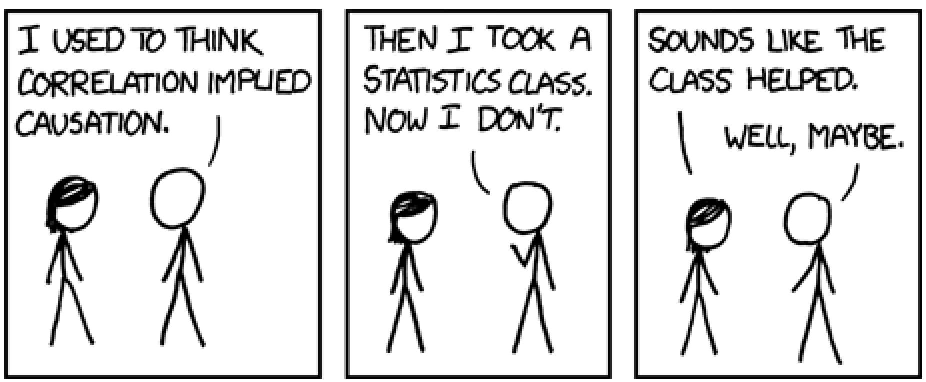
# Regressions

# Cause and Effect

**Correlation**: Two variables are correlated when changes in one variable occur in a pattern corresponding to changes in the other.

# Cause and Effect

**Causation**: One variable moves, and the second variable changes because of the movement of the first.



# Questioning Causality

When we suspect a causal relationship (that $x$ causes $y$), it is important to ask ourselves several questions:

1. Is it possible that $y$ causes $x$ instead?
2. Is it possible that $z$ (a new factor that we haven't considered before) is causing both $x$ and $y$?
3. Could the relationship have been observed by chance?

# Establishing Causality

In order to establish causality, we need to meet several conditions:

- We can explain (or at lest hypothesize) **why** $x$ causes $y$
- We can demonstrate that **nothing else is driving the changes** (within reason)
- We can show that there is a **correlation** between $x$ and $y$

# Ceteris Paribus

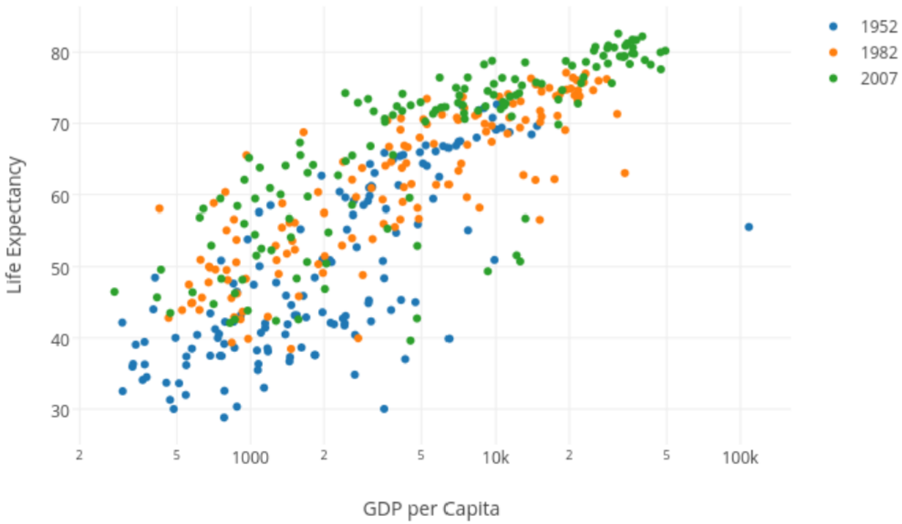
*ceteris paribus* means "all else equal"

# Regression analysis

- Allows us to **act as if nothing else were changing**
- Mathematicaly isolates the effect of each individual **variable** on the outcome of interest
    - Variables are the factors that we want to include in our model


# Regression analysis

- Think about it like a trend line!

