Learning and Linear Regression

Richard Sowers

University of Illinois at Urbana-Champaign

ILLINOIS
ISE | Industrial & Enterprise Systems Engineering
GRAINGER COLLEGE OF ENGINEERING

ILLINOIS
Mathematics
COLLEGE OF LIBERAL ARTS & SCIENCES

Linear Regression 1: Best Line

Richard Sowers

University of Illinois at Urbana-Champaign

ILLINOIS
ISE | Industrial & Enterprise Systems Engineering
GRAINGER COLLEGE OF ENGINEERING

ILLINOIS
Mathematics
COLLEGE OF LIBERAL ARTS & SCIENCES

Let's understand *linear regression* as a type of machine learning. Let's consider

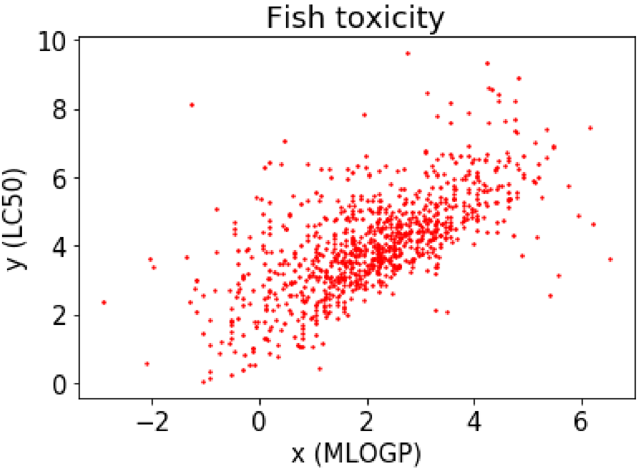
```
https://archive.ics.uci.edu/ml/datasets/QSAR+fish+toxicity
```

as a sample dataset.

	CIC0	SM1_Dz(Z)	GATS1i	NdsCH	NdssC	MLOGP	LC50
0	3.260	0.829	1.676	0	1	1.453	3.770
1	2.189	0.580	0.863	0	0	1.348	3.115
2	2.125	0.638	0.831	0	0	1.348	3.531
3	3.027	0.331	1.472	1	0	1.807	3.510
4	2.094	0.827	0.860	0	0	1.886	5.390

We want to understand how an $\mathbb{R}$-valued *response* (y) depends on an $\mathbb{R}$-valued *feature* (x) (perhaps one of several possible features) We have N test datapoints which we want to *learn* from.

feature (x): MLOGP
response (y): LC50
N=908

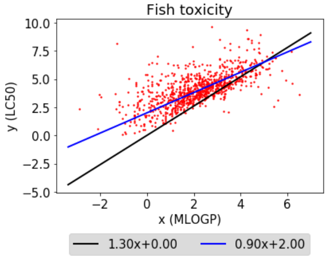


The scatterplot sort of clusters around a line

We want to *model* the data as

$$y = \underbrace{mx + b}_{\ell_{m,b}(x)}$$

where $\ell_{m,b}$ is a line parametrized by slope m and y-intercept b. Instead of fixing the parameters m and b and studying $\ell_{m,b}$, we are here given N datapoints $\{(x_n, y_n)\}_{n=1}^N$ and one tries to *learn* m and b.

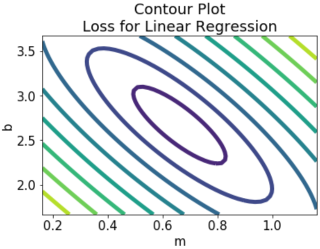


We want to find the parameters *m* and *b* which *best* explain the data.

