Risk and Models

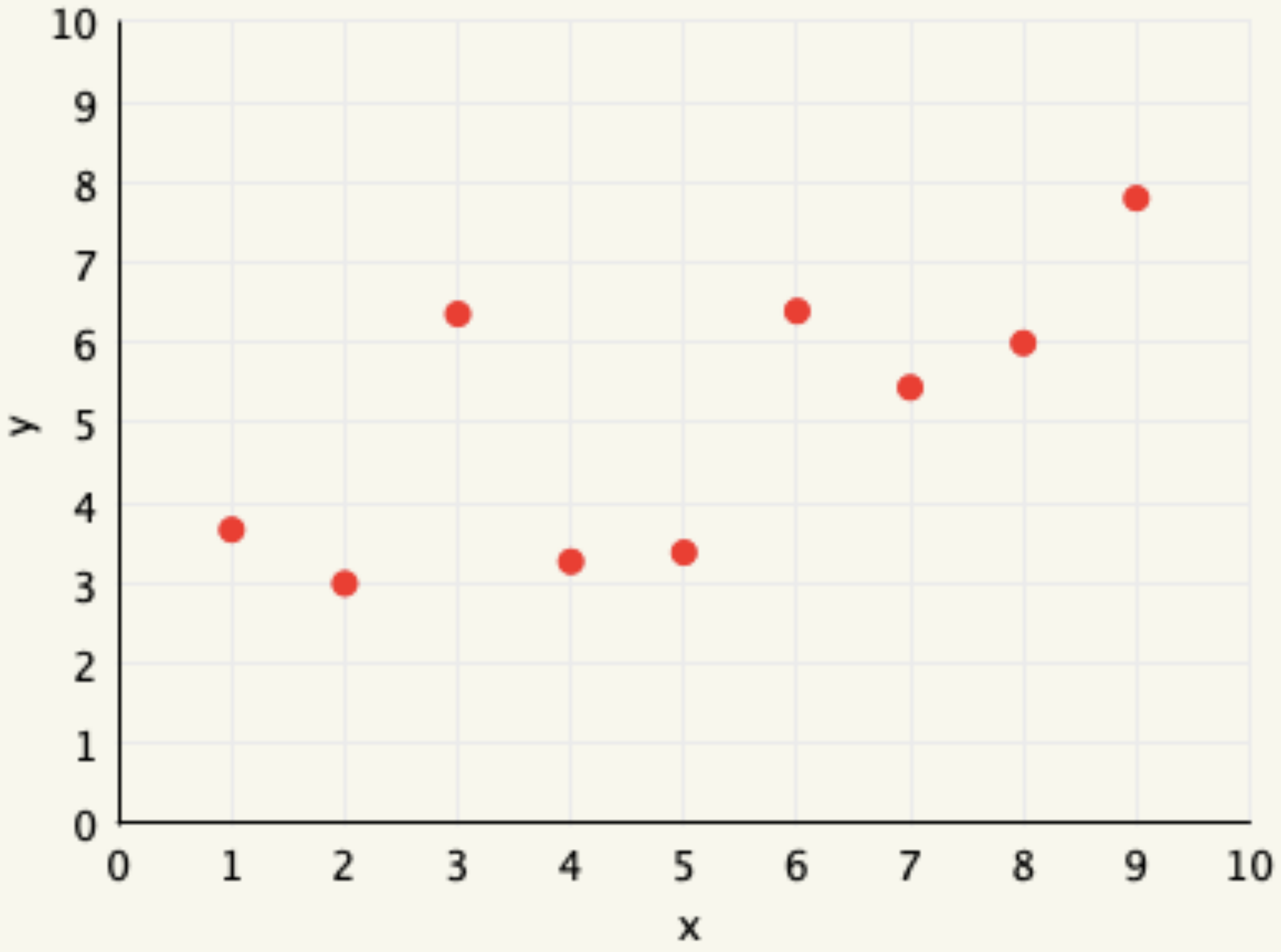
Univ.AI

Part of:

The Essence of Learning

All of AI is the estimation of functions.

Fitting a model to data



Here is some data. We would like to learn which function or model $f(x)$ **generated** this data..

What does it mean to say: *generated this data*? It does not mean that the data came directly from the function, but rather, that the function, corresponding to some physical, social, or other process, along with some noise gave rise to this data.

We do not know what this function f is. The best we can do is to find some other function g, the **fits this data best**, for some meaning of "best".

HYPOTHESIS SPACES

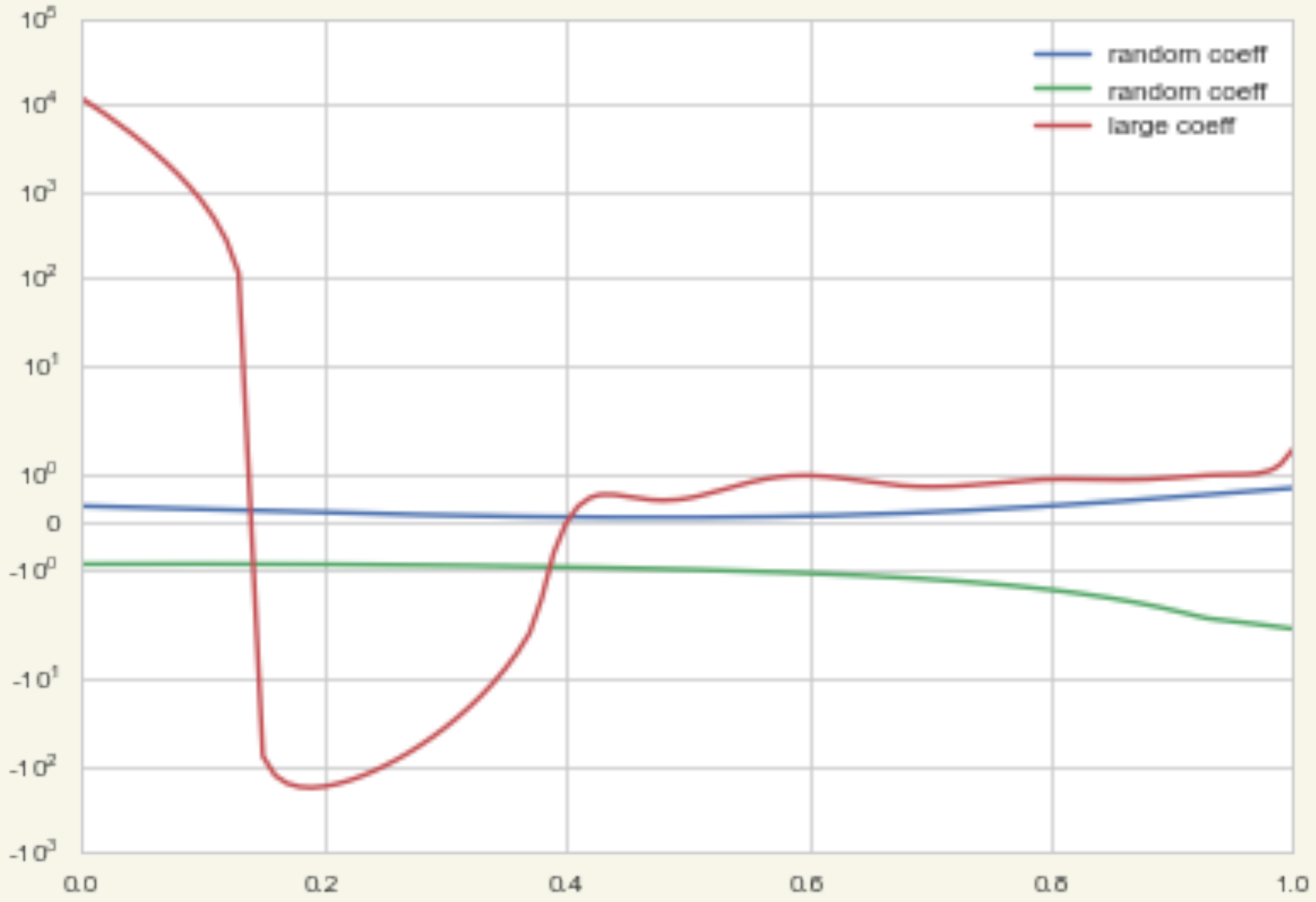
Where is this $g(x)$ chosen from?

It is chosen from a **Hypothesis Space**.

For example, a polynomial looks so:

$$h(x) = \theta_0 + \theta_1 x^1 + \theta_2 x^2 + \ldots + \theta_d x^d = \sum_{i=0}^{d} \theta_i x^i$$

All polynomials of a degree or complexity d constitute a hypothesis space.



For example:

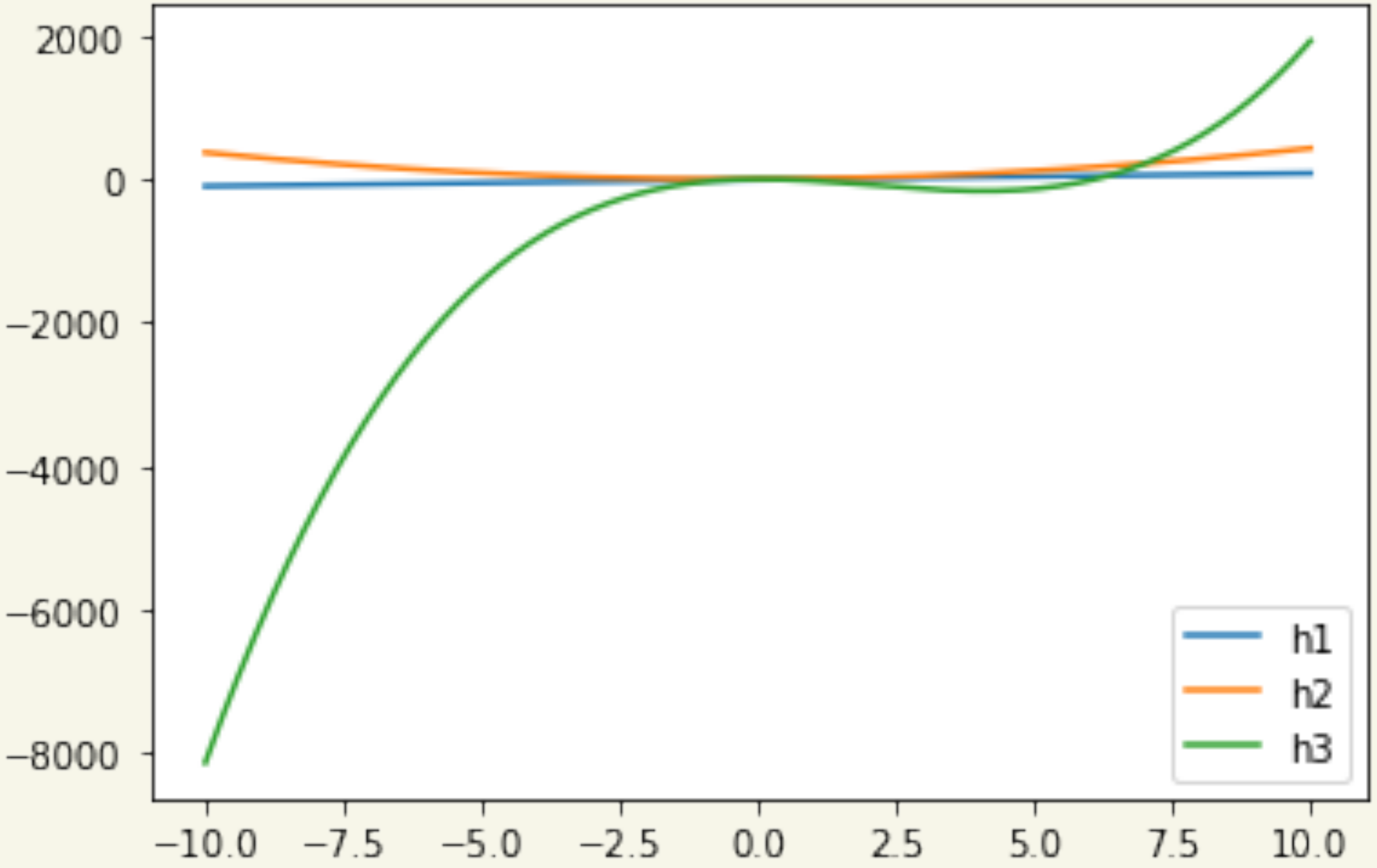
$$\mathcal{H}_1 : all\, h_1(x) = \theta_0 + \theta_1 x$$

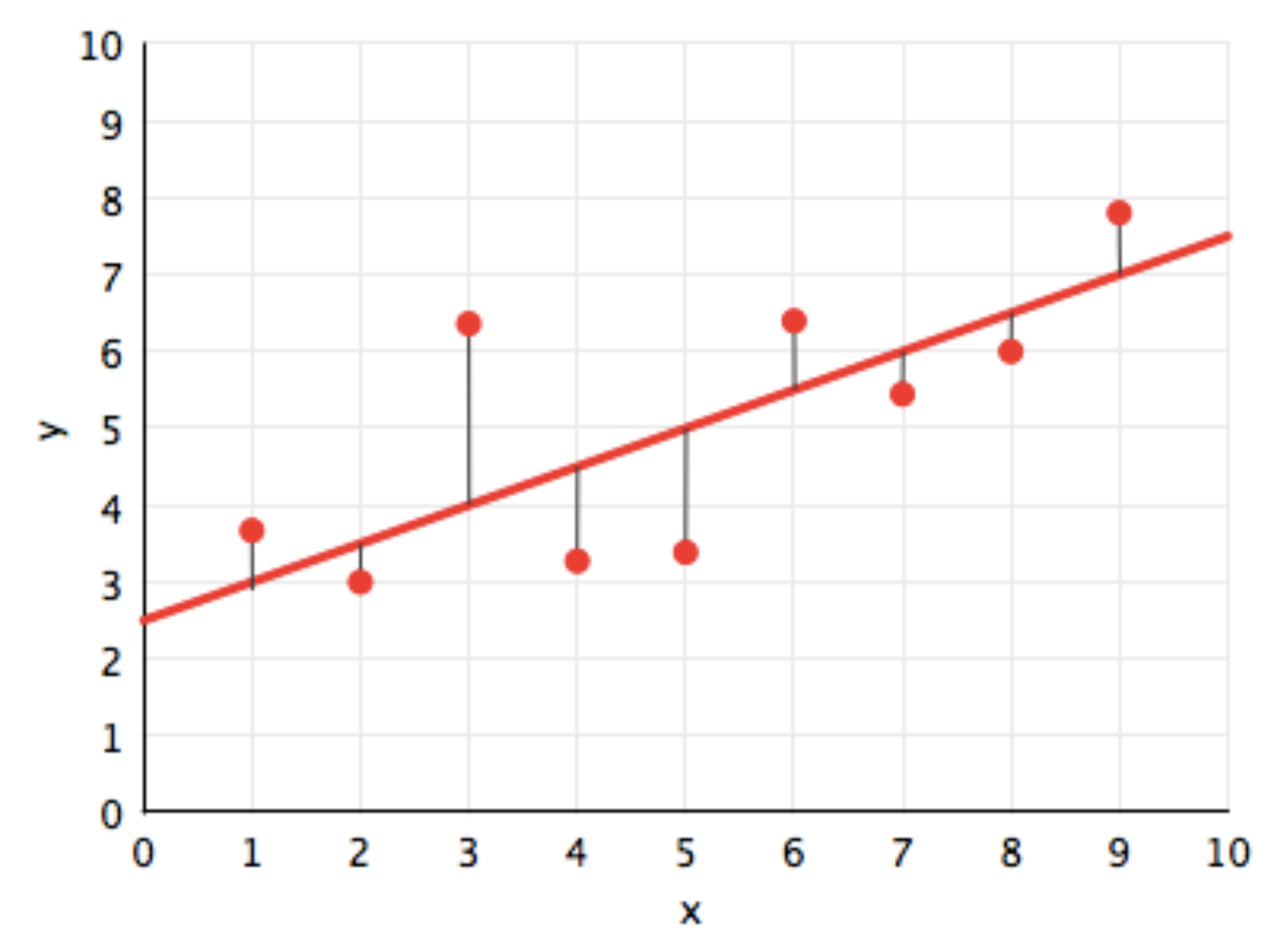
$$\mathcal{H}_{20} : all\, h_{20}(x) = \sum_{i=0}^{20} \theta_i x^i$$

Here are some examples:

```
h1 = lambda x: 9*x - 7
h2 = lambda x: 4*x**2 +3*x + 2
h3 = lambda x: 5*x**3 -31*x**2 + 3*x
```

Note the larger coefficient on $h_3(x)$ and the corresponding larger values in the plot.





What does it mean to FIT?

Minimize distance from the line?

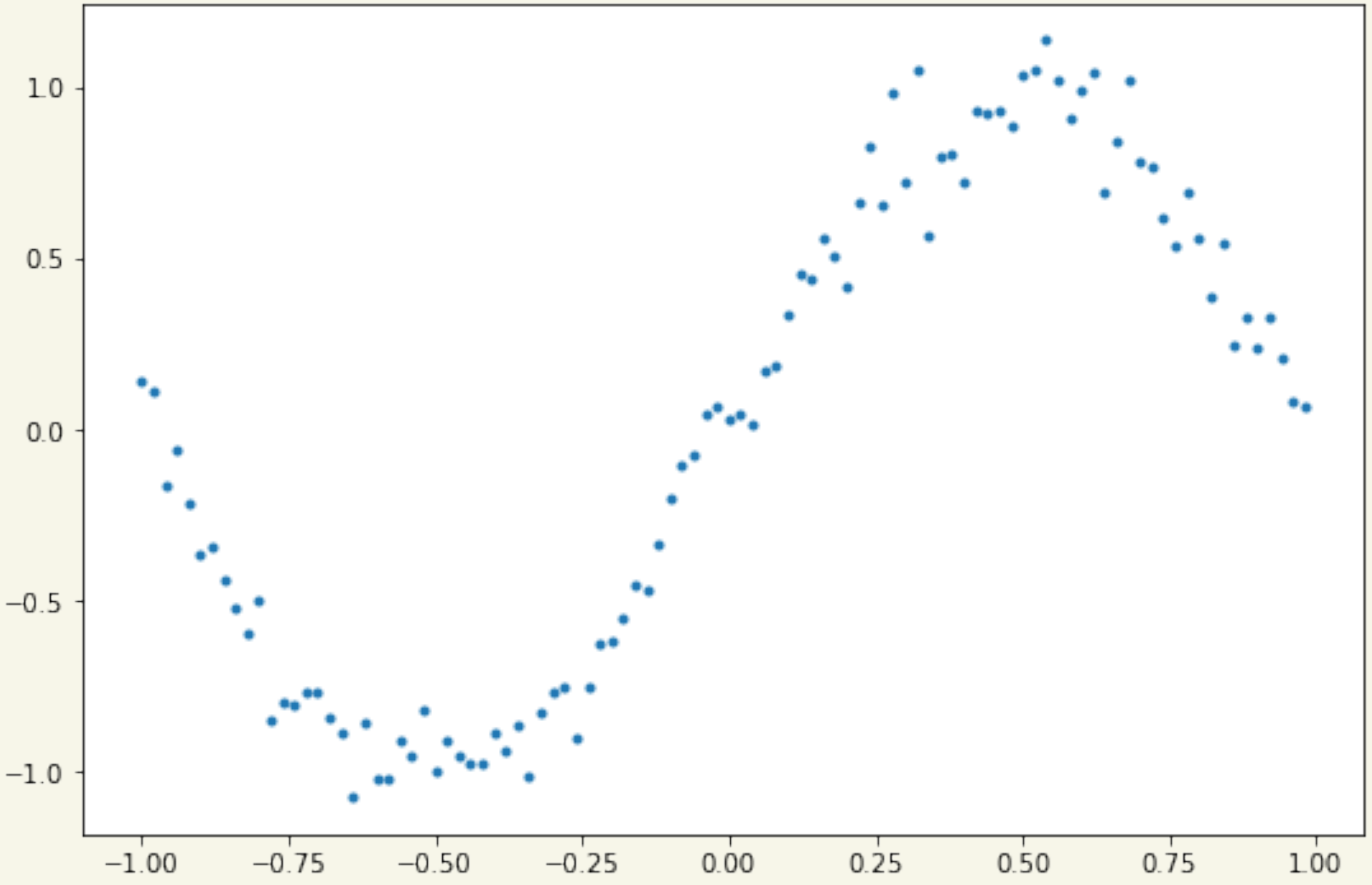
$$R_{\mathcal{D}}(h_1(x)) = \frac{1}{N} \sum_{y_i \in \mathcal{D}} (y_i - h_1(x_i))^2$$

Minimize squared distance from the line averaged over the points in our dataset.

This is called Empirical Risk Minimization. The squared distance is called the **mean squared error**.

$$g_1(x) = \arg \min_{h_1(x) \in \mathcal{H}_1} R_{\mathcal{D}}(h_1(x)).$$

Target or Generating Function



We'll say that the data points come from the generating model $f(x)$ when:

$$y = f(x) + \epsilon$$

where ϵ is **noise**.

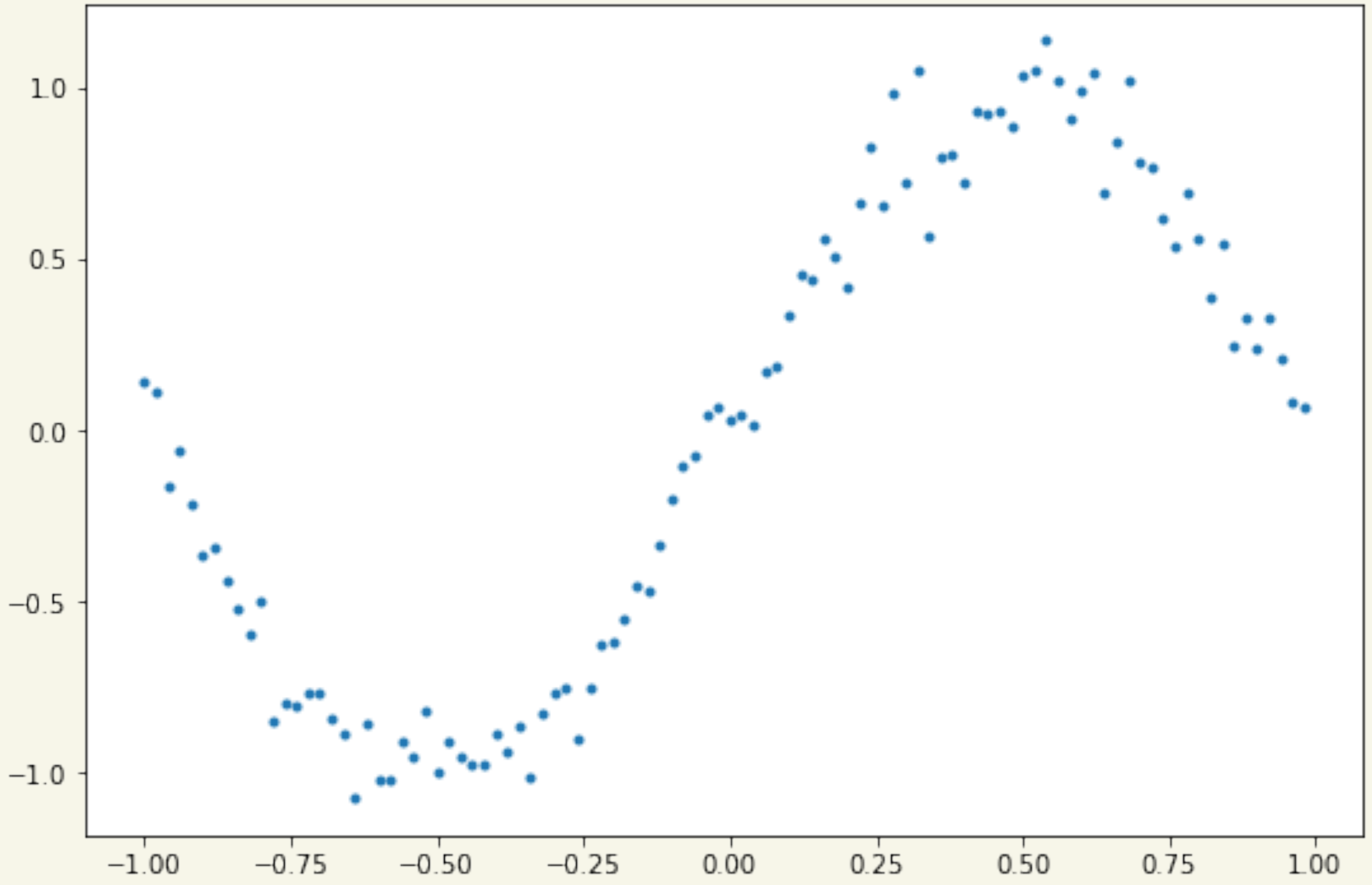
Approximation is when we assume that there is no noise, i.e. $\epsilon = 0$.

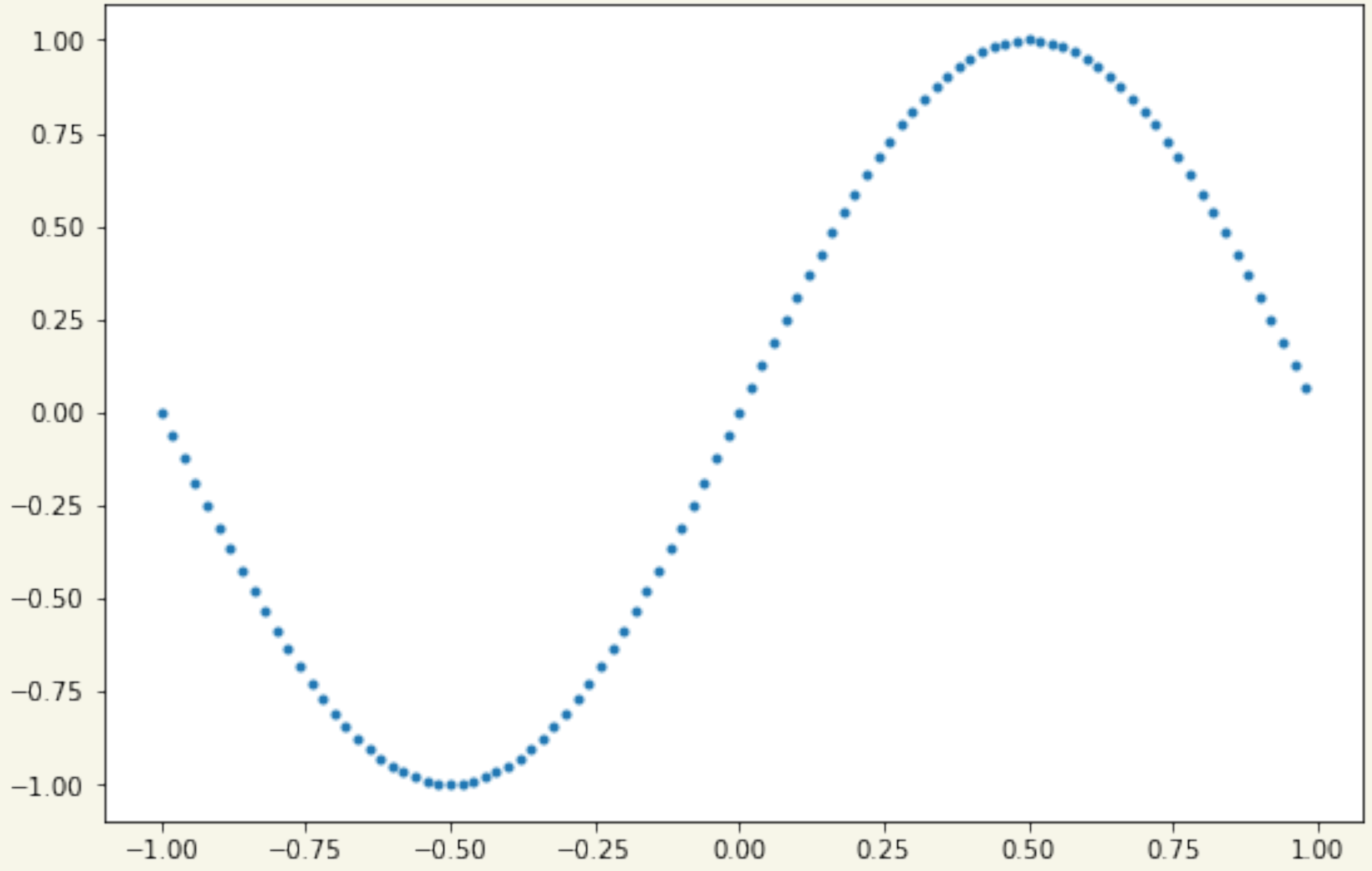
Then

$$y = f(x)$$

Univ.AI

Noise vs No Noise

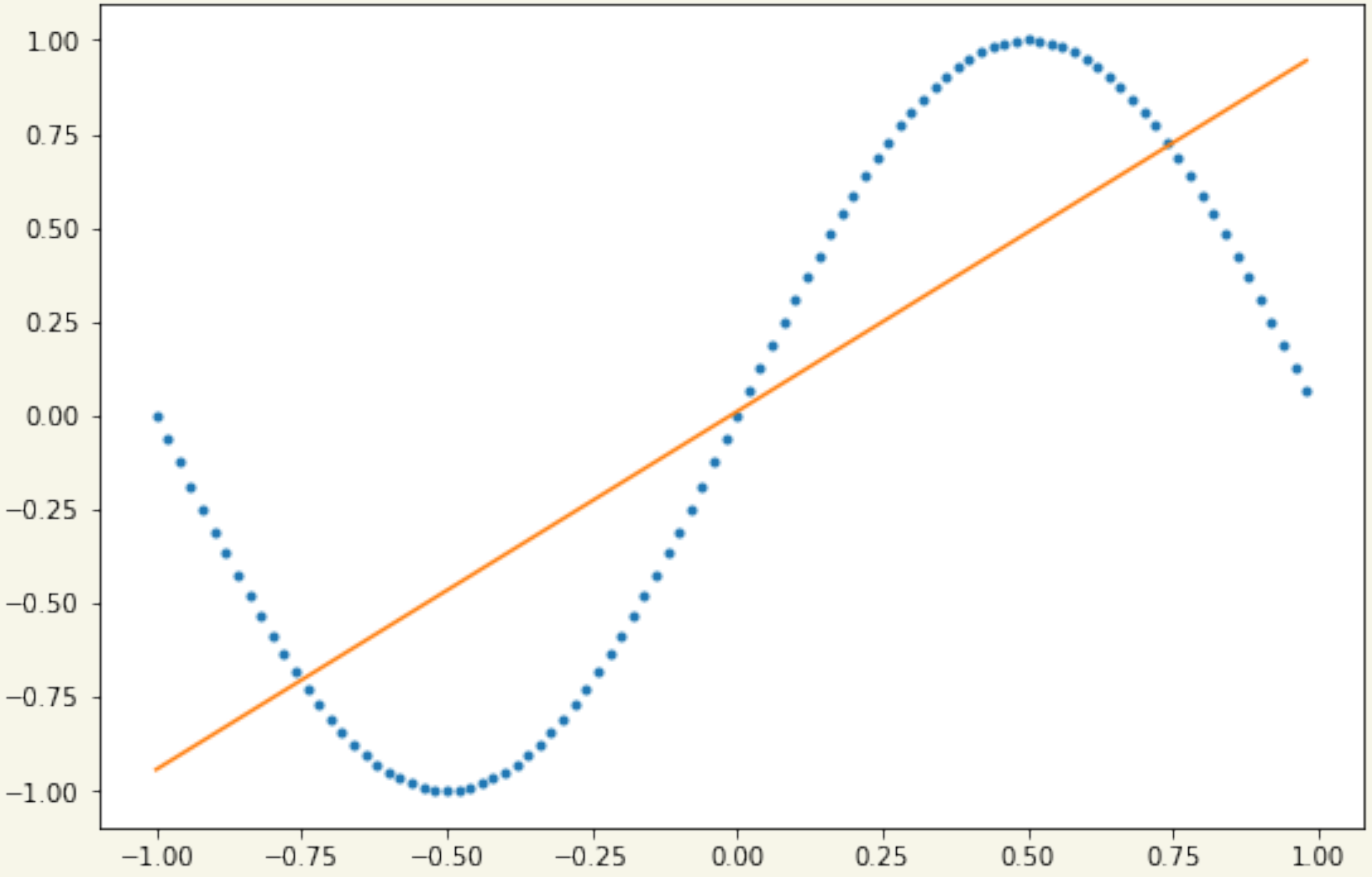




Fitting a hypothesis to the function on the right is called **approximation**.

Population And Samples

- We have a target function $f(x)$ that we do not know, with or without noise
- We'll call points $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$ a **population** of points that make up our **data**.
- You are usually not so lucky to get a population. E.g., to do a pre-election poll in a state, you might ask a **sample** of 1000 people for their choices, even though the voting population might be in the millions.
- Lets call the **sample** of data points from the population, $(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)$. We call this the **sample** or **training examples** $\mathcal{D}$.
- We are interested in using this sample (or the population if you have it) to estimate a function $g(x)$



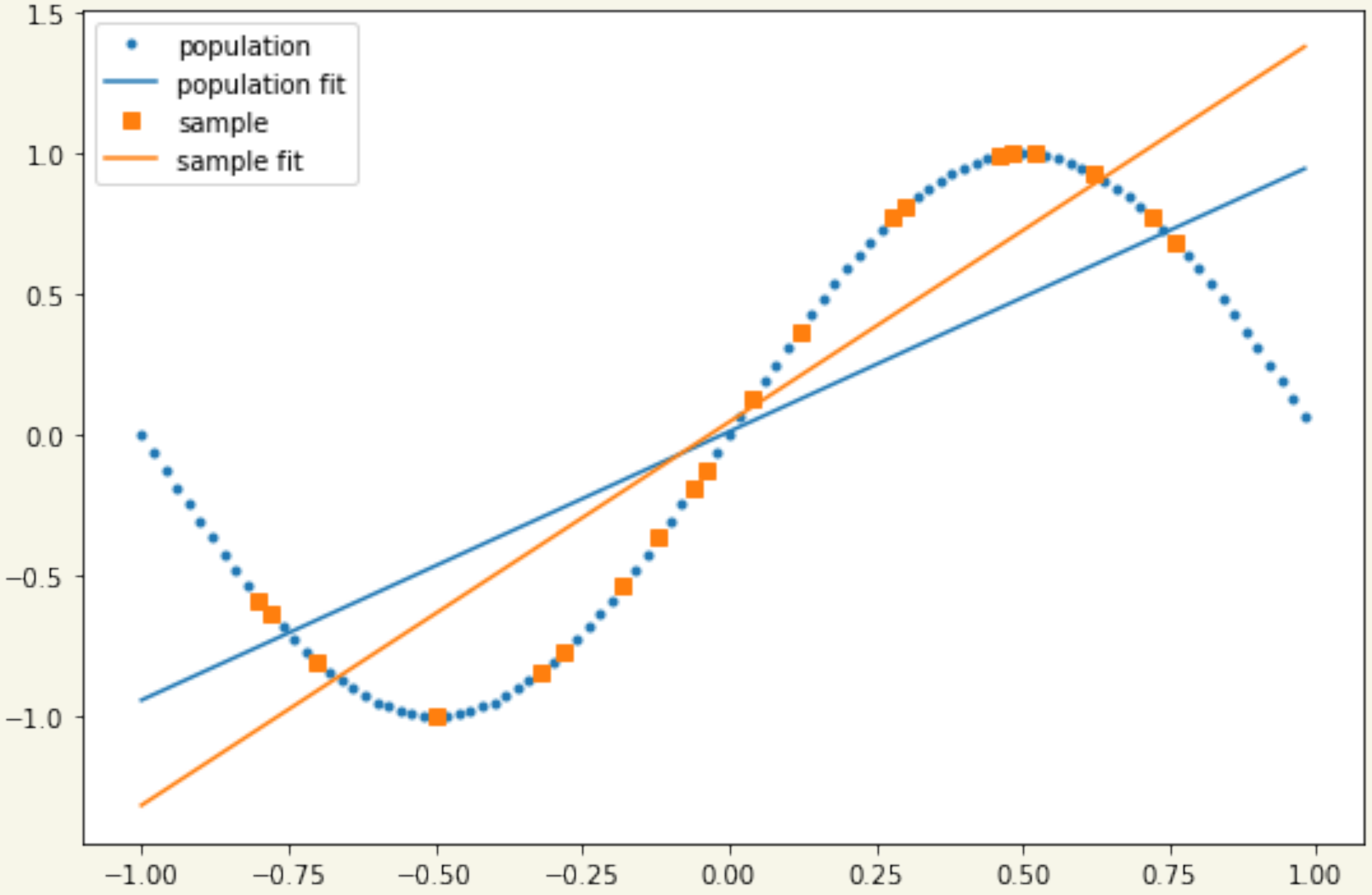
Approximating the sine curve on the population with a line

```
Xgrid = xgrid.reshape(-1,1)
learner = LinearRegression()
learner.fit(Xgrid, fgrid)
ggrid = learner.predict(Xgrid)
```

For any straight line, we compute the mean squared error (also called risk, cost). Then we compare all the lines and choose the one with the lowerst mean squared error.

