

Linear models for regression

Our old simple (perhaps unrealistic) regression example:

Square footage	Price
1200	\$120
1340	\$125
1390	\$105
1400	\$130
1420	\$135
1500	\$145
1550	\$160
1700	\$155
1900	\$140
2150	\$130
2300	\$135

Linear prediction model:

$$\text{price} \approx \theta_0 + \theta_1 \times \text{square footage} \quad (1)$$

We can write this as:

$$[\theta_1 \quad \theta_0] \begin{bmatrix} 1200 & 1 \\ 1340 & 1 \\ 1390 & 1 \\ 1400 & 1 \\ 1420 & 1 \\ 1500 & 1 \\ 1550 & 1 \\ 1700 & 1 \\ 1900 & 1 \\ 2150 & 1 \\ 2300 & 1 \end{bmatrix}^\top = \begin{bmatrix} 120 \\ 125 \\ 105 \\ 130 \\ 135 \\ 145 \\ 160 \\ 155 \\ 140 \\ 130 \\ 135 \end{bmatrix} \quad (2)$$

It is easy to check that no $\{\theta_0, \theta_1\}$ exists that satisfies this.

Define a cost function, *least-squares*:

$$J(\theta) = \frac{1}{2} \sum_{n=1}^N [\theta^\top x_n - y_n]^2$$

This cost function penalizes outliers.

Now, we've changed the learning problem to an optimization problem: find θ to minimize $J(\theta)$.

Gradient Descent

Iteratively update θ according to:

$$\theta^{(t+1)} = \theta^{(t)} - \alpha \frac{\partial}{\partial \theta} J(\theta)$$

For the least-squares cost function, the partial is:

$$\frac{\partial}{\partial \theta} J(\theta) = \sum_{n=1}^N [\theta^\top x_n - y_n] x_n$$

The gradient is big on examples for which there is a high error.

α is a learning rate. Too low \rightarrow slow convergence, too high \rightarrow no convergence.

It turns out that we can actually obtain a solution in closed form. Let X be the data matrix, let Y be a (column) vector containing the targets. Then $X\theta - Y$ is a column vector whose n th element is $\theta^\top x_n - y_n$. So:

$$J(\theta) = \frac{1}{2} [X\theta - Y]^\top [X\theta - Y]$$

Then, we can compute the gradient:

$$\begin{aligned} \nabla_{\theta} J(\theta) &= \nabla_{\theta} \frac{1}{2} [X\theta - Y]^\top [X\theta - Y] \\ &= \frac{1}{2} \nabla_{\theta} [\theta^\top X^\top X\theta - \theta^\top X^\top Y - Y^\top X\theta + Y^\top Y] \\ &= \frac{1}{2} \nabla_{\theta} \text{tr} [\theta^\top X^\top X\theta - \theta^\top X^\top Y - Y^\top X\theta + Y^\top Y] \\ &= \frac{1}{2} \nabla_{\theta} [\text{tr} \theta^\top X^\top X\theta - 2 \text{tr} Y^\top X\theta] \\ &= \frac{1}{2} [X^\top X\theta + X^\top X\theta - 2X^\top Y] \\ &= X^\top X\theta - X^\top Y \end{aligned}$$

Thus, setting the gradient equal to zero, we obtain:

$$X^\top X\theta = X^\top Y$$

So:

$$\theta = [X^\top X]^{-1} X^\top Y$$

Maximum Likelihood

An alternative formulation: $y = \theta^\top x + \epsilon$, where $\epsilon \sim \mathcal{Nor}(0, \sigma^2)$. Then $y \sim \mathcal{Nor}(\theta^\top x, \sigma^2)$. Now, find θ to maximize likelihood of the training set.

$$L(D; \theta) = \prod_n \text{Nor}(y_n | \theta^\top x_n, \sigma^2)$$

$$= \prod_n \frac{1}{\sqrt{2\pi\sigma^2}} \exp \left[-\frac{1}{2\sigma^2} (y_n - \theta^\top x_n)^2 \right]$$

Take log to yield *log-likelihood*:

