Linear Regression

Presented by David John Baker
January 2020

FLATIRON SCHOOL

Why Linear Regression ?

- LR is a fundamental tool in the data scientist’s kit.
- Practically speaking, using it is one or two lines of code.
- But it’s crucial for us to understand the theory underlying it:
  - Building block for more complex tools
  - We are better data scientists if we understand both HOW and WHY

But first... what is a model?

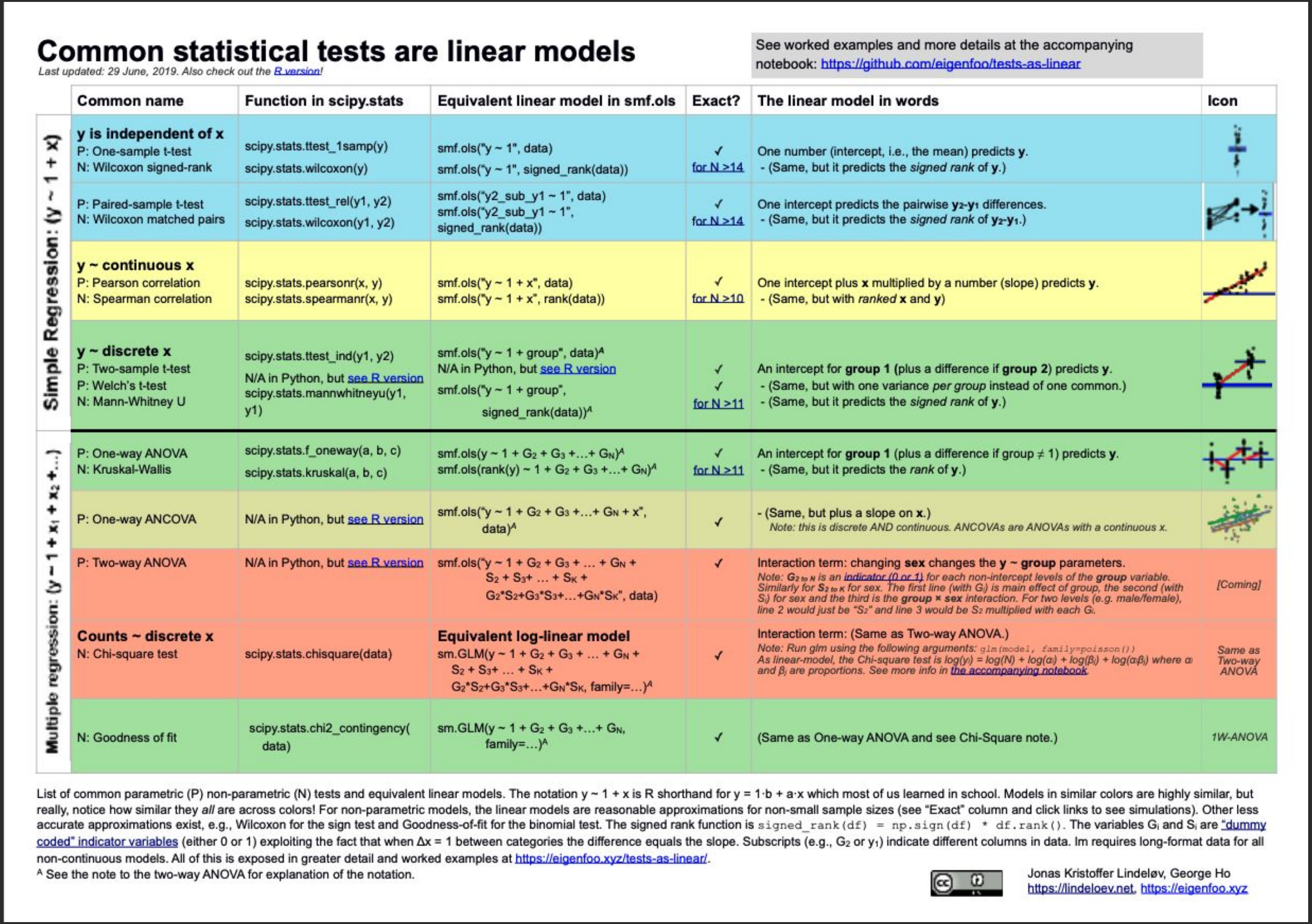
- Take three minutes to talk to your partner and answer the question… **<u>“What is a statistical model?”</u>**
- Come up with a one line definition
- Be prepared to share it with the class
- Thoughts to get you going:
  - What are models used for?
  - What’s in a model?
  - Why do we make models?

Statistical (Mathematical) Models

- Formalization and quantification of our assumptions about the state of the world
- Distillation of the data (think, mean/median/model)
- Gives us common mathematical language to talk about ways in which we are wrong, incomplete, or inaccurate
- Also used for inference, to make predictions about future data
- “All models wrong, some are useful” - George Box

(Simple) Linear Regression

- (Simple): functions of a single variable: $Y = f(X)$
- Linear: models are lines
- Regression: dependent variable is continuously-valued



Common statistical tests are linear models

Last updated: 29 June, 2019. Also check out the R version!