Univ.AI

Using a sample instead

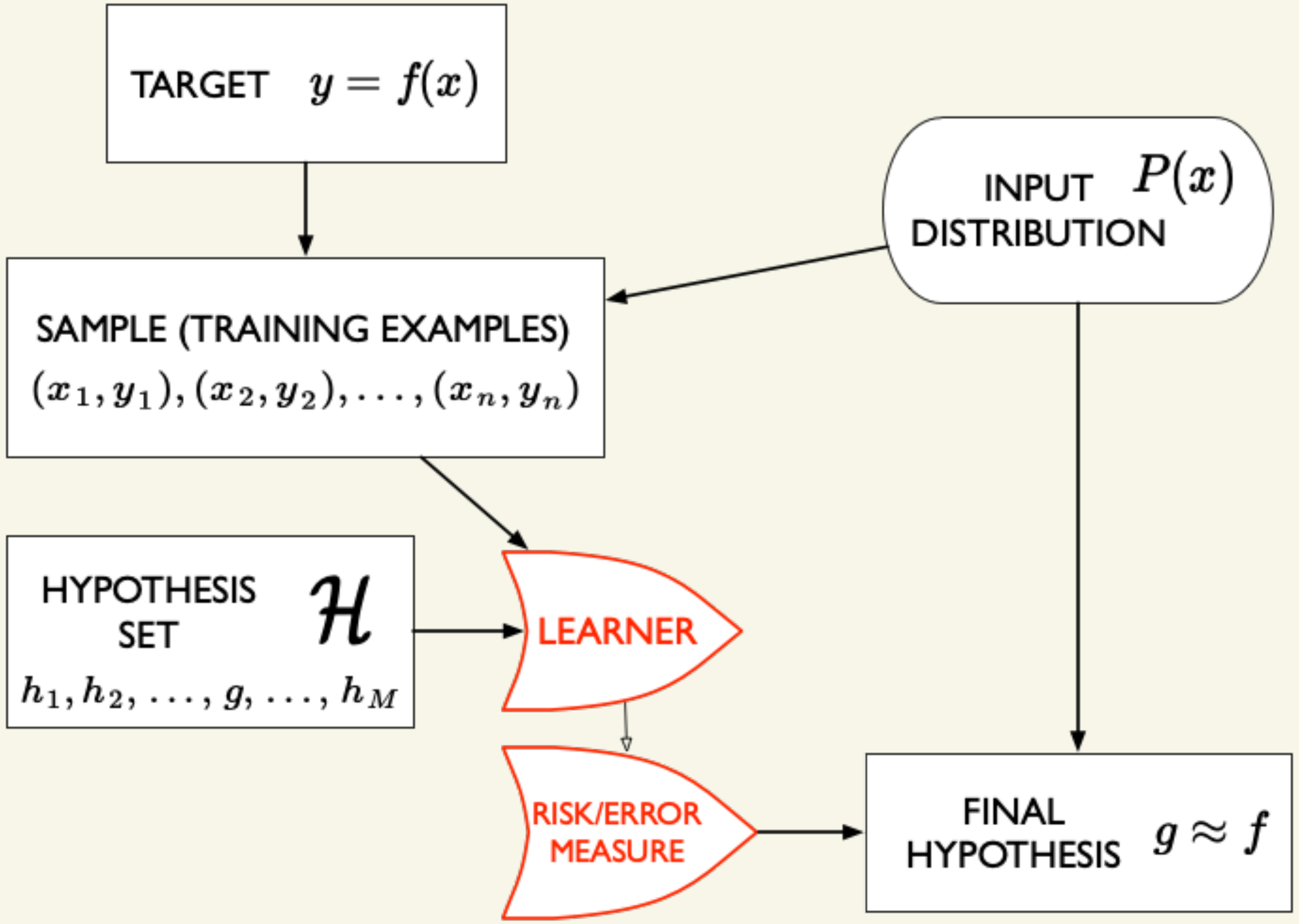


Say we have a sample of 20 points. We minimize the mean squared error over all possible lines (by minimizing the error with respect to the slope and intercept).

This approximation is very bad to start with on the population, but note that making the approximation on a sample does give is a different answer, but the general direction of the line is not so hugely different.

Univ.AI

Diagram of the approximation fitting process

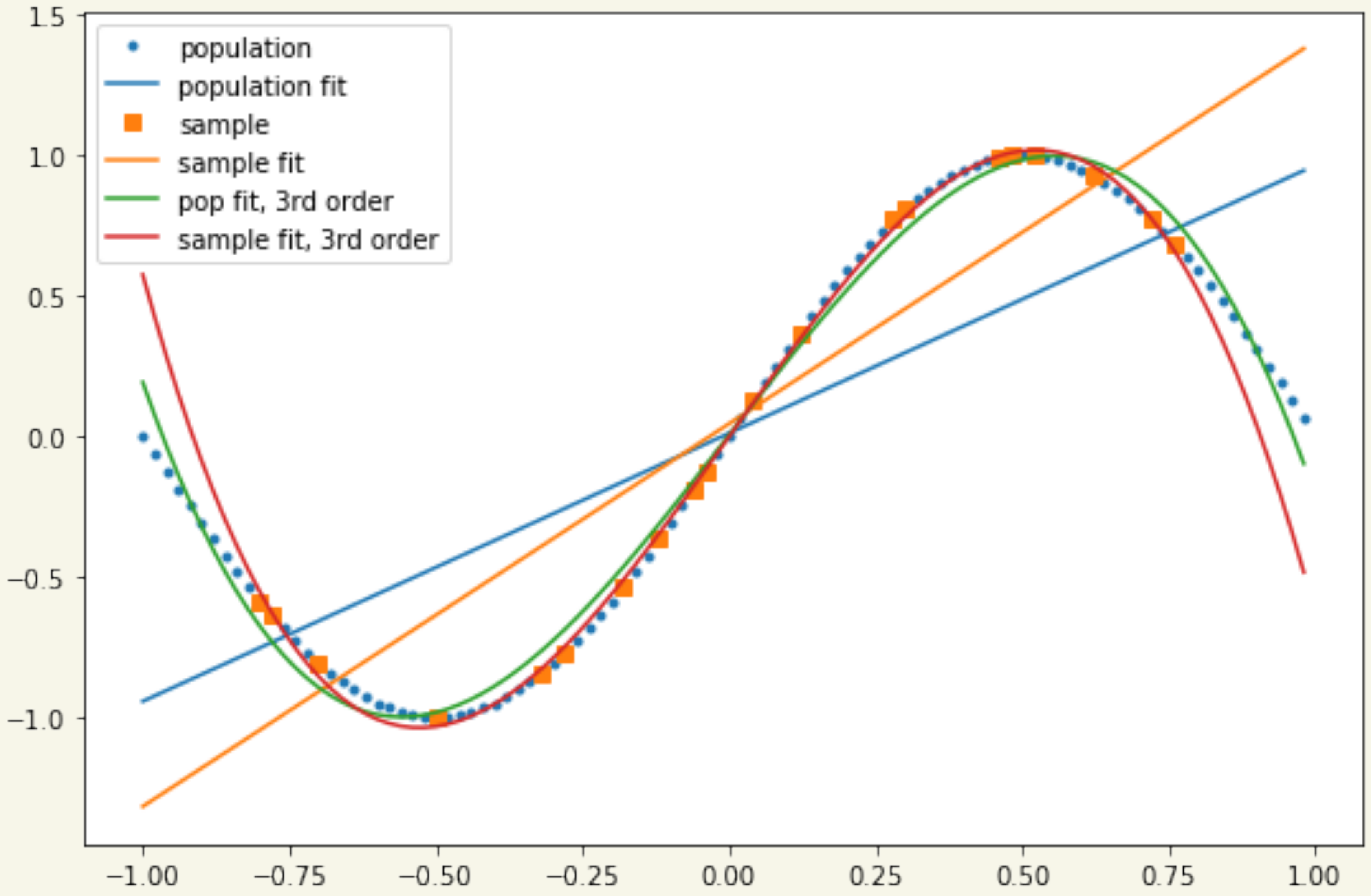

TARGET $y = f(x)$
INPUT DISTRIBUTION $P(x)$
SAMPLE (TRAINING EXAMPLES)
$(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$
HYPOTHESIS SET $\mathcal{H}$
$h_1, h_2, \ldots, g, \ldots, h_M$
LEARNER
RISK/ERROR MEASURE
FINAL HYPOTHESIS $g \approx f$

Third order Polynomial on both the population and sample

Because cubics can bend and become negative, this polynomial is far more able to approximate the sine curve

Thus the overall mean squared error is lower.

The population and sample cubics are quite close, so this sample does a good job of surfacing population behavior.


● population
population fit
■ sample
sample fit
pop fit, 3rd order
sample fit, 3rd order