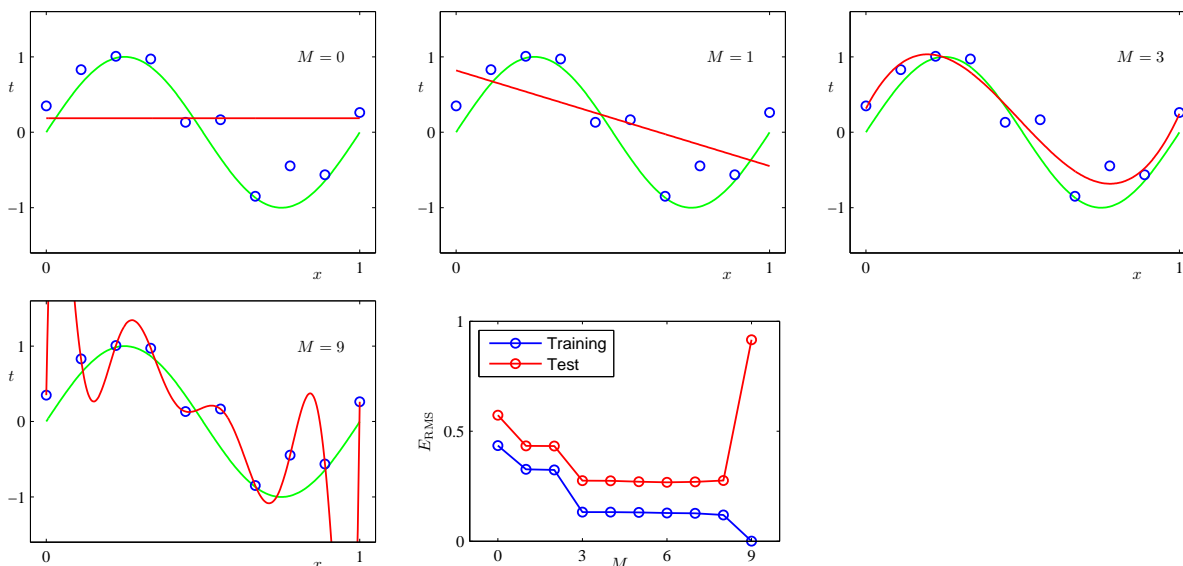
$$\ell(D; \theta) = \log L(D; \theta)$$

$$= \sum_n \left\{ -\frac{1}{2} \log(2\pi\sigma^2) - \frac{1}{2\sigma^2} (y_n - \theta^\top x_n)^2 \right\}$$

$$= -m \frac{1}{2} \log(2\pi\sigma^2) - \frac{1}{2\sigma^2} \sum_n (y_n - \theta^\top x_n)^2$$

The θ that maximizes this is clearly the same as in the least-squares formulation!

Overfitting



In other words: why should fitting the training data mean that we'll do well on test data?

In general, it holds that:

$$\text{expected loss} \leq \text{empirical loss} + \lambda \times \text{function complexity}$$

More on this in a few weeks.

Go back to least-squares regression. We want to penalize large weights. Try:

$$J(\theta) = \frac{1}{2} \sum_n [\theta^\top x_n - y_n]^2 + \frac{\lambda}{2} \theta^\top \theta$$

This is an ℓ_2 penalty. λ controls how complex functions we allow.

Easy to compute gradient:

$$\begin{aligned}\nabla_{\theta} J(\theta) &= \nabla_{\theta} \frac{1}{2} [X\theta - Y]^{\top} [X\theta - Y] + \frac{\lambda}{2} \theta^{\top} \theta \\ &= X^{\top} X \theta - X^{\top} Y + \lambda \theta\end{aligned}$$

So we can solve for θ :

$$\begin{aligned}(X^{\top} X + \lambda I) \theta &= X^{\top} Y \\ \implies \theta &= [X^{\top} X + \lambda I]^{-1} X^{\top} Y\end{aligned}$$

This is especially nice when $X^{\top} X$ is illconditioned.

We can also do a probabilistic interpretation, putting a prior on θ : $\theta \sim \mathcal{Nor}(0, \lambda^{-1})$.

In general, too many features is bad, too few is bad. Why? We want to minimize the expected cost (going back to un-regularized). Suppose $f = f(x)$ and $t = f + \epsilon$. Write y for $\theta^{\top} x$. Then:

$$\mathbb{E}[J(\theta)] = \mathbb{E} \left[\frac{1}{2} \sum_n (t_n - y_n)^2 \right] = \frac{1}{2} \sum_n \mathbb{E} [(t_n - y_n)^2]$$

Let's look at the expectation:

$$\begin{aligned}\mathbb{E} [(t_n - y_n)^2] &= \mathbb{E} [(t_n - f_n + f_n - y_n)^2] \\ &= \mathbb{E} [(t_n - f_n)^2] + \mathbb{E} [(f_n - y_n)^2] + 2\mathbb{E} [(f_n - y_n)(t_n - y_n)] \\ &= \mathbb{E} [\epsilon^2] + \mathbb{E} [(f_n - y_n)^2] + 2 \{ \mathbb{E}[f_n t_n] - \mathbb{E}[f_n^2] - \mathbb{E}[y_n t_n] + \mathbb{E}[y_n f_n] \} \\ &= \mathbb{E} [\epsilon^2] + \mathbb{E} [(f_n - y_n)^2]\end{aligned}$$

AND

$$\begin{aligned}\mathbb{E} [(f_n - y_n)^2] &= \mathbb{E} [(f_n - \mathbb{E}[y_n] + \mathbb{E}[y_n] - y_n)^2] \\ &= \mathbb{E} [(f_n - \mathbb{E}[y_n])^2] + \mathbb{E} [(\mathbb{E}[y_n] - y_n)^2] + 2\mathbb{E} [(\mathbb{E}[y_n] - y_n)(f_n - y_n)] \\ &= \mathbb{E} [(f_n - \mathbb{E}[y_n])^2] + \mathbb{E} [(\mathbb{E}[y_n] - y_n)^2] \\ \implies \\ \mathbb{E} [(t_n - y_n)^2] &= \mathbb{E} [\epsilon^2] + \mathbb{E} [(f_n - \mathbb{E}[y_n])^2] + \mathbb{E} [(\mathbb{E}[y_n] - y_n)^2] \\ &= \mathbb{V}[\text{noise}] + \text{bias}^2 + \mathbb{V}[y]\end{aligned}$$