See worked examples and more details at the accompanying notebook: https://github.com/eigenfoo/tests-as-linear

	Common name	Function in scipy.stats	Equivalent linear model in smf.ols	Exact?	The linear model in words	Icon

Simple Regression: (y ~ 1 + x)	**y is independent of x**
	P: One-sample t-test
	N: Wilcoxon signed-rank
		scipy.stats.ttest_1samp(y)
		scipy.stats.wilcoxon(y)
			smf.ols("y ~ 1", data)
			smf.ols("y ~ 1", signed_rank(data))
				✓
				for N >14
					One number (intercept, i.e., the mean) predicts **y**.
					- (Same, but it predicts the *signed rank* of **y**.)	

	P: Paired-sample t-test
	N: Wilcoxon matched pairs
		scipy.stats.ttest_rel(y1, y2)
		scipy.stats.wilcoxon(y1, y2)
			smf.ols("y2_sub_y1 ~ 1", data)
			smf.ols("y2_sub_y1 ~ 1", signed_rank(data))
				✓
				for N >14
					One intercept predicts the pairwise **y_2-y_1** differences.
					- (Same, but it predicts the *signed rank* of **y_2-y_1**.)	

	y ~ continuous x
	P: Pearson correlation
	N: Spearman correlation
		scipy.stats.pearsonr(x, y)
		scipy.stats.spearmanr(x, y)
			smf.ols("y ~ 1 + x", data)
			smf.ols("y ~ 1 + x", rank(data))
				✓
				for N >10
					One intercept plus **x** multiplied by a number (slope) predicts **y**.
					- (Same, but with *ranked* **x** and **y**)	

	y ~ discrete x
	P: Two-sample t-test
	P: Welch's t-test
	N: Mann-Whitney U
		scipy.stats.ttest_ind(y1, y2)
		N/A in Python, but see R version
		scipy.stats.mannwhitneyu(y1, y1)
			smf.ols("y ~ 1 + group", data)^A
			N/A in Python, but see R version
			smf.ols("y ~ 1 + group", signed_rank(data))^A
				✓
				✓
				for N >11
					An intercept for **group 1** (plus a difference if **group 2**) predicts **y**.
					- (Same, but with one variance *per group* instead of one common.)
					- (Same, but it predicts the *signed rank* of **y**.)	

Multiple regression: (y ~ 1 + x_1 + x_2 +...)	P: One-way ANOVA
	N: Kruskal-Wallis
		scipy.stats.f_oneway(a, b, c)
		scipy.stats.kruskal(a, b, c)
			smf.ols(y ~ 1 + G_2 + G_3 +...+ G_N)^A
			smf.ols(rank(y) ~ 1 + G_2 + G_3 +...+ G_N)^A
				✓
				for N >11
					An intercept for **group 1** (plus a difference if group ≠ 1) predicts **y**.
					- (Same, but it predicts the *rank* of **y**.)	

	P: One-way ANCOVA	N/A in Python, but see R version	smf.ols("y ~ 1 + G_2 + G_3 +...+ G_N + x", data)^A	✓	- (Same, but plus a slope on **x**.) *Note: this is discrete AND continuous. ANCOVAs are ANOVAs with a continuous x.*	
	P: Two-way ANOVA	N/A in Python, but see R version	smf.ols("y ~ 1 + G_2 + G_3 + ... + G_N + S_2 + S_3+ ... + S_K + G_2*S_2+G_3*S_3+...+G_N*S_K", data)	✓	Interaction term: changing **sex** changes the **y ~ group** parameters. *Note: **$G_{2 \text{ to } N}$** is an indicator (0 or 1) for each non-intercept levels of the **group** variable. Similarly for **$S_{2 \text{ to } K}$** for sex. The first line (with G_i) is main effect of group, the second (with S_i) for sex and the third is the **group × sex** interaction. For two levels (e.g. male/female), line 2 would just be "S_2" and line 3 would be S_2 multiplied with each G_i.*	*[Coming]*
	Counts ~ discrete x N: Chi-square test	scipy.stats.chisquare(data)	**Equivalent log-linear model** sm.GLM(y ~ 1 + G_2 + G_3 + ... + G_N + S_2 + S_3+ ... + S_K + G_2*S_2+G_3*S_3+...+G_N*S_K, family=...)^A	✓	Interaction term: (Same as Two-way ANOVA.) *Note: Run glm using the following arguments:* `glm(model, family=poisson())` *As linear-model, the Chi-square test is $\log(y_i) = \log(N) + \log(\alpha_i) + \log(\beta_j) + \log(\alpha_i\beta_j)$ where α_i and β_j are proportions. See more info in the accompanying notebook.*	*Same as Two-way ANOVA*
	N: Goodness of fit	scipy.stats.chi2_contingency(data)	sm.GLM(y ~ 1 + G_2 + G_3 +...+ G_N, family=...)^A	✓	(Same as One-way ANOVA and see Chi-Square note.)	*1W-ANOVA*

List of common parametric (P) non-parametric (N) tests and equivalent linear models. The notation y ~ 1 + x is R shorthand for y = 1·b + a·x which most of us learned in school. Models in similar colors are highly similar, but really, notice how similar they *all* are across colors! For non-parametric models, the linear models are reasonable approximations for non-small sample sizes (see "Exact" column and click links to see simulations). Other less accurate approximations exist, e.g., Wilcoxon for the sign test and Goodness-of-fit for the binomial test. The signed rank function is `signed_rank(df) = np.sign(df) * df.rank()`. The variables G_i and S_i are "dummy coded" indicator variables (either 0 or 1) exploiting the fact that when Δx = 1 between categories the difference equals the slope. Subscripts (e.g., G_2 or y_1) indicate different columns in data. lm requires long-format data for all non-continuous models. All of this is exposed in greater detail and worked examples at https://eigenfoo.xyz/tests-as-linear/.

^{A} See the note to the two-way ANOVA for explanation of the notation.

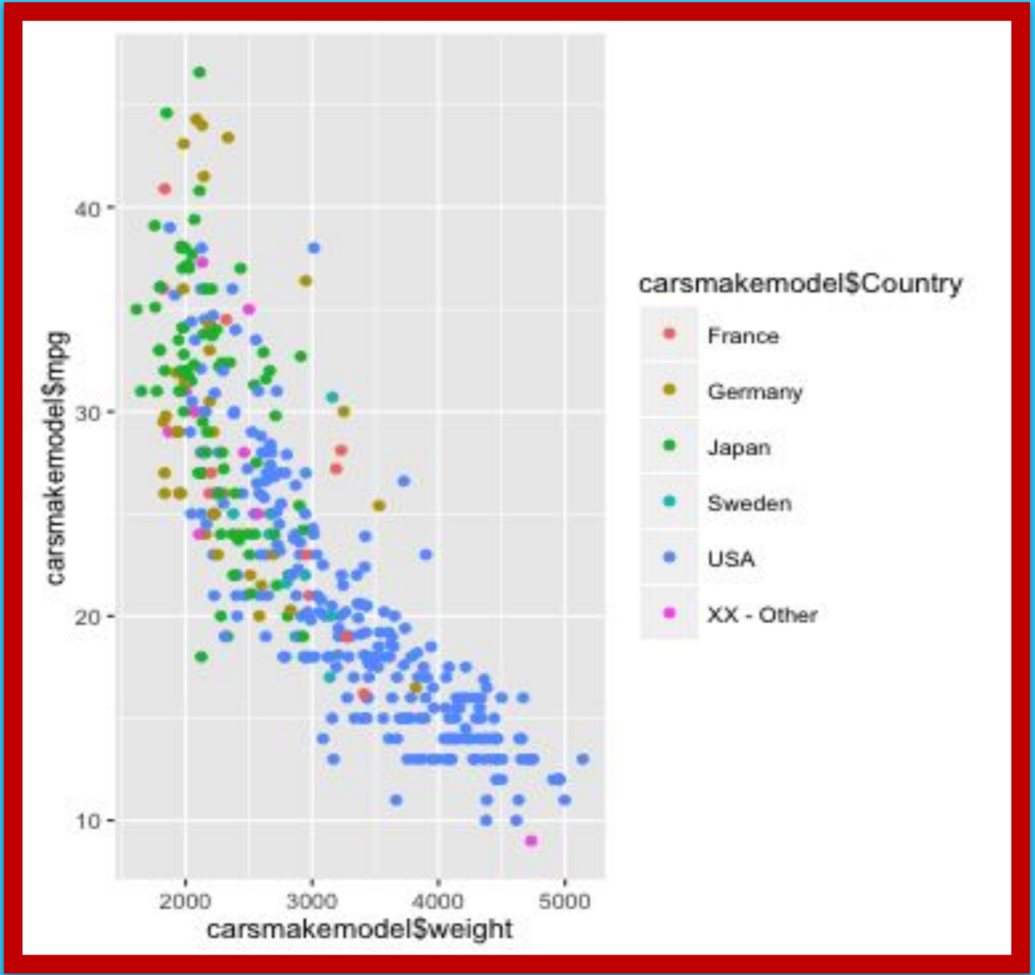
Jonas Kristoffer Lindeløv, George Ho
https://lindeloev.net, https://eigenfoo.xyz

https://eigenfoo.xyz/tests-as-linear/

Using one Variable to Predict Another

- As population density increases, so do housing prices.
- As the number of trees decreases, the concentration of CO_2 goes up.

Example:
Car Weight and MPG



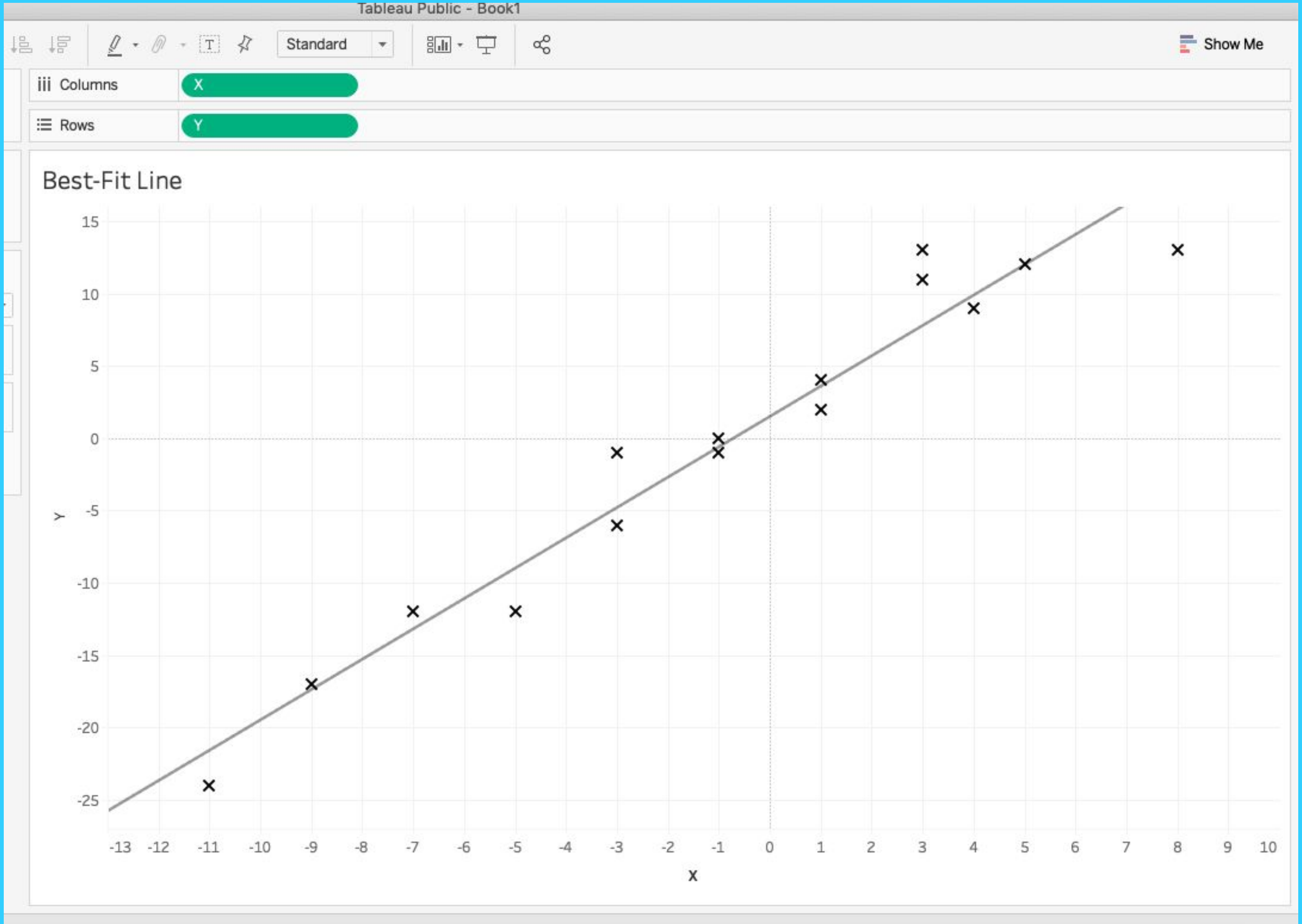


Tableau Public - Book1

Columns: X

Rows: Y

Best-Fit Line

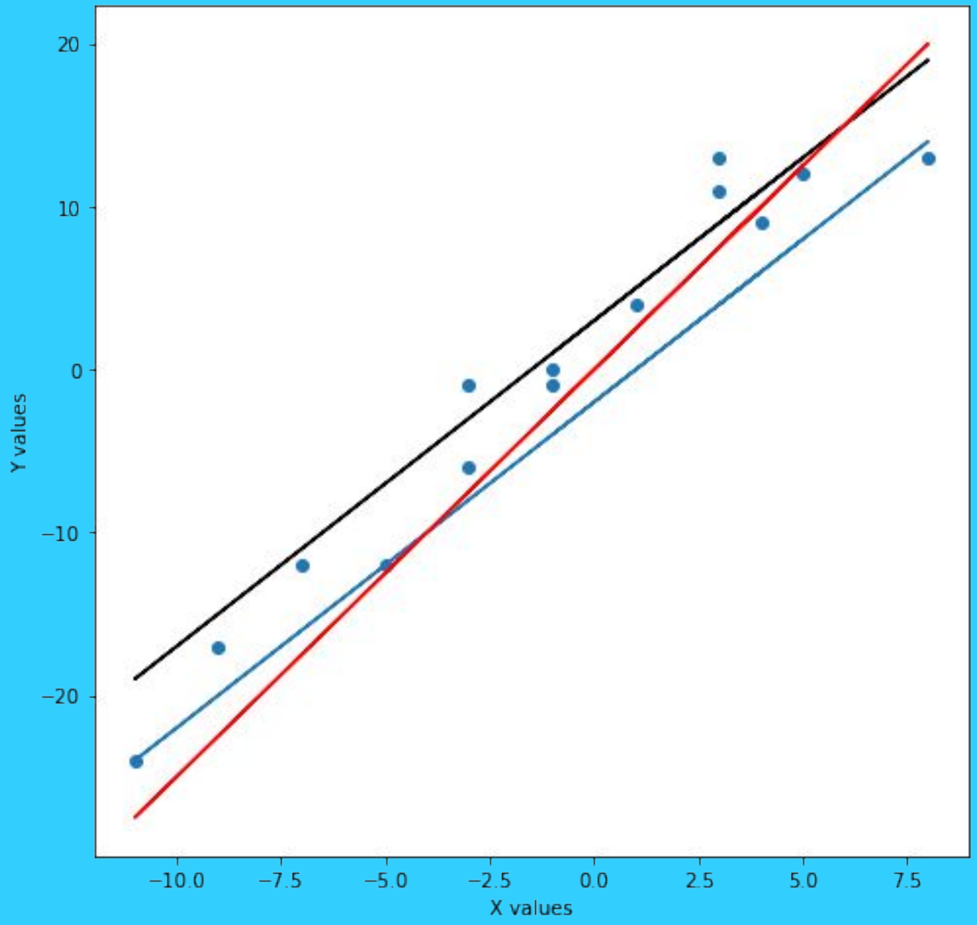
A Line as a Model

- Predictions for *all* values of the X variable
  - Model shape: $\hat{y} = \beta_1 x + \beta_0$
- Error as the distance between real and predicted values:

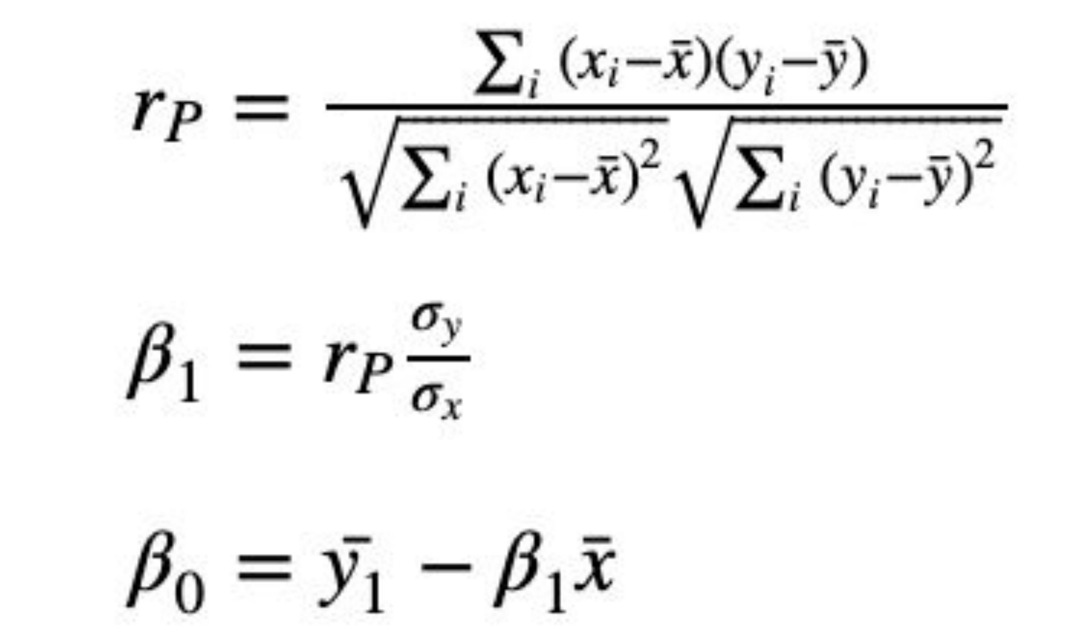
$$E = y - \hat{y}$$
$$E^2 = (y - \hat{y})^2$$

Goal: Minimize Error

- Which of these lines fits the data best?



How to Construct the Best-Fit Line

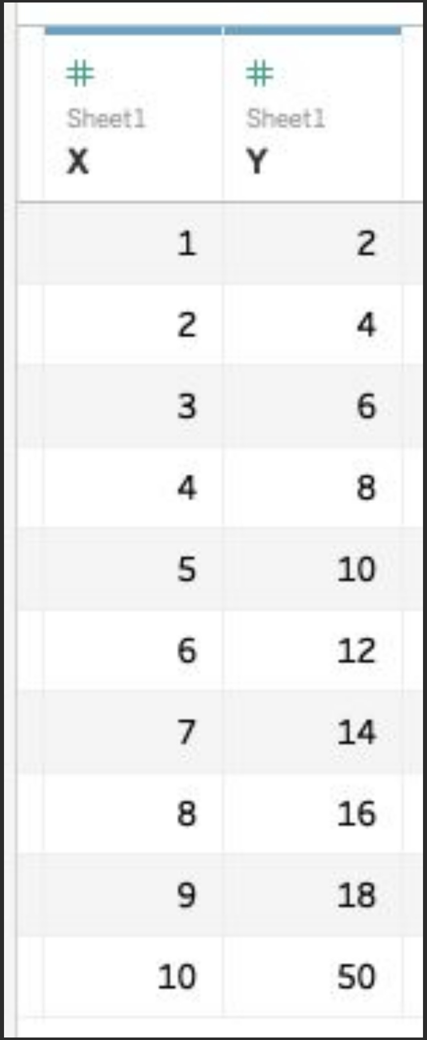


$$r_P = \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_i (x_i - \bar{x})^2}\sqrt{\sum_i (y_i - \bar{y})^2}}$$

$$\beta_1 = r_P \frac{\sigma_y}{\sigma_x}$$

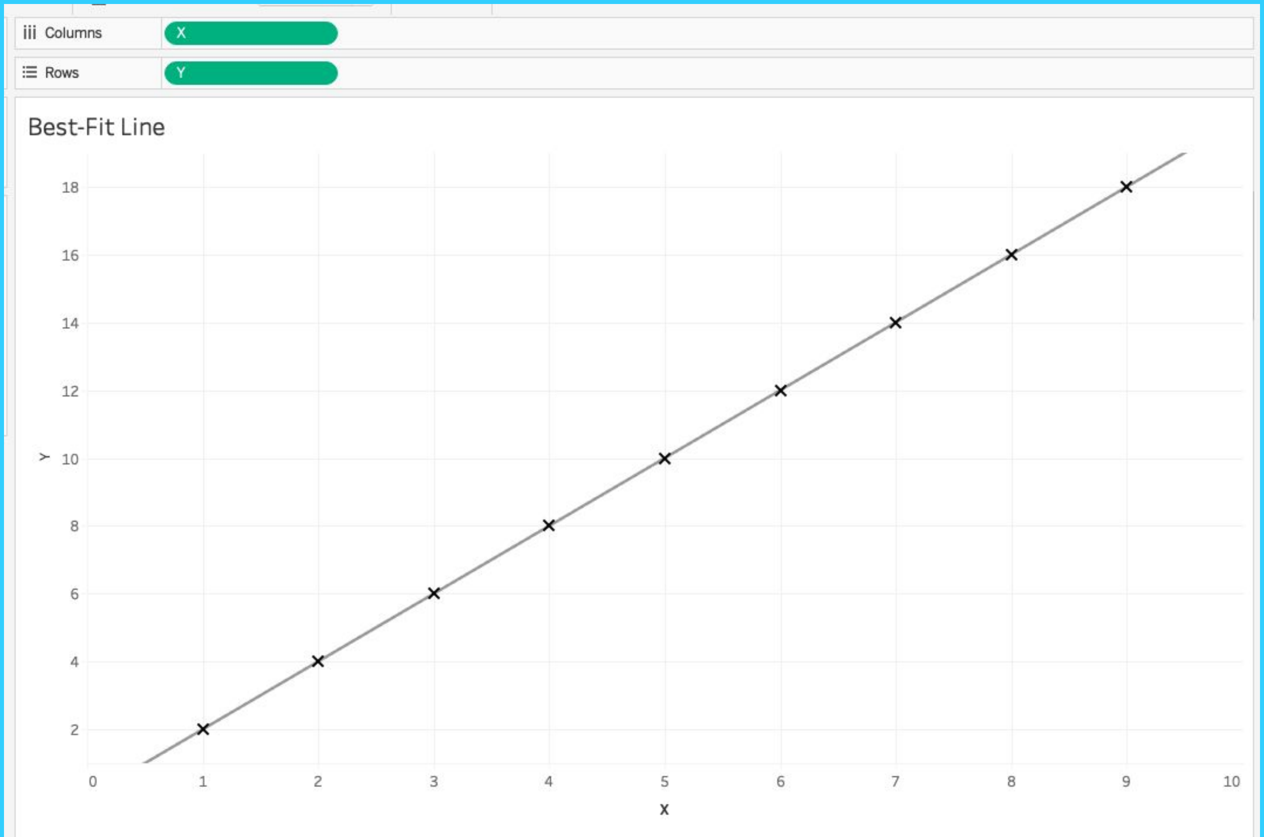
$$\beta_0 = \bar{y}_1 - \beta_1 \bar{x}$$

Outliers

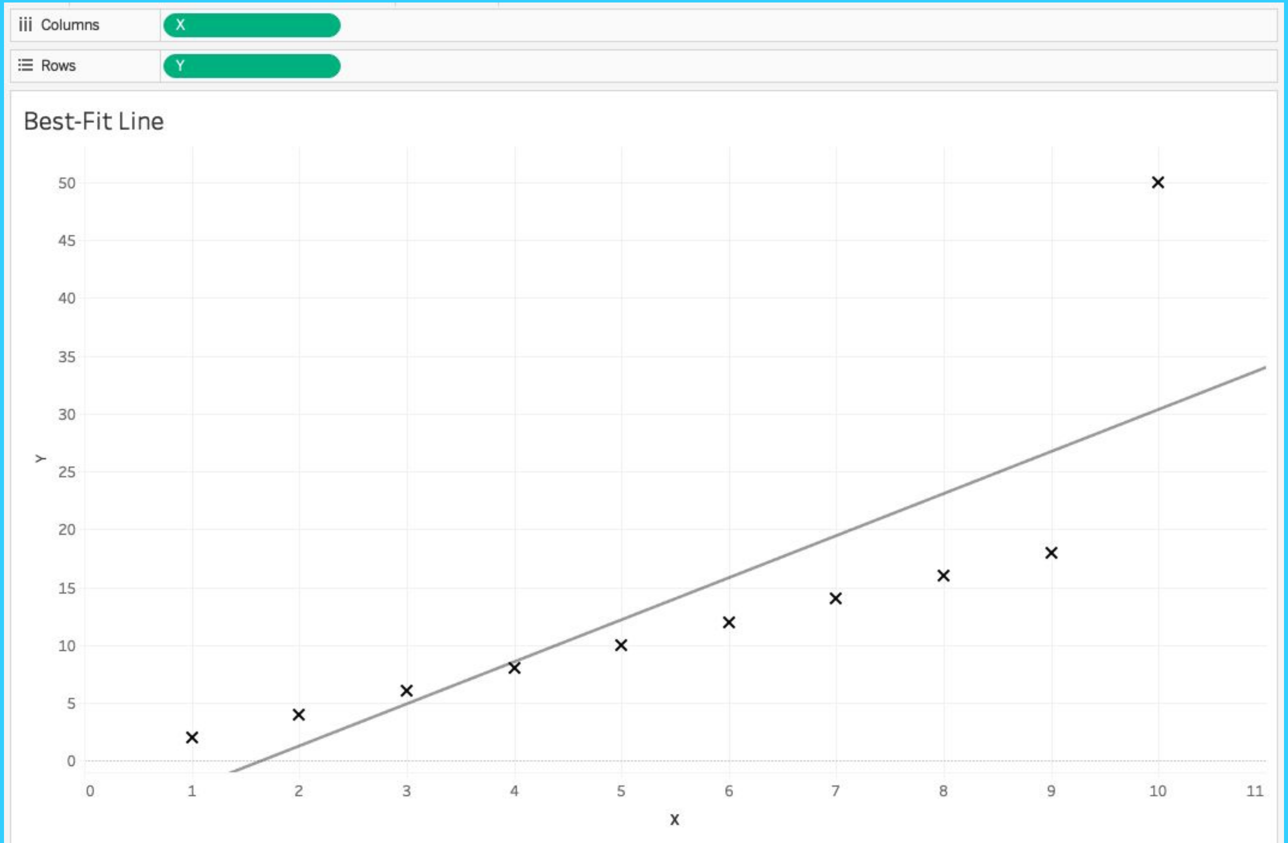


#	#
Sheet1	Sheet1
X	Y
1	2
2	4
3	6
4	8
5	10
6	12
7	14
8	16
9	18
10	50

Dropping Outliers



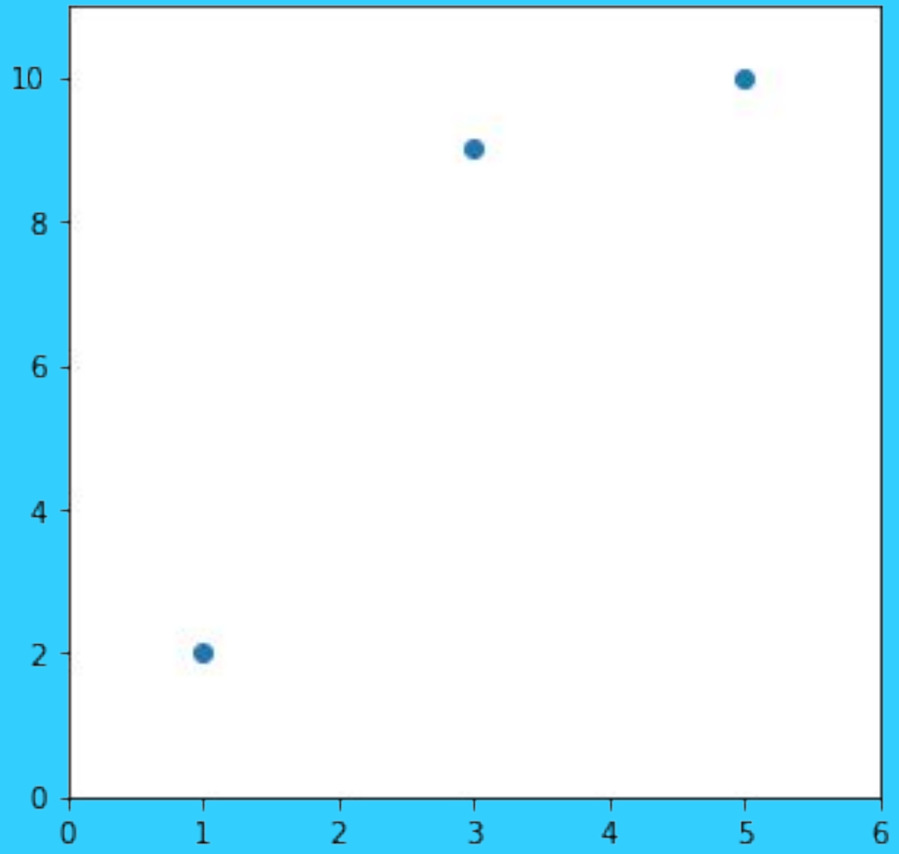
Keeping Outliers



Columns X

Rows Y

Example:
Construct the best-fit line for the points:
(1, 2), (3, 9), and (5, 10).



$$x_i : [1, 3, 5]$$
$$y_i : [2, 9, 10]$$

First we'll calculate x_bar and y_bar:

$$\bar{x} = \frac{1+3+5}{3} = 3$$

$$\bar{y} = \frac{2+9+10}{3} = 7$$

$$x_i : [1, 3, 5]$$
$$y_i : [2, 9, 10]$$

Now we can calculate r_P:

$$\Sigma(x_i - \bar{x})(y_i - \bar{y}) = (1 - 3)(2 - 7) + (3 - 3)(9 - 7) + (5 - 3)(10 - 7) = 16$$