We quantify *best* by writing down the the error as a *loss* function

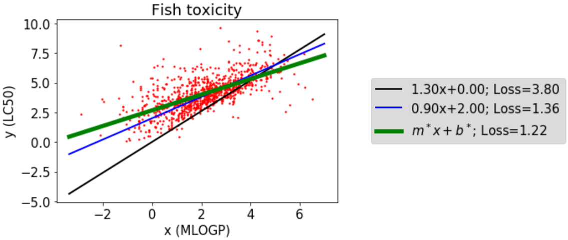
$$\Lambda(m,b) \stackrel{\text{def}}{=} \frac{1}{N}\sum_{n=1}^{N} \underbrace{\left(\overbrace{y_n - \ell_{m,b}(x_n)}^{\varepsilon_n(m,b)}\right)^2}_{\lambda_n(m,b)}$$

(the mean square error) and then minimizing over $(m,b) \in \mathbb{R}^2$.



$$\Lambda(m^*, b^*) = \min_{(m,b)\in\mathbb{R}^2} \Lambda(m, b).$$

$m^* = 0.66$
$b^* = 2.67$
minimal loss=1.22



note: The cost function Λ can be understood as the *variance* of the error

$$y_n = mx_n + b + \text{ERROR}_n$$

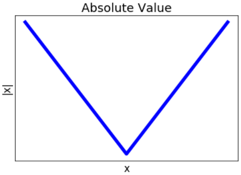
if the ERROR_n's are statistically independent and identically distributed random variables.

Metrics

Once we have identified the “best” parameters (m^*, b^*) (which minimize Λ), we can then compute a *metric*

$$\mu_{\text{metric}} \stackrel{\text{def}}{=} \frac{1}{N}\sum_{n=1}^{N}\left|\underbrace{y_n - \ell_{m^*,b^*}(x_n)}_{\varepsilon_n(m^*,b^*)}\right|$$

$$\boxed{\mu_{metric} = 0.82}$$



which tells us how good our machine learning procedure turned out to be. In this case, the metric μ

- Is not differentiable ($x \mapsto |x|$ fails to be differentiable at 0)
- Lacks the statistical appeal of the loss function Λ.

Metrics vs Losses

Generally,

- The *metric* often reflects statistical assessment. The *loss* function tries to capture the idea of the metric, but is regular enough to (efficiently) apply (robust) optimization algorithms.
- There are range of existing losses and metrics for existing problems. Losses and metrics for new problems are something of an art. A loss is a “good enough” approximation of metric, where “good enough” respects the fact that meaningful applications of machine learning often have noise and complex structure (as opposed to mathematical models).

Summary

Let's write down some salient points:

- We have a set $\{(\text{feature}_n, \text{response}_n)\}_{n=1}^{N}$ of *ground truth* datapoints which we will use to *supervise* our efforts.
- We have a *model* $\ell_{m,b}$ of the feature-response relationship; this model depends on a finite collection of model parameters. which we want to identify
- We have a *loss* function Λ which quantifies the error (between the model and the ground truth data) as a function of the model parameters. This loss can further be broken down as a normalized sum of loss functions for each data point (the λ_n's). We want to minimize this aggregate loss function over choices of the model parameters.
- We have a *metric* which quantifies how well we are able to express the ground truth in terms of the optimal model.

ILLINOIS

illinois.edu

Linear Regression 2: Loss Function

Richard Sowers

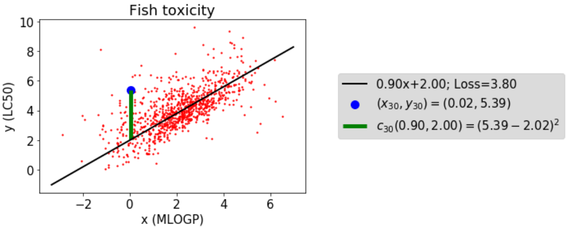
University of Illinois at Urbana-Champaign

ILLINOIS
ISE | Industrial & Enterprise
Systems Engineering
GRAINGER COLLEGE OF ENGINEERING

