters.
- Today we don't go too much into this, as it is a whole new topic.

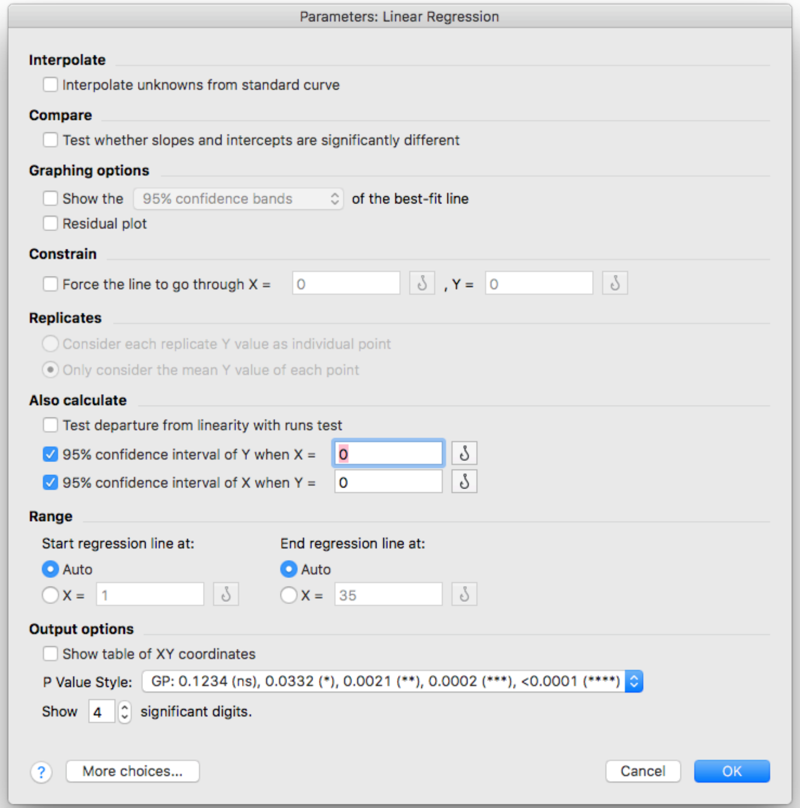


If your linear regression data is of very large size and of high dimension it can be better to fit your data using something like stochastic gradient descent rather than least squares.

Sources: https://gerardnico.com/data_mining/multiple_regression

Its easy-peasy to do regression with Graphpad Prism.

- Good for checking your methods :-). But only if simple case.



Parameters: Linear Regression

Interpolate

☐ Interpolate unknowns from standard curve

Compare

☐ Test whether slopes and intercepts are significantly different

Graphing options

☐ Show the 95% confidence bands of the best-fit line

☐ Residual plot

Constrain

☐ Force the line to go through X = 0 , Y = 0

Replicates

○ Consider each replicate Y value as individual point

● Only consider the mean Y value of each point

Also calculate

☐ Test departure from linearity with runs test

☑ 95% confidence interval of Y when X = 0

☑ 95% confidence interval of X when Y = 0

Range

Start regression line at: ● Auto ○ X = 1

End regression line at: ● Auto ○ X = 35

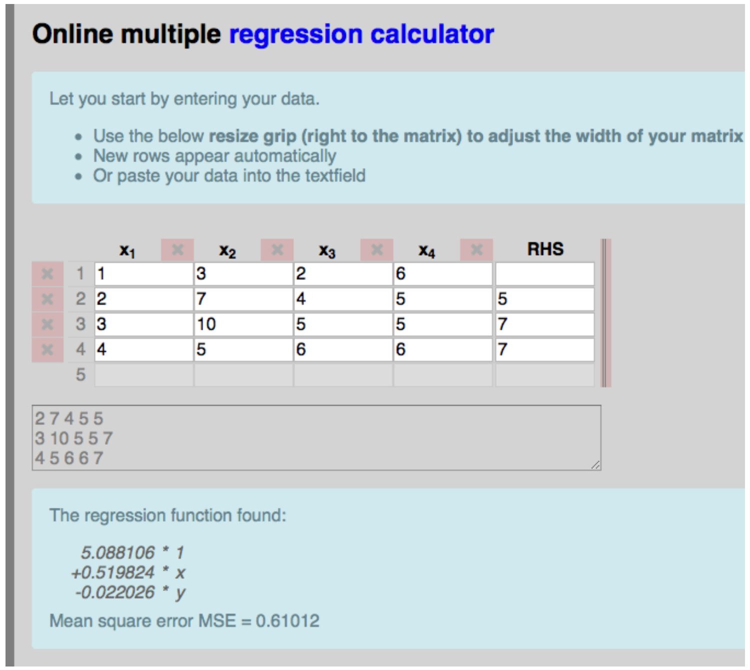
Output options

☐ Show table of XY coordinates

P Value Style: GP: 0.1234 (ns), 0.0332 (*), 0.0021 (**), 0.0002 (***), <0.0001 (****)

Show 4 significant digits.

- Also online methods for doing multiple linear regression
- http://regression.structural-analyser.com



Online multiple regression calculator

Let you start by entering your data.

- Use the below **resize grip (right to the matrix) to adjust the width of your matrix**
- New rows appear automatically
- Or paste your data into the textfield

	x_1	x_2	x_3	x_4	RHS
1	1	3	2	6	
2	2	7	4	5	5
3	3	10	5	5	7
4	4	5	6	6	7
5					

```
2 7 4 5 5
3 10 5 5 7
4 5 6 6 7
```

The regression function found:

*5.088106 * 1*
*+0.519824 * x*
*-0.022026 * y*

Mean square error MSE = 0.61012

Sources:

Limitations of linear regression

- Only describes linear relationships
- Linear Regression Is Sensitive to Outliers
- Data Must Be Independent (Thats why they are called ‘independent’)

Advantages of linear regression

- Fast.
- Easy to use.
- Easy to understand.

Conclusion

- Its good to know how things work.
- People take linear regression for granted, it is used a lot.

Sources:

Practicals.

- https://github.com/dwaithe/linear_regression_practical
- clone the repository. download: Linear Regression practical.ipynb
- Work through ask questions.

Sources: