COMS20011 Regression

Laurence Aitchison

March 2022

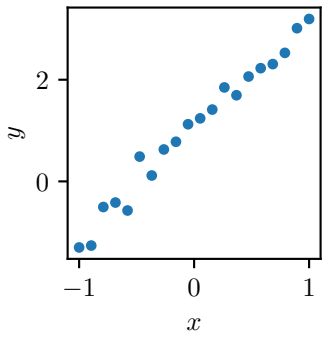
1 Motivation: Prediction

Loads of problems in business and data science fundamentally involve prediction.

For instance, you might be trying to work out how much product to produce next month. You can make a plot, where the time (in months) is x, and the amount you have sold is y. Your goal is then to predict y for the next month (x).

2 Regression as curve fitting

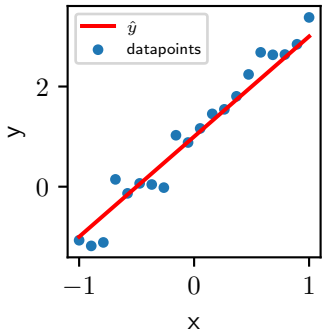
Ultimately, regression is just curve-fitting. And the simplest form of regression is fitting a straight line to a dataset involving inputs, x_i and targets, y_i.



You can draw a line by hand, or guess a good function in simple cases such as this. For instance, looking at the plot, we can see that the intercept (where the line crosses $x = 0$) is around 1 and the slope is around 2 (it goes up roughly 2 on the y-axis if we move right 1 on the x-axis). Thus, we could guess:

$$\hat{y}(x) = 2x + 1 \tag{1}$$

where $\hat{y}(x_i)$ is our prediction of the value of y_i.



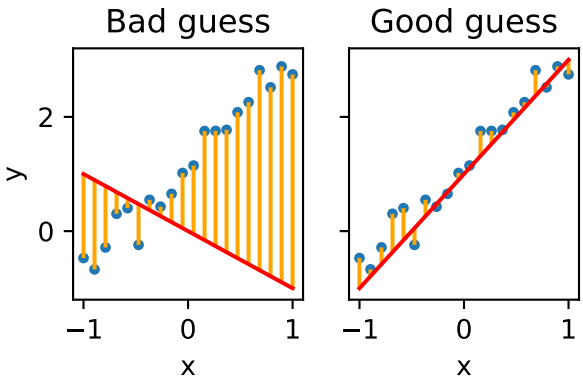
But there are many situations where this simple approach isn't going to work:

- Too many datapoints to visualise easily.
- Too complicated function to guess.
- Too many input "features" (our guesses might depend on size, shape, color, weight of a product).

In these more complicated contexts, we can't just draw something by eye. We're going to need to fit the curves using maths. We're going to start in simple settings where we confirm the intuition works out, but the maths we develop is going to extend naturally to these more complex settings.

3 Quantifying good and bad predictions using distances/losses

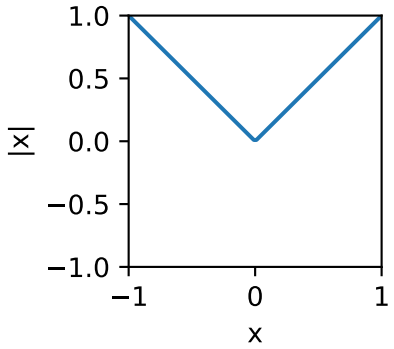
To get a mathematical procedure for fitting good predictions, we need to know what makes a good prediction and what makes a bad prediction. Intuitively, a good prediction lies close to the true value, and a bad prediction lies far from the true value.



But to choose better predictors, we need to formally nail down this notion of distance. Fundamentally, there are a huge number of sensible things we could choose, corresponding to different mathematical formalisations of "distance". Any given notion of distance will give rise to a "loss", $\mathcal{L}$. The goal is to find a function $\hat{y}(x_i)$ which is "close" to the true values, y_i, and thus has a small loss. Generally speaking, the loss is a sum of the distance for each datapoint, because we want the prediction for every datapoint to be close to the true value. Perhaps the most obvious is to use the sum of distances between y_i and $\hat{y}_i$,

$$\mathcal{L} = \sum_i \left| \hat{y}(x_i) - y_i \right|, \tag{2}$$

where,



This has lots of nice properties, and is a good option in many cases. However, this loss makes the maths really hard (actually, we can't do anything and need to resort to numerical methods). To make the maths easier (or at least possible), we instead use the sum of *squared* distances,

$$\mathcal{L} = \sum_i \left(\hat{y}(x_i) - y_i \right)^2, \tag{3}$$

To check that this loss does something sensible, we computed it for the Good Guess/Bad Guess Figure above. The obviously badly-fitting line (left) has a loss of 34.1 and the obviously better-fitting line (right) has a loss of 4.8. So the better-fitting line indeed has a lower loss.

4 Different choices of functions

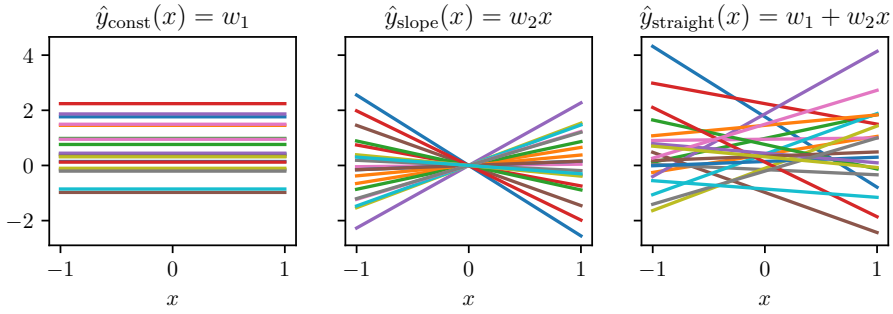
We now have a "loss" the quantifies how good any given prediction-function, $\hat{y}(x_i)$, is. In the previous section, we used the loss to show that one particular choice of prediction-function was better than another choice of prediction-function. So we might want to try loads of different choices of prediction-function, and find the one with the lowest loss. But that doesn't scale:

- Where are these possible prediction-functions going to come from?
- What if we have *loads* of possible prediction-functions?