ILLINOIS
Mathematics
COLLEGE OF LIBERAL ARTS & SCIENCES

Mathematics of Loss Function

Let's look a bit more closely at the λ_n's. Each $\lambda_n(m, b)$ is the square of the vertical distance $\varepsilon_n(m, b)$ from the line $\ell_{m,b}$ to the point (x_n, y_n).



$$\boxed{\begin{aligned} (x_{30}, y_{30}) &= (0.02, 5.39) \\ \lambda_{30}(m, b) &= (5.39 - 0.02m - b)^2 \end{aligned}}$$

Mathematically,

$$\overbrace{\varepsilon_n(m,b)}^{\substack{\mathbf{p}=\begin{pmatrix} m \\ b \end{pmatrix} \\ =\text{parameters of model}}} = y_n - mx_n - b = y_n - \overbrace{\begin{pmatrix} x_n & 1 \end{pmatrix}}^{\mathbf{X}_n^T} \begin{pmatrix} m \\ b \end{pmatrix}$$

$$\lambda_n(\mathbf{p}) = (\varepsilon_n(\mathbf{p}))^2$$

The augmented feature variable

$$\mathbf{X}_n \stackrel{\text{def}}{=} \begin{pmatrix} x_n \\ 1 \end{pmatrix}$$

allows us to efficiently consider shifts in the data.

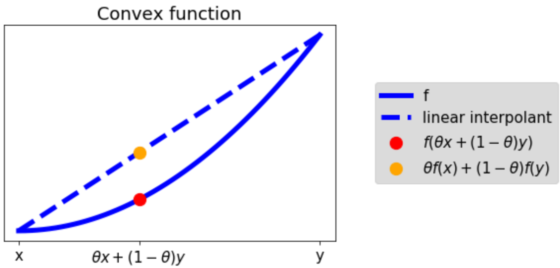
We claim that λ_n is *convex* in $\mathbf{p}$; i.e., for any $\mathbf{p}_1$ and $\mathbf{p}_2$ in $\mathbb{R}^2$ and $\theta \in [0,1]$,

$$
\begin{aligned}
\underbrace{\varepsilon_n}_{\text{affine}}(\theta \mathbf{p}_1 + (1-\theta)\mathbf{p}_2) &= y_n - \mathbf{X}_n^T \{\theta \mathbf{p}_1 + (1-\theta)\mathbf{p}_2\} \\
&= y_n - \theta \mathbf{X}_n^T \mathbf{p}_1 - (1-\theta)\mathbf{X}_n^T \mathbf{p}_2 \\
&= \theta \left\{ y_n - \mathbf{X}_n^T \mathbf{p}_1 \right\} + (1-\theta)\left\{ y_n - \mathbf{X}_n^T \mathbf{p}_2 \right\} \\
&= \theta \varepsilon_n(\mathbf{p}_1) + (1-\theta)\varepsilon_n(\mathbf{p}_2) \\
\lambda_n(\theta \mathbf{p}_1 + (1-\theta)\mathbf{p}_2) &= \underbrace{(\theta \varepsilon_n(\mathbf{p}_1) + (1-\theta)\varepsilon_n(\mathbf{p}_2))^2}_{f(\theta)}
\end{aligned}
$$

Then

$$
\begin{aligned}
f(\theta) &= (\varepsilon_n(\mathbf{p}_2) + \theta\{\varepsilon_n(\mathbf{p}_1) - \varepsilon_n(\mathbf{p}_2)\})^2 \\
f'(\theta) &= 2(\varepsilon_n(\mathbf{p}_2) + \theta\{\varepsilon_n(\mathbf{p}_1) - \varepsilon_n(\mathbf{p}_2)\})\{\varepsilon_n(\mathbf{p}_1) - \varepsilon_n(\mathbf{p}_2)\} \\
f''(\theta) &= 2\{\varepsilon_n(\mathbf{p}_1) - \varepsilon_n(\mathbf{p}_2)\}^2 \geq 0
\end{aligned}
$$