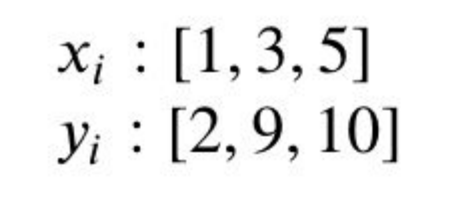
$$\Sigma(x_i - \bar{x})^2 = (1 - 3)^2 + (3 - 3)^2 + (5 - 3)^2 = 8$$

$$\Sigma(y_i - \bar{y})^2 = (2 - 7)^2 + (9 - 7)^2 + (10 - 7)^2 = 38$$

$$x_i \ : [1, 3, 5]$$
$$y_i \ : [2, 9, 10]$$

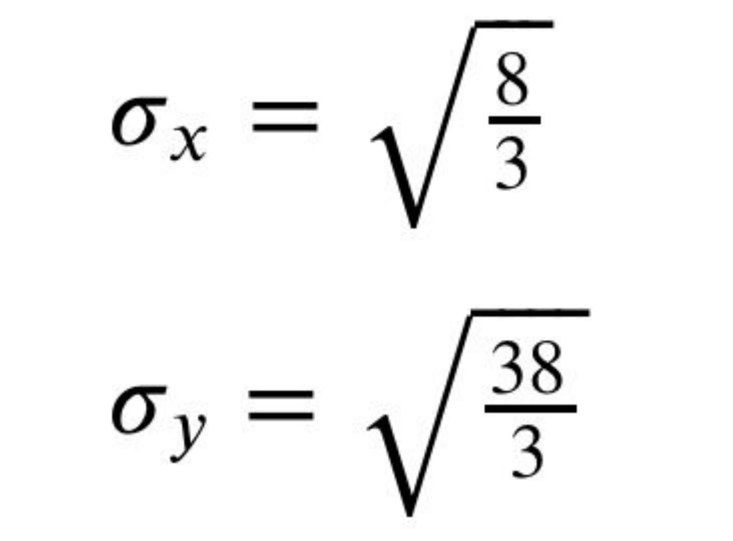
Now we can calculate r_P:

$$r_P = \frac{\Sigma(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\Sigma_i(x_i - \bar{x})^2}\sqrt{\Sigma_i(y_i - \bar{y})^2}} = \frac{16}{\sqrt{(8)(38)}} = \frac{4}{\sqrt{19}}$$



$$x_i : [1, 3, 5]$$
$$y_i : [2, 9, 10]$$

Now we'll calculate the standard deviations of x and y ...



$$\sigma_x = \sqrt{\frac{8}{3}}$$

$$\sigma_y = \sqrt{\frac{38}{3}}$$

$$x_i : [1, 3, 5]$$
$$y_i : [2, 9, 10]$$

… and use those to calculate beta_1:

$$\beta_1 = r_P \frac{\sigma_y}{\sigma_x} = \frac{4}{\sqrt{19}} \left(\frac{\sqrt{\frac{38}{3}}}{\sqrt{\frac{8}{3}}} \right) = \frac{4}{\sqrt{19}} \left(\frac{\sqrt{38}}{\sqrt{8}} \right) = \frac{4\sqrt{2}}{2\sqrt{2}} = 2$$

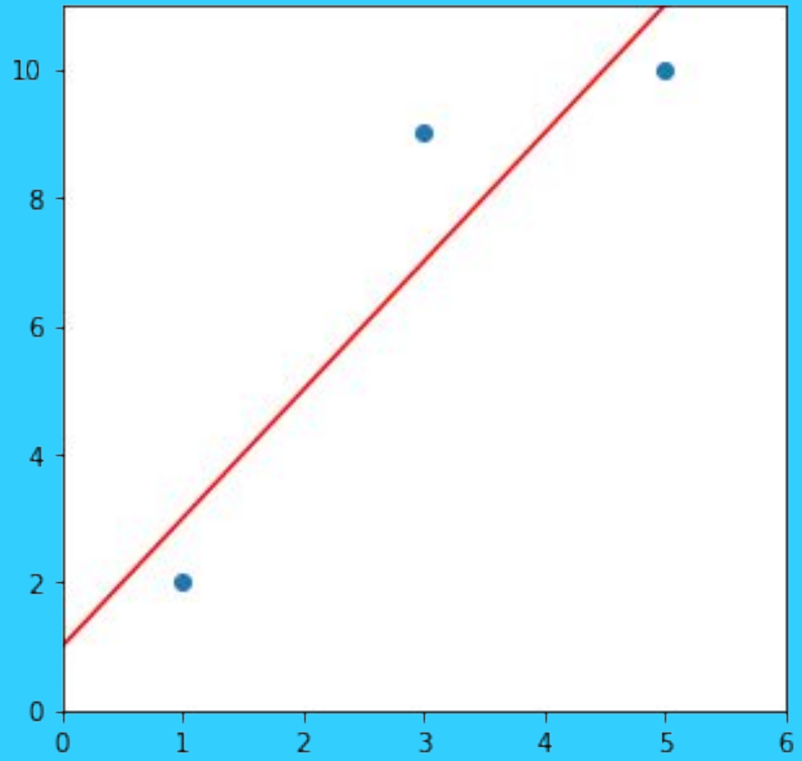
$$x_i : [1, 3, 5]$$
$$y_i : [2, 9, 10]$$

Finally, we’ll use beta_1 to calculate beta_0:

$$\beta_0 = \bar{y} - \beta_1 \bar{x} = 7 - (2)(3) = 1$$

So now we have our linear equation!

$$\hat{y} = \beta_1 x + \beta_0 = 2x + 1$$


10
8
6
4
2
0
0
1
2
3
4
5
6