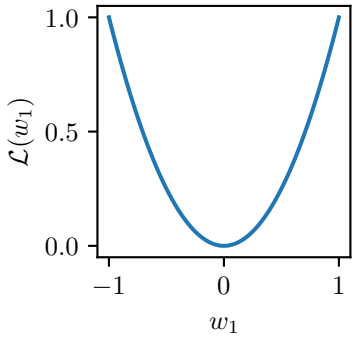
To solve these problems, we're going to write our family of possible prediction functions using parameters. For instance we could consider three predictors: the constant predictor, the "slope" predictor, and the straight-line predictor,

$$\hat{y}_{\text{const}}(x_i) = w_1 \qquad \hat{y}_{\text{slope}}(x_i) = w_2 x_i \qquad \hat{y}_{\text{straight}}(x_i) = w_1 + w_2 x_i. \tag{4}$$

We can choose different values for the parameters, w_1 and w_2. Possible functions in that family are,



There are infinitely many functions in these families, because there are infinitely many possible choices of w_1 and w_2. That means we can't try all possible values of w_1 and w_2. What can we do instead? It turns out we can find the lowest-loss function analytically, using calculus. Remember that the minimum of a function is often where the function has slope 0. For instance, for the constant predictor we might have,



4.1 Finding the best constant predictor

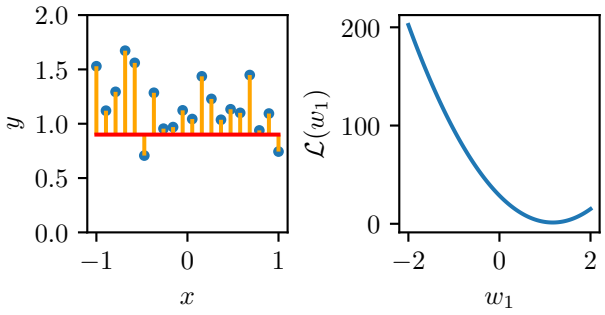
The constant predictor always predicts that y is w_1,

$$\hat{y}_{\text{const}}(x_i) = w_1 \tag{5}$$

That means the loss (sum of squared distances between the prediction and actual value) becomes,

$$\mathcal{L}_{\text{const}}(w_1) = \sum_i \left(\hat{y}_{\text{const}}(x_i) - y_i\right)^2 = \sum_i \left(w_1 - y_i\right)^2 \tag{6}$$

Our goal is to to find the prediction-function with the lowest loss. We can do this graphically,



and we can see that the optimal value of w_1 is about 1.1, which is about in the middle of the data.

But can we do a better job? Look at the smooth curve of loss against w_1: the minimum of the curve is the location where the gradient is zero. So, we can find that minimum by treating $\mathcal{L}_{\text{const}}(w_1)$, as a function of the parameters, and differentiating to find the value of w_1 for which the gradient is zero. We start by differentating the loss,

$$0 = \frac{\partial}{\partial w_1}\left[\mathcal{L}_{\text{const}}(w_1)\right] \tag{7}$$

$$0 = \frac{\partial}{\partial w_1}\left[\sum_{i=1}^{N}\left(y_i - w_1\right)^2\right]. \tag{8}$$

Put the gradient inside the sum,

$$0 = \sum_{i=1}^{N}\frac{\partial}{\partial w_1}\left(y_i - w_1\right)^2. \tag{9}$$

Expand the brackets,

$$0 = \sum_{i=1}^{N}\frac{\partial}{\partial w_1}\left[y_i^2 - 2y_i w_1 + w_1^2\right]. \tag{10}$$

Take the gradient,

$$0 = \sum_{i=1}^{N}\left[-2y_i + 2w_1^2\right]. \tag{11}$$

Divide by 2,

$$0 = \sum_{i=1}^{N} (w_1 - y_i) \tag{12}$$

Expand the bracket,

$$0 = \sum_{i=1}^{N} w_1 - \sum_{i=1}^{N} y_i \tag{13}$$

The first term, $\sum_{i=1}^{N} w_1$, is just $w_1 + w_1 + \cdots + w_1$ N times, so it equals Nw_1,

$$0 = Nw_1 - \sum_{i=1}^{N} y_i \tag{14}$$

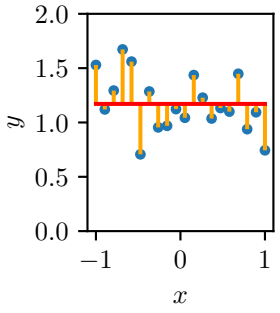
Add $\sum_{i=1}^{N} y_i$ to both sides,

$$Nw_1 = \sum_{i=1}^{N} y_i \tag{15}$$

Finally, solve for w_1, by dividing both sides by N,

$$w_1 = \frac{1}{N} \sum_{i=1}^{N} y_i. \tag{16}$$

Is what we've ended up with sensible? Yes! Remember that this was a constant predictor, $\hat{y}_{\text{const}}(x_i) = w_1$. What might be a sensible choice of constant predictor, $\hat{y}_{\text{const}}(x_i) = w_1$? Well, we might expect w_1 to be in the middle of the datapoints, y_i. Averages formalise this notion of the "middle" of the datapoints, so we might expect w_1 to be some kind of average of the data, y_i, maybe the mean or median. What have we come out with? The mean!



And this optimal / best fitting predictor looks pretty good!