Since $f'' \geq 0$, f' is nondecreasing, so f should be convex.



$$f(\theta) - f(0) = \int_{s=0}^{\theta} f'(s)ds \leq f'(\theta)\theta$$

$$f(1) - f(\theta) = \int_{s=\theta}^{1} f'(s)ds \geq f'(\theta)(1-\theta);$$

thus

$$\theta\{f(1) - f(\theta)\} \geq f'(\theta)\theta(1-\theta) \geq (1-\theta)\{f(\theta) - f(0)\}$$

which can be rearranged as

$$\underbrace{\theta f(1) + (1-\theta)f(0)}_{\theta\lambda_n(\mathbf{p}_1)+(1-\theta)\lambda_n(\mathbf{p}_2)} \geq \theta f(\theta) + (1-\theta)f(\theta) = \underbrace{f(\theta)}_{\lambda_n(\theta\mathbf{p}_1+(1-\theta)\mathbf{p}_2)}.$$

Since each λ_n is convex,

$$
\begin{aligned}
\overbrace{\frac{1}{N}\sum_{n=1}^N \underbrace{\lambda_n(\theta \mathbf{p}_1 + (1-\theta)\mathbf{p}_2)}_{f_n(\theta)}}^{\Lambda(\theta \mathbf{p}_1 + (1-\theta)\mathbf{p}_2)}
&\le \frac{1}{N}\sum_{n=1}^N \left\{\theta \underbrace{\lambda_n(\mathbf{p}_1)}_{f_n(1)} + (1-\theta)\underbrace{\lambda_n(\mathbf{p}_2)}_{f_n(0)}\right\} \\
&= \theta \underbrace{\frac{1}{N}\sum_{n=1}^N \lambda_n(\mathbf{p}_1)}_{\Lambda(\mathbf{p}_1)} + (1-\theta)\underbrace{\frac{1}{N}\sum_{n=1}^N \lambda_n(\mathbf{p}_2)}_{\Lambda(\mathbf{p}_2)}
\end{aligned}
$$

so Λ is also convex.

Convex functions are nice; if

$$\underbrace{\min_{\mathbf{p}\in\mathbb{R}^2}\Lambda(\mathbf{p})}_{\underline{\Lambda}} = \Lambda(\mathbf{p}_1^*) = \Lambda(\mathbf{p}_2^*) \qquad \text{(two minimizers)}$$

then for any $\theta \in [0,1]$,

$$\begin{aligned}\underbrace{\min_{\mathbf{p}\in\mathbb{R}^2}\Lambda(\mathbf{p})}_{\underline{\Lambda}} &\le \Lambda\left(\theta\mathbf{p}_1^* + (1-\theta)\mathbf{p}_2^*\right)\\ &\le \theta\underbrace{\Lambda(\mathbf{p}_2^*)}_{=\underline{\Lambda}} + (1-\theta)\underbrace{\Lambda(\mathbf{p}_1^*)}_{=\underline{\Lambda}} = \underline{\Lambda}\end{aligned}$$

so

$$\Lambda\left(\theta\mathbf{p}_1^* + (1-\theta)\mathbf{p}_2^*\right) = \underline{\Lambda};$$

the set of minimizers is convex (and thus connected).

If Λ is *strictly* convex, inequalities are strict, leading to a contradiction if the set of minimizer is not unique.

Let's next look at level sets of ε_n (and thus λ_n). For any $\mathbf{p} \in \mathbb{R}^2$ and any $\alpha \in \mathbb{R}$,