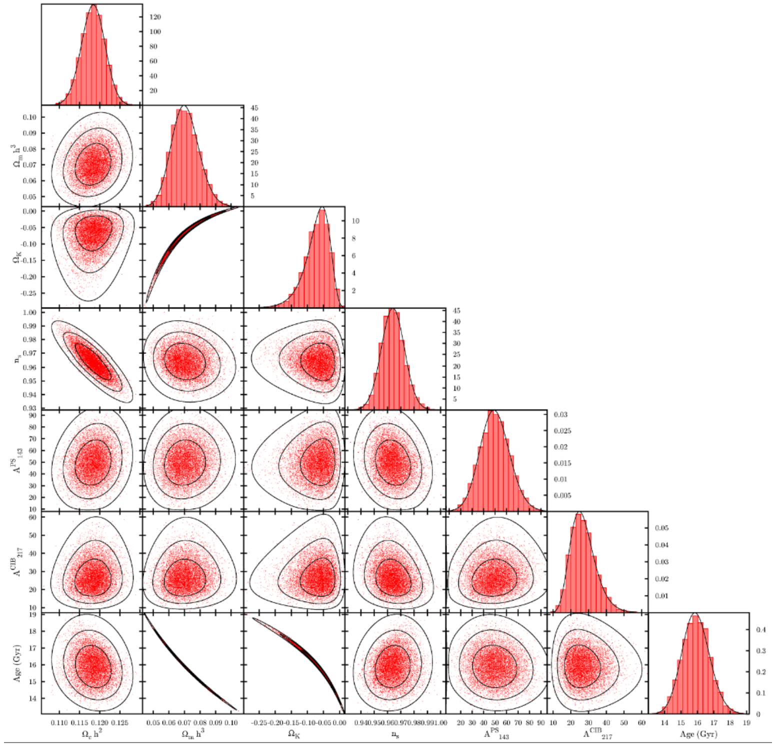



# Regression analysis

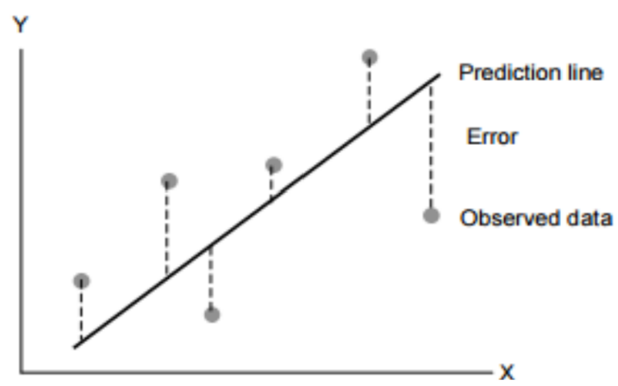
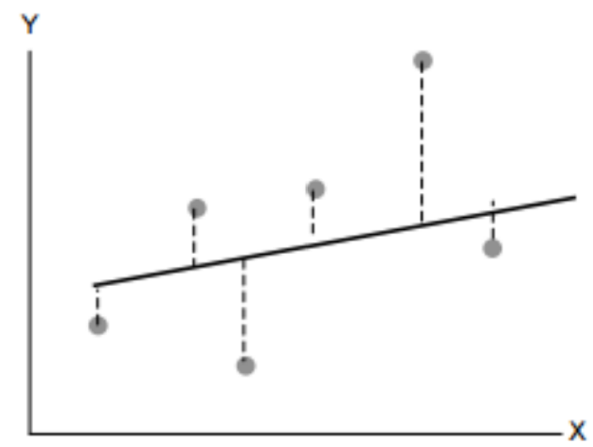
Whoops! It turns out there is another variable. What now!?

# Regression analysis

Yikes....