4.2 Finding the best "slope" predictor

Obviously, that was alot of work to just end up back at the mean. But we confirmed that minimizing the sum-squared error loss does something sensible. And we're going to be able to extend the sum-squared error loss idea to more complex cases. The next step is to consider a predictor that only has one parameter, w_2, but depends on the data,

$$\hat{y}_{\text{slope}}(x_i) = w_2 x_i \tag{17}$$

Again, the loss can be written as a function of the parameter, w_2.

$$\mathcal{L}_{\text{slope}}(w_2) = \sum_i \left(\hat{y}_{\text{slope}}(x_i) - y_i\right)^2 = \sum_i \left(w_2 x_i - y_i\right)^2 . \tag{18}$$

Our goal is to to find the prediction-function with the lowest loss. To do that, we need to find the place where the derivative wrt w_2 is zero.

$$0 = \frac{\partial}{\partial w_2}\left[\mathcal{L}_{\text{slope}}(w_2)\right] \tag{19}$$

$$0 = \frac{\partial}{\partial w_2}\left[\sum_{i=1}^{N}\left(w_2 x_i - y_i\right)^2\right] \tag{20}$$

expand the brackets

$$0 = \frac{\partial}{\partial w_2}\left[\sum_{i=1}^{N}\left(y_i^2 - 2w_2 x_i y_i + w_2^2 x_i^2\right)\right] \tag{21}$$

swap the derivative and sum,

$$0 = \sum_{i=1}^{N}\frac{\partial}{\partial w_2}\left[y_i^2 - 2w_2 x_i y_i + w_2^2 x_i^2\right] \tag{22}$$

$$0 = \sum_{i=1}^{N}\left(\frac{\partial}{\partial w_2}[y_i^2] - \frac{\partial}{\partial w_2}[2w_2 x_i y_i] + \frac{\partial}{\partial w_2}[w_2^2 x_i^2]\right) \tag{23}$$

use $\frac{\partial x^p}{\partial x} = p x^{p-1}$ or $\frac{\partial w_1^p}{\partial w_1} = p w_1^{p-1}$,

$$0 = \sum_{i=1}^{N}\left(0 - 2x_i y_i + 2w_2 x_i^2\right) \tag{24}$$

Divide both sides by 2,

$$0 = \sum_{i=1}^{N}\left(w_2 x_i^2 - y_i x_i\right). \tag{25}$$

Now we sum over each term separately,

$$0 = \sum_{i=1}^{N} w_2 x_i^2 - \sum_{i=1}^{N} y_i x_i. \tag{26}$$

Notice that w_2 doesn't depend on i, so it can be brought outside the sum,

$$0 = w_2 \sum_{i=1}^{N} x_i^2 - \sum_{i=1}^{N} y_i x_i. \tag{27}$$

Now rearrange (add $\sum_{i=1}^{N} y_i x_i$ to both sides, and divide by $\sum_{i=1}^{N} x_i^2$,)

$$w_2 = \frac{\sum_{i=1}^{N} y_i x_i}{\sum_{i=1}^{N} x_i^2}. \tag{28}$$

Does this make sense? Yes! Remember that w_2 is the slope. The slope is bigger if for positive x_i, we have big positive y_i, and for negative x_i we have big negative y_i. Indeed, the numerator in the final fraction is big in that case too!

4.3 Straight line fitting

Above, we discussed the basic idea, but we only ever tried to fit a single parameter. What happens when we fit a full straight line,

$$\hat{y}_{\text{straight}}(x_i) = w_1 + w_2 x_i \tag{29}$$

with an intercept, w_1 and slope, w_2 parameter? Conceptually, it is the same idea we've seen previously: the loss is a function of w_1 and w_2,

$$\mathcal{L}_{\text{straight}}(w_1, w_2) = \sum_i \left(\hat{y}_{\text{slope}}(x_i) - y_i\right)^2 = \sum_i \left(w_1 + w_2 x_i - y_i\right)^2 \tag{30}$$

We find the value of w_1, w_2 such that both derivatives are simultaneously zero,

$$0 = \frac{\partial}{\partial w_1} \mathcal{L}_{\text{slope}}(w_1, w_2) \qquad\qquad 0 = \frac{\partial}{\partial w_2} \mathcal{L}_{\text{slope}}(w_1, w_2) \tag{31}$$

This is harder but still possible, so we're not going to give the proof here.

Instead, we're going to write down the solution for w_2, then give the implied value of w_1. Written in full, the optimal value of w_2,

$$w_2 = \frac{\frac{1}{N}\sum_{i=1}^{N} x_i y_i - (\frac{1}{N}\sum_{i=1}^{N} x_i)(\frac{1}{N}\sum_{i=1}^{N} y_i)}{\frac{1}{N}\sum_i x_i^2 - (\frac{1}{N}\sum_i x_i)^2} \tag{32}$$

As that's a bit painful, we can write it more succinctly as,

$$w_2 = \frac{\overline{xy} - \overline{x}\,\overline{y}}{\overline{x^2} - (\overline{x})^2} \tag{33}$$

where,

$$\overline{xy} = \tfrac{1}{N}\sum_{i=1}^{N} x_i y_i \tag{34}$$

$$\overline{x} = \tfrac{1}{N}\sum_{i=1}^{N} x_i \tag{35}$$

$$\overline{y} = \tfrac{1}{N}\sum_{i=1}^{N} y_i \tag{36}$$

$$\overline{x^2} = \tfrac{1}{N}\sum_{i=1}^{N} x_i^2 \tag{37}$$

$$(\overline{x})^2 = (\tfrac{1}{N}\sum_{i=1}^{N} x_i)^2 \tag{38}$$

FYI: technically the "overbar" notation for sample averages and $\mathbb{E}[X]$ for "true" averages (e.g. when we define X to be Gaussian with zero mean). I (Laurence) will try to make this distinction, but it won't always happen!

Now that we have w_2, we can solve for w_1.

$$0 = \frac{\partial}{\partial w_1}\mathcal{L}_{\text{slope}}(w_1, w_2) \tag{39}$$

$$0 = \frac{\partial}{\partial w_1}\sum_{i=1}^{N}\left(w_1 + w_2 x_i - y_i\right)^2 \tag{40}$$

Group all the terms that don't depend on w_1,

$$0 = \frac{\partial}{\partial w_1}\sum_{i=1}^{N}\left(w_1 + (w_2 x_i - y_i)\right)^2 \tag{41}$$

Expand the bracket (with all the terms that don't depend on w_1 in $\cdots$),

$$0 = \frac{\partial}{\partial w_1}\sum_{i=1}^{N}\left(w_1^2 + 2w_1(w_2 x_i - y_i) + (w_2 x_i - y_i)^2\right) \tag{42}$$

Take the derivative,

$$0 = \sum_{i=1}^{N}\left[2w_1 + 2(w_2 x_i - y_i)\right] \tag{43}$$

Divide by two and put the sum inside the bracket,

$$0 = \sum_{i=1}^{N} w_1 - \sum_{i=1}^{N}(y_i - w_2 x_i) \tag{44}$$

As w_1 does not vary with i,

$$0 = Nw_1 - \sum_{i=1}^{N}(y_i - w_2 x_i). \tag{45}$$

Adding $\sum_{i=1}^{N}(y_i - w_2 x_i)$ to both sides,

$$Nw_1 = \sum_{i=1}^{N}(y_i - w_2 x_i). \tag{46}$$

And dividing both sides by N,

$$w_1 = \tfrac{1}{N}\sum_{i=1}^{N}(y_i - w_2 x_i). \tag{47}$$

This makes sense: the intercept, w_1, is the average error between y_i and $w_2 x_i$. For instance, if y_i is generally much bigger than $w_2 x_i$, then the intercept is positive.

Overall, we have,

$$w_2 = \frac{\overline{xy} - \overline{x}\,\overline{y}}{\overline{x^2} - (\overline{x})^2} \tag{48a}$$

$$w_1 = \tfrac{1}{N}\sum_{i=1}^{N}(y_i - w_2 x_i). \tag{48b}$$

5 Multivariable regression

So far, we've always considered single-variable inputs, x, so that we can fit a curve on a 2D plot. But in the real-world, we might want to predict an outcome from many features. For instance, we might want to predict a product's sales from many different features (length, x_1, height, x_2, weight, x_3 etc.) How can we do predictions with multiple input features? To start with, we need to talk about how we're going to represent the data. We say the data has D features (e.g. for length, height, weight, we would have $D = 3$ features) and N datapoints. In that case, we could represent all the input features as an big $N \times D$ matrix, $\mathbf{X}$. To give an example for $D = 3$.

$$\mathbf{X} = \begin{pmatrix} X_{11} & X_{12} & X_{13} \\ X_{21} & X_{22} & X_{23} \\ \vdots & \vdots & \vdots \\ X_{N1} & X_{N2} & X_{N3} \end{pmatrix} \qquad \mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{pmatrix} \tag{49}$$

For both $\mathbf{X}$ and $\mathbf{y}$, each row corresponds to a datapoint. Thus, for each datapoint, we have an output y_i and a row-vector of input features,

$$\mathbf{x}_i^T = \begin{pmatrix} X_{i1} & X_{i2} & X_{i3} \end{pmatrix} \tag{50}$$

Now, the prediction becomes a function of the whole vector of input points,

$$\hat{y}(\mathbf{x}_i) = \mathbf{x}_i^T \mathbf{w} = \sum_j X_{ij} w_j \tag{51}$$

The optimal weights are,

$$\mathbf{w}^* = \left(\mathbf{X}^T \mathbf{X}\right)^{-1} \mathbf{X}^T \mathbf{y} \tag{52}$$

Lets look a bit closer at the sizes,

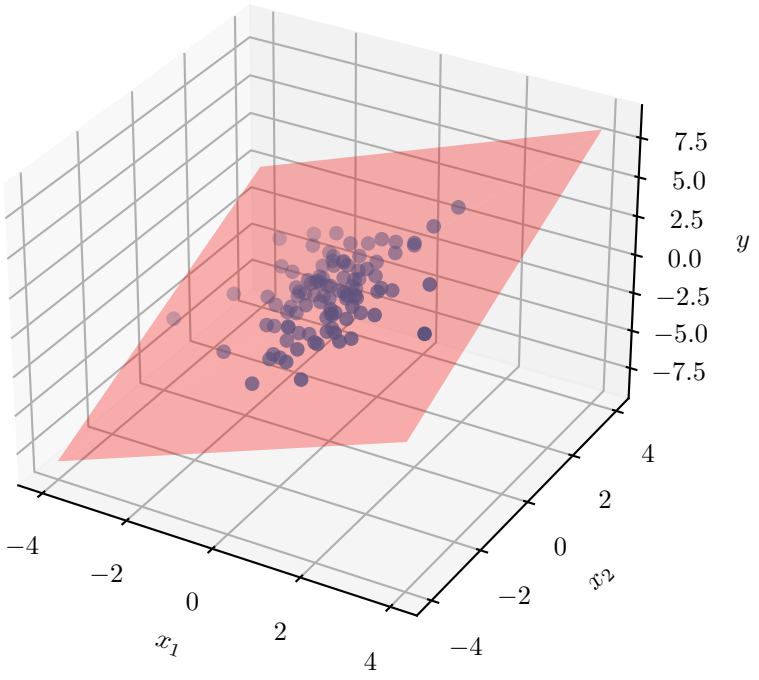
$$\underbrace{\mathbf{w}^*}_{D\times 1} = \big(\underbrace{\mathbf{X}^T}_{D\times N}\underbrace{\mathbf{X}}_{N\times D}\big)^{-1}\underbrace{\mathbf{X}^T}_{D\times N}\underbrace{\mathbf{y}}_{N\times 1} \tag{53}$$

Or, if we look at $\left(\mathbf{X}^T \mathbf{X}\right)^{-1}$ and $\mathbf{X}^T \mathbf{y}$ separately,

$$\underbrace{\mathbf{w}^*}_{D\times 1} = \underbrace{\left(\mathbf{X}^T \mathbf{X}\right)^{-1}}_{D\times D}\underbrace{\mathbf{X}^T \mathbf{y}}_{D\times 1}. \tag{54}$$

When doing this by hand, you'd usually compute $\left(\mathbf{X}^T \mathbf{X}\right)^{-1}$ and $\mathbf{X}^T \mathbf{y}$ separately, as the number of dimensions, D, is usually going to be smaller than the number of datapoints.

The result will look something like,

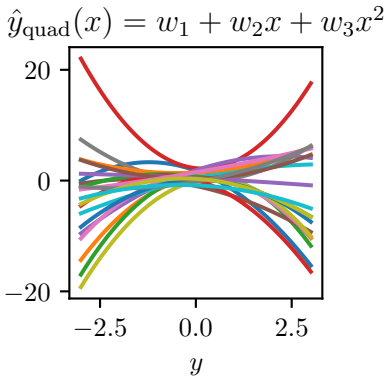


6 Regression with nonlinear features

What if we want to fit a model of the form,

$$\hat{y}_{\text{quad}}(x) = w_1 + w_2 x + w_3 x^2. \tag{55}$$

If we plot a bunch of these functions, with random w_1, w_2 and w_3, we get,



This is different from anything we've seen before because it is nonlinear: $\hat{y}_{\text{quad}}(x)$ is a nonlinear function of x. As such, it doesn't seem like anything we've done so far would help: we've only seen how to work with linear functions with a single or with multiple inputs. However, it turns out that we can adapt the previous formula (for multivaraible regression) for this case. The basic idea is to form multiple inputs for multivariable linear regression by applying a bunch of different functions (here, $f_1(\cdot)$, $f_2(\cdot)$ and $f_3(\cdot)$), to the actual input, x_i,

$$\mathbf{x}^T(x) = \begin{pmatrix} f_1(x) & f_2(x) & f_3(x) \end{pmatrix} \tag{56}$$

So the big matrix $\mathbf{X}$ becomes,

$$\mathbf{X} = \begin{pmatrix} f_1(x_1) & f_2(x_1) & f_3(x_1) \\ f_1(x_2) & f_2(x_2) & f_3(x_2) \\ \vdots & \vdots & \vdots \\ f_1(x_N) & f_2(x_N) & f_3(x_N) \end{pmatrix} \tag{57}$$

Then, our prediction is a linear combination of the functions, $f_1(x)$, $f_2(x)$ and $f_3(x)$,

$$\hat{y}(x) = \mathbf{x}^T(x)\mathbf{w} = w_1 f_1(x) + w_2 f_2(x) + w_3 f_3(x). \tag{58}$$

To get back to our quadratic model (Eq. 55), we choose,

$$f_1(x) = 1 \tag{59}$$
$$f_2(x) = x \tag{60}$$
$$f_3(x) = x^2 \tag{61}$$

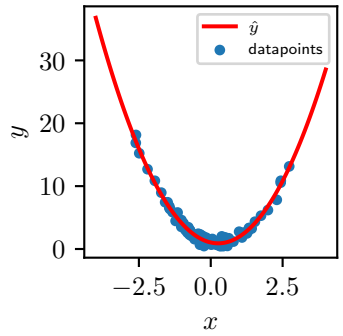
These choices for the functions give a feature vector (Eq. 56) of,

$$\mathbf{x}^T(x) = \begin{pmatrix} 1 & x & x^2 \end{pmatrix} \tag{62}$$

That gives a big matrix of the form,

$$\mathbf{X} = \begin{pmatrix} 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \\ \vdots & \vdots & \vdots \\ 1 & x_N & x_N^2 \end{pmatrix} \tag{63}$$

To get the optimal weights, we can plug this $\mathbf{X}$ and the usual outputs, $\mathbf{y}$ into Eq. (52). The resulting fitted line looks something like,



7 Regularisation

We saw (on the Python Notebook) that functions that are too complex can lead to overfitting. One approach to mitigating overfitting is to use "regularisation". So far, we have been minimizing the following loss problem,

$$\mathcal{L}(\mathbf{w}) = \sum_{i=1}^{N} (y_i - \hat{y}(x_i))^2 = \sum_{i=1}^{N} (y_i - \mathbf{w}^T \mathbf{x}(x_i))^2. \tag{64}$$

That miminimizes the squared distance between real data, y_i, and predictions, $\hat{y}(x_i)$. However, it doesn't tell us anything about what happens outside the training datapoints, $(x_1, x_2, \ldots, x_N)$. And the whole point is to do well outside of the training datapoints. One of the observations from the Notebook was that when our predictions go crazy (i.e. they get really big away from the training points) the weights also go crazy (i.e. they get really big). One approach to mitigating overfitting is therefore to add a term to the objective which penalises very large weights,

$$\mathcal{L}(\mathbf{w}) = \sum_{i=1}^{N} (y_i - \mathbf{w}^T \mathbf{x}(x_i))^2 + \lambda \sum_{i=1}^{D} w_i^2. \tag{65}$$

The optimal weights are now very similar to, but slightly different from the original optimal weights (in Eq. 52)

$$\mathbf{w}^* = \left(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I}\right)^{-1}\mathbf{X}^T\mathbf{y} \tag{66}$$

8 Exercises

All these exercises are based on the following dataset,

x	y
-2	-6.2
-1	-2.6
0	0.5
1	2.7
2	5.7

Exercise 1. *Which of the following predictions has the lowest loss (specifically, the sum-squared distance),*

$$\hat{y}_1(x) = x \tag{67}$$
$$\hat{y}_2(x) = 2x \tag{68}$$
$$\hat{y}_3(x) = 3x \tag{69}$$

Exercise 2. *Fit a straight line of the form,*

$$\hat{y}(x) = w_1 + w_2 x \tag{70}$$

using the formulae in Sec. 4.3 (specifically, Eq. 48).

Exercise 3. *Fit a straight line of the same form,*

$$\hat{y}(x) = w_1 + w_2 x \tag{71}$$

but this time, use the formulae in Sec. 5. Specifically, use two features:

$$f_1(x) = 1 \qquad\qquad f_2(x) = x \tag{72}$$

Exercise 4. *Fit a nonlinear prediction of the form,*

$$\hat{y}(x) = w_1 x + w_2 x^2. \tag{73}$$

Again, you must use the formulae in Sec. 5. What f_1 and f_2 do you need to use?

9 Answers

Answer 1. *Remember, the goal was to find which of the following prediction functions,*

$$\hat{y}_1(x) = x \tag{74}$$
$$\hat{y}_2(x) = 2x \tag{75}$$
$$\hat{y}_3(x) = 3x \tag{76}$$

had the lowest loss. To do that, we need to evaluate the loss for $\hat{y}_1$, $\hat{y}_2$ and $\hat{y}_3$. The predictions for $\hat{y}_1$ are,

$$\hat{y}_1(x_1) = x_1 = -2 \tag{77}$$
$$\hat{y}_1(x_2) = x_2 = -1 \tag{78}$$
$$\hat{y}_1(x_3) = x_3 = 0 \tag{79}$$
$$\hat{y}_1(x_4) = x_4 = 1 \tag{80}$$
$$\hat{y}_1(x_5) = x_5 = 2 \tag{81}$$