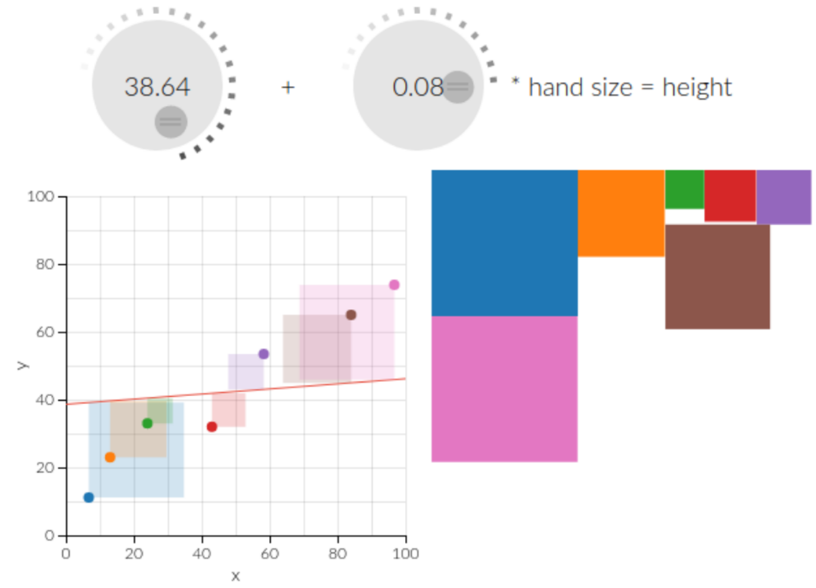


# Minimize Errors and Best Fit Lines

| Best Fit | Something Else |
|---|---|
|   |   |

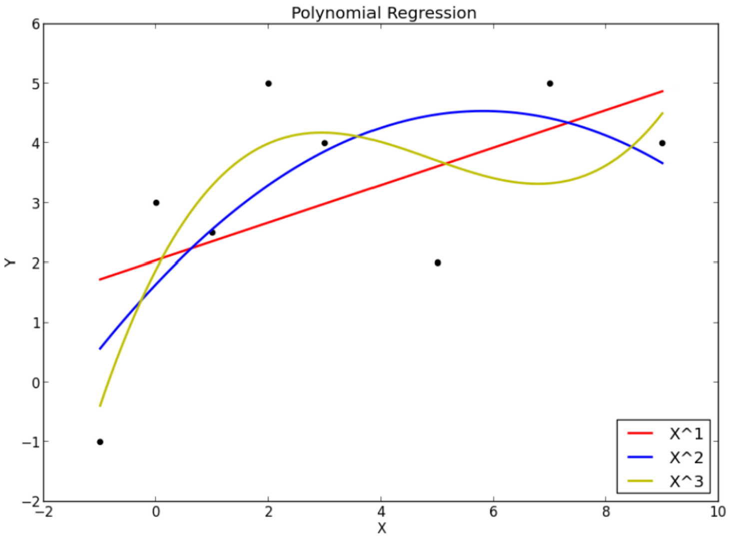
# Minimize Errors and Best Fit Lines

Try it by hand!



# Why LINEAR regression?

- Faster
- More honest



# OLS in Python (with Statsmodels)

```python
import pandas as pd
import statsmodels.formula.api as smf

data = pd.read_csv(
    "https://github.com/dustywhite7/pythonMikkeli/raw/master/exampleData/fishWeight.csv")

reg = smf.ols("Weight ~ Length1", data=data)

reg = reg.fit()

print(reg.summary())
```

In [5]:
```
reg.summary()
```
Out[5]:

OLS Regression Results

| | | | |
|---|---|---|---|
| **Dep. Variable:** | Weight | **R-squared:** | 0.839 |
| **Model:** | OLS | **Adj. R-squared:** | 0.837 |
| **Method:** | Least Squares | **F-statistic:** | 815.3 |
| **Date:** | Tue, 09 Jun 2020 | **Prob (F-statistic):** | 4.75e-64 |
| **Time:** | 20:09:35 | **Log-Likelihood:** | -1015.1 |
| **No. Observations:** | 159 | **AIC:** | 2034. |
| **Df Residuals:** | 157 | **BIC:** | 2040. |
| **Df Model:** | 1 | | |
| **Covariance Type:** | nonrobust | | |

| | **coef** | **std err** | **t** | **P>\|t\|** | **[0.025** | **0.975]** |
|---|---|---|---|---|---|---|
| **Intercept** | -462.3751 | 32.243 | -14.340 | 0.000 | -526.061 | -398.690 |
| **Length1** | 32.7922 | 1.148 | 28.554 | 0.000 | 30.524 | 35.061 |

| | | | |
|---|---|---|---|
| **Omnibus:** | 9.385 | **Durbin-Watson:** | 0.369 |
| **Prob(Omnibus):** | 0.009 | **Jarque-Bera (JB):** | 9.768 |
| **Skew:** | -0.489 | **Prob(JB):** | 0.00757 |
| **Kurtosis:** | 3.721 | **Cond. No.** | 79.2 |

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

# Regression Equations

```
dependent ~ x1 + x2 + x3 + ...
```

We can force variables to be categorical:

```
dependent ~ x1 + x2 + C(x3) + ...
```

Here, we make `x3` categorical

# Regression Equations

```
dependent ~ x1 + x2 + x3 + ...
```

We can use arithmetic transformations:

```
dependent ~ x1 + I(x2**2) + x3 + ...
```

Here, we square `x2`

## When OLS Fails

OLS is typically considered an inappropriate model when you have a binary or discrete dependent variable (think "yes or no" questions)

In this case, you should use Logistic Regression instead. More details can be found in the class notes on Github.

# Implementing Logistic Regressions

```
formula = "y ~ x1 + x2 + ..."

reg = smf.logit(formula, data)

reg = reg.fit()

reg.summary()
```

# Lab Time!