Thus, the loss for $\hat{y}_1$ is

$$\mathcal{L}_1 = \sum_{i=1}^{5} (y_i - \hat{y}_1(x_i))^2 \tag{82}$$
$$= ((-6.2) - (-2))^2 + ((-2.6) - (-1))^2 + (0.5 - 0)^2 + (2.7 - 1)^2 + (5.7 - 2)^2$$
$$= (-4.2)^2 + (-1.6)^2 + (0.5)^2 + (1.7)^2 + (3.7)^2 \tag{83}$$
$$= 37.03 \tag{84}$$

The predictions for $\hat{y}_2$ are,

$$\hat{y}_2(x_1) = 2x_1 = 2 \times (-2) = -4 \tag{85}$$
$$\hat{y}_2(x_2) = 2x_2 = 2 \times (-1) = -2 \tag{86}$$
$$\hat{y}_2(x_3) = 2x_3 = 2 \times 0 = 0 \tag{87}$$
$$\hat{y}_2(x_4) = 2x_4 = 2 \times 1 = 2 \tag{88}$$
$$\hat{y}_2(x_5) = 2x_5 = 2 \times 2 = 4 \tag{89}$$

Thus, the loss for $\hat{y}_2$ is

$$\mathcal{L}_2 = \sum_{i=1}^{5} (y_i - \hat{y}_2(x_i))^2 \tag{90}$$
$$= ((-6.2) - (-4))^2 + ((-2.6) - (-2))^2 + (0.5 - 0)^2 + (2.7 - 2)^2 + (5.7 - 4)^2$$
$$= (-2.2)^2 + (-0.6)^2 + (0.5)^2 + (0.7)^2 + (1.7)^2 \tag{91}$$
$$= 8.83 \tag{92}$$

The predictions for $\hat{y}_3$ are,

$$\hat{y}_3(x_1) = 3x_1 = 3 \times (-2) = -6 \tag{93}$$
$$\hat{y}_3(x_2) = 3x_2 = 3 \times (-1) = -3 \tag{94}$$
$$\hat{y}_3(x_3) = 3x_3 = 3 \times 0 = 0 \tag{95}$$
$$\hat{y}_3(x_4) = 3x_4 = 3 \times 1 = 3 \tag{96}$$
$$\hat{y}_3(x_5) = 3x_5 = 3 \times 2 = 6 \tag{97}$$

Thus, the loss for $\hat{y}_3$ is

$$\mathcal{L}_3 = \sum_{i=1}^{5} (y_i - \hat{y}_3(x_i))^2 \tag{98}$$
$$= ((-6.2) - (-6))^2 + ((-2.6) - (-3))^2 + (0.5 - 0)^2 + (2.7 - 3)^2 + (5.7 - 6)^2$$
$$= (-0.2)^2 + (0.4)^2 + (0.5)^2 + (-0.3)^2 + (-0.3)^2 \tag{99}$$
$$= 0.63 \tag{100}$$

Overall, the three losses are:

$$\mathcal{L}_1 = 37.03 \tag{101}$$
$$\mathcal{L}_2 = 8.83 \tag{102}$$
$$\mathcal{L}_3 = 0.63 \tag{103}$$

So the best (lowest loss) prediction is $\hat{y}_3$. This makes sense if we e.g. sketch a plot of the data itself: the slope is about 3.

Answer 2. *We start by computing w_2 in Eq. (48),*

$$w_2 = \frac{\overline{xy} - \overline{x}\,\overline{y}}{\overline{x^2} - (\overline{x})^2} \tag{104}$$

That requires us to compute all the "overbar" quantities,

$$\begin{aligned} \overline{xy} &= \tfrac{1}{N} \sum_{i=1}^{N} x_i y_i \\ &= \tfrac{1}{5} \left((-2)\times(-6.2) + (-1)\times(-2.6) + 0\times 0.5 + 1\times 2.7 + 2\times 5.7 \right) \\ &= \tfrac{1}{5} \left(12.4 + 2.6 + 0 + 2.7 + 10.4 \right) \\ &= \tfrac{1}{5} 29.1 = 5.82. \end{aligned} \tag{105}$$

$$\begin{aligned} \overline{x} &= \tfrac{1}{N} \sum_{i=1}^{N} x_i \\ &= \tfrac{1}{5} \left((-2) + (-1) + 0 + 1 + 2 \right) \\ &= \tfrac{1}{5} \times 0 = 0 \end{aligned} \tag{106}$$

$$\begin{aligned}\overline{y} &= \tfrac{1}{N}\sum_{i=1}^{N} y_i \\ &= \tfrac{1}{5}\left((-6.2) + (-2.6) + 0.5 + 2.7 + 5.7\right) \\ &= \tfrac{1}{5}0.1 = 0.02 \end{aligned} \tag{107}$$

$$\begin{aligned}\overline{x^2} &= \tfrac{1}{N}\sum_{i=1}^{N} x_i^2 \\ &= \tfrac{1}{5}\left((-2)^2 + (-1)^2 + 0^2 + 1^2 + 2^2\right) \\ &= \tfrac{1}{5}\left(4 + 1 + 0 + 1 + 4\right) \\ &= \tfrac{1}{5}10 = 2 \end{aligned} \tag{108}$$

Substituting these quantities into the formula for w_2, we get,

$$w_2 = \frac{5.82 - 0 \times 0.02}{2 - (0)^2} = \frac{5.82}{2} = 2.91 \tag{109}$$

Now we can ask whether this is in the right ballpark. The answer is yes. Remember from the first question that we expected the slope to be about 3 *(e.g. if we sketch a plot of the data).*

Now, we can compute w_1 in Eq. (48)*,*

$$w_1 = \tfrac{1}{N}\sum_{i=1}^{N}(y_i - w_2 x_i). \tag{110}$$

While we could compute this directly, with a little bit of manipulation, we can re-use our values for $\overline{y}$ and $\overline{x}$. Specifically, we expand the bracket,

$$w_1 = \tfrac{1}{N}\sum_{i=1}^{N} y_i - w_2 \tfrac{1}{N}\sum_{i=1}^{N} x_i. \tag{111}$$

Now, note that the first sum is $\overline{y}$ and the second sum is $\overline{x}$,

$$w_1 = \overline{y} - w_2\overline{x}. \tag{112}$$

Substituting the previously computed values of $\overline{y}$, $\overline{x}$, and w_2,

$$w_1 = 0.02 - 2.91 \times 0 = 0.02 \tag{113}$$

Answer 3. *Remember that when we're doing regression with multiple features, our predictor is Eq.* (58)*,*

$$\hat{y}(x) = \mathbf{x}^T(x)\mathbf{w} = w_1 f_1(x) + w_2 f_2(x). \tag{114}$$

Substituting the $f_1(x)$ and $f_2(x)$ given in the question, we get back Eq. (71)*, as we'd hope,*

$$\hat{y}(x) = \mathbf{x}^T(x)\mathbf{w} = w_1 + w_2 x. \tag{115}$$

Now, we can compute the optimal weights using,

$$\mathbf{w}^* = \left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{y}. \tag{116}$$

To do that, we first need to construct the big feature matrix $\mathbf{X}$ and the big result vector, $\mathbf{y}$. The result vector is just a vector of the y's

$$\mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{pmatrix} = \begin{pmatrix} -6.2 \\ -2.6 \\ 0.5 \\ 2.7 \\ 5.7 \end{pmatrix} \tag{117}$$

We get the feature matrix, $\mathbf{X}$, by applying $f_1(\cdot)$ and $f_2(\cdot)$ to the x's,

$$\mathbf{X} = \begin{pmatrix} f_1(x_1) & f_2(x_1) \\ f_1(x_2) & f_2(x_2) \\ f_1(x_3) & f_2(x_3) \\ f_1(x_4) & f_2(x_4) \\ f_1(x_5) & f_2(x_5) \end{pmatrix} = \begin{pmatrix} 1 & x_1 \\ 1 & x_2 \\ 1 & x_3 \\ 1 & x_4 \\ 1 & x_5 \end{pmatrix} = \begin{pmatrix} 1 & -2 \\ 1 & -1 \\ 1 & 0 \\ 1 & 1 \\ 1 & 2 \end{pmatrix} \tag{118}$$

Now, in principle all we have to do is to substitute this form for $\mathbf{X}$ and $\mathbf{y}$ into the form for the optimal weights,

$$\mathbf{w}^* = \left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{y}. \tag{119}$$

However, it turns out that we can simplify the computation a bit if we think carefully about the order of operations. And it turns out this thinking is helpful if we're doing the computation by hand (e.g. in an exam) or at a large scale on a computer. Specifically, remember that

$$\underbrace{\mathbf{w}^*}_{D\times 1} = \underbrace{\left(\mathbf{X}^T\mathbf{X}\right)^{-1}}_{D\times D}\underbrace{\mathbf{X}^T\mathbf{y}}_{D\times 1}. \tag{120}$$

and that the number of features ($D = 2$ in our case) is typically smaller than the number of datapoints ($N = 5$ in our case). Thus, it is easiest to first compute $\left(\mathbf{X}^T\mathbf{X}\right)^{-1}$ as that's just a 2×2 matrix, and to compute $\mathbf{X}^T\mathbf{y}$ as that's just a length 2 vector. Thus, we start by computing the two-by-two matrix, $\mathbf{X}^T\mathbf{X}$,

$$\mathbf{X}^T\mathbf{X} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ -2 & -1 & 0 & 1 & 2 \end{pmatrix}\begin{pmatrix} 1 & -2 \\ 1 & -1 \\ 1 & 0 \\ 1 & 1 \\ 1 & 2 \end{pmatrix} = \begin{pmatrix} 5 & 0 \\ 0 & 10 \end{pmatrix} \tag{121}$$

For instance, the top-left element is given by,

$$5 = 1\times1 + 1\times1 + 1\times1 + 1\times1 + 1\times1 = 1 + 1 + 1 + 1 + 1 \tag{122}$$

For instance, and top-right element is given by,

$$0 = 1\times(-2) + 1\times(-1) + 1\times0 + 1\times1 + 1\times2 = (-2) + (-1) + 0 + 1 + 2 \tag{123}$$

We can then compute $\left(\mathbf{X}^T\mathbf{X}\right)^{-1}$ *by using the usual formula for the* 2×2 *matrix inverse,*

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}^{-1} = \frac{1}{ad-bc}\begin{pmatrix} d & -b \\ -c & a \end{pmatrix} \tag{124}$$

Thus,

$$\left(\mathbf{X}\mathbf{X}^T\right)^{-1} = \begin{pmatrix} 5 & 0 \\ 0 & 10 \end{pmatrix}^{-1} \tag{125}$$

$$= \frac{1}{5\times10 - 0\times0}\begin{pmatrix} 10 & 0 \\ 0 & 5 \end{pmatrix} \tag{126}$$

$$= \frac{1}{50}\begin{pmatrix} 10 & 0 \\ 0 & 5 \end{pmatrix} \tag{127}$$

$$= \begin{pmatrix} \frac{1}{5} & 0 \\ 0 & \frac{1}{10} \end{pmatrix} \tag{128}$$

Next, we compute $\mathbf{X}^T\mathbf{y}$*, which is a matrix-vector product,*

$$\mathbf{X}^T\mathbf{y} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ -2 & -1 & 0 & 1 & 2 \end{pmatrix}\begin{pmatrix} -6.2 \\ -2.6 \\ 0.5 \\ 2.7 \\ 5.7 \end{pmatrix} \tag{129}$$

$$= \begin{pmatrix} 1\times(-6.2) + 1\times(-2.6) + 1\times0.5 + 1\times2.7 + 1\times5.7 \\ (-2)\times(-6.2) + (-1)\times(-2.6) + 0\times0.5 + 1\times2.7 + 2\times5.7 \end{pmatrix} \tag{130}$$

$$= \begin{pmatrix} (-6.2) + (-2.6) + 0.5 + 2.7 + 5.7 \\ 12.4 + 2.6 + 0 + 2.7 + 11.4 \end{pmatrix} \tag{131}$$

$$= \begin{pmatrix} 0.1 \\ 29.1 \end{pmatrix} \tag{132}$$

Now, we can substitute our values for $\left(\mathbf{X}^T\mathbf{X}\right)^{-1}$ *and* $\mathbf{X}^T\mathbf{y}$ *into the form for the optimal weights,*

$$\mathbf{w}^* = \left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{y} \tag{133}$$

$$\begin{pmatrix} w_1 \\ w_2 \end{pmatrix} = \begin{pmatrix} \frac{1}{5} & 0 \\ 0 & \frac{1}{10} \end{pmatrix}\begin{pmatrix} 0.1 \\ 29.1 \end{pmatrix} = \begin{pmatrix} 0.02 \\ 2.91 \end{pmatrix} \tag{134}$$

So $w_1 = 0.02$ *and* $w_2 = 2.91$*, which is exactly the same answer using the alternative strategy in the previous question!*

Answer 4. *Remember that when we're doing regression with multiple features, our predictor is (Eq. 58),*

$$\hat{y}(x) = \mathbf{x}^T(x)\mathbf{w} = w_1 f_1(x) + w_2 f_2(x). \tag{135}$$

Looking back at Eq. (73)*,*

$$\hat{y}(x) = w_1 x + w_2 x^2, \tag{136}$$

we need to choose,

$$f_1(x) = x \qquad\qquad f_2(x) = x^2. \tag{137}$$

Now, we can compute the optimal weights using,

$$\mathbf{w}^* = \left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{y}. \tag{138}$$

To do that, we first need to construct the big feature matrix $\mathbf{X}$ *and the big result vector,* $\mathbf{y}$*. The result vector is just a vector of the* y*'s*

$$\mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{pmatrix} = \begin{pmatrix} -6.2 \\ -2.6 \\ 0.5 \\ 2.7 \\ 5.7 \end{pmatrix} \tag{139}$$

We get the feature matrix, $\mathbf{X}$*, by applying* $f_1(\cdot)$ *and* $f_2(\cdot)$ *to the* x_i*'s,*

$$\mathbf{X} = \begin{pmatrix} f_1(x_1) & f_2(x_1) \\ f_1(x_2) & f_2(x_2) \\ f_1(x_3) & f_2(x_3) \\ f_1(x_4) & f_2(x_4) \\ f_1(x_5) & f_2(x_5) \end{pmatrix} = \begin{pmatrix} x_1 & x_1^2 \\ x_2 & x_2^2 \\ x_3 & x_3^2 \\ x_4 & x_4^2 \\ x_5 & x_5^2 \end{pmatrix} = \begin{pmatrix} -2 & (-2)^2 \\ -1 & (-1)^2 \\ 0 & 0^2 \\ 1 & 1^2 \\ 2 & 2^2 \end{pmatrix} = \begin{pmatrix} -2 & 4 \\ -1 & 1 \\ 0 & 0 \\ 1 & 1 \\ 2 & 4 \end{pmatrix} \tag{140}$$

We start by computing the two-by-two matrix, $\mathbf{X}^T\mathbf{X}$*,*

$$\mathbf{X}^T\mathbf{X} = \begin{pmatrix} -2 & -1 & 0 & 1 & 2 \\ 4 & 1 & 0 & 1 & 4 \end{pmatrix} \begin{pmatrix} -2 & 4 \\ -1 & 1 \\ 0 & 0 \\ 1 & 1 \\ 2 & 4 \end{pmatrix} = \begin{pmatrix} 10 & 0 \\ 0 & 34 \end{pmatrix} \tag{141}$$

We can then compute $\left(\mathbf{X}^T\mathbf{X}\right)^{-1}$ *by using the usual formula for the* 2×2 *matrix inverse,*

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}^{-1} = \frac{1}{ad - bc} \begin{pmatrix} d & -b \\ -c & a \end{pmatrix} \tag{142}$$

Thus,

$$\left(\mathbf{X}\mathbf{X}^T\right)^{-1} = \begin{pmatrix} 10 & 0 \\ 0 & 34 \end{pmatrix}^{-1} \tag{143}$$

$$= \frac{1}{10 \times 34 - 0 \times 0} \begin{pmatrix} 34 & 0 \\ 0 & 10 \end{pmatrix} \tag{144}$$

$$= \frac{1}{340} \begin{pmatrix} 34 & 0 \\ 0 & 10 \end{pmatrix} \tag{145}$$

$$= \begin{pmatrix} \frac{1}{10} & 0 \\ 0 & \frac{1}{34} \end{pmatrix} \tag{146}$$

Next, we compute $\mathbf{X}^T\mathbf{y}$*, which is a matrix-vector product,*

$$\mathbf{X}^T\mathbf{y} = \begin{pmatrix} -2 & -1 & 0 & 1 & 2 \\ 4 & 1 & 0 & 1 & 4 \end{pmatrix} \begin{pmatrix} -6.2 \\ -2.6 \\ 0.5 \\ 2.7 \\ 5.7 \end{pmatrix} \tag{147}$$

$$= \begin{pmatrix} (-2)\times(-6.2) + (-1)\times(-2.6) + 0\times0.5 + 1\times2.7 + 2\times5.7 \\ 4\times(-6.2) + 1\times(-2.6) + 0\times0.5 + 1\times2.7 + 4\times5.7 \end{pmatrix} \tag{148}$$

$$= \begin{pmatrix} 29.1 \\ -1.9 \end{pmatrix} \tag{149}$$

Now, we can substitute our values for $\left(\mathbf{X}^T\mathbf{X}\right)^{-1}$ *and* $\mathbf{X}^T\mathbf{y}$ *into the form for the optimal weights,*

$$\mathbf{w}^* = \left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{y} \tag{150}$$

$$\begin{pmatrix} w_1 \\ w_2 \end{pmatrix} = \begin{pmatrix} \frac{1}{10} & 0 \\ 0 & \frac{1}{34} \end{pmatrix} \begin{pmatrix} 29.1 \\ -1.9 \end{pmatrix} = \begin{pmatrix} 2.91 \\ 0.0559 \end{pmatrix} \tag{151}$$

So $w_1 = 2.91$*. Remember that this is now the linear weight (i.e. the term that multiplies* x*). Surprisingly, it has the same value as we saw previously (this doesn't happen in general, and only occurs in this case because the off-diagonals of* $\mathbf{X}^T\mathbf{X}$ *are zero). Additionally, the quadratic weight,* $w_2 = 0.0559$*, is small. That makes sense because the underlying data looks more like a straight line than a quadratic.*