$$\begin{aligned}
\varepsilon_n\left(\mathbf{p} + \alpha \underbrace{\begin{pmatrix} 1 \\ -x_n \end{pmatrix}}_{\substack{\mathbf{X}_n^\perp \\ =\text{orthogonal complement of } \mathbf{X}_n}}\right) &= y_n - \mathbf{X}_n^T \left\{\mathbf{p} + \alpha \mathbf{X}_n^\perp\right\} \\
&= \underbrace{y_n - \mathbf{X}_n^T \mathbf{p}}_{\varepsilon_n(\mathbf{p})} - \underbrace{\alpha \mathbf{X}_n^T \mathbf{X}_n^\perp}_{\substack{\begin{pmatrix} x_n & 1 \end{pmatrix}\begin{pmatrix} 1 \\ -x_n \end{pmatrix} \\ =0}}
\end{aligned}$$

so

$$\lambda_n\left(\mathbf{p} + \alpha \mathbf{X}_n^\perp\right) = \left(\varepsilon_n\left(\mathbf{p} + \alpha \mathbf{X}_n^\perp\right)\right)^2 = (\varepsilon_n(\mathbf{p}))^2$$

i.e., λ_n is constant along displacements in the direction of $\mathbf{X}_n^\perp$ (and thus λ_n is not strictly convex).

We can graphically understand

$$\left\{\mathbf{p} \in \mathbb{R}^2 : \lambda_n(\mathbf{p}) = 1.44\right\};$$

cost is square of vertical distance from (x_n, y_n) to line with parameters $\mathbf{p} = (m, b)$. $\lambda_n(\mathbf{p}) = 1.44\,(= 1.2^2)$ corresponds to a line (of some slope m) passing through $(x_n, y_n \pm 1.2)$. A line of slope m passing through $(x_n, y_n \pm 1.2)$ is given by (point-slope formula)

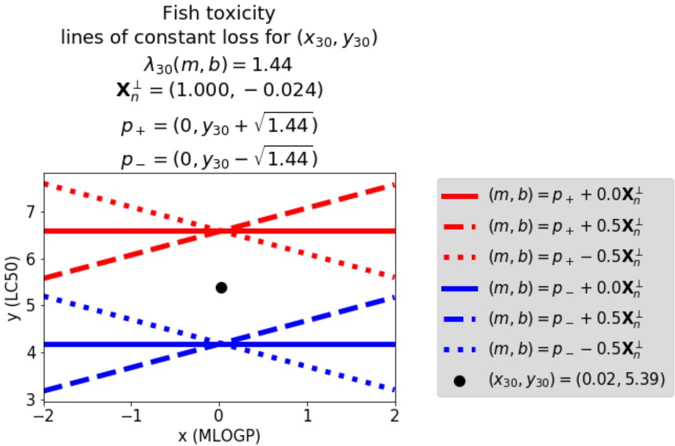
$$\frac{y - (y_n \pm 1.2)}{x - x_n} = m;$$

namely

$$y = mx + \underbrace{\{y_n \pm 1.2 - mx_n\}}_{b}$$

so

$$\mathbf{p} = \begin{pmatrix} m \\ b \end{pmatrix} = \begin{pmatrix} m \\ y_n \pm 1.2 - mx_n \end{pmatrix} = \begin{pmatrix} 0 \\ y_n \pm 1.2 \end{pmatrix} + m \underbrace{\begin{pmatrix} 1 \\ -x_n \end{pmatrix}}_{\mathbf{x}_n^\perp}$$


Fish toxicity
lines of constant loss for (x_{30}, y_{30})
$\lambda_{30}(m,b) = 1.44$
$\mathbf{X}_n^{\perp} = (1.000, -0.024)$
$p_+ = (0, y_{30} + \sqrt{1.44})$
$p_- = (0, y_{30} - \sqrt{1.44})$
y (LC50)
x (MLOGP)
$(m,b) = p_+ + 0.0\mathbf{X}_n^{\perp}$
$(m,b) = p_+ + 0.5\mathbf{X}_n^{\perp}$
$(m,b) = p_+ - 0.5\mathbf{X}_n^{\perp}$
$(m,b) = p_- + 0.0\mathbf{X}_n^{\perp}$
$(m,b) = p_- + 0.5\mathbf{X}_n^{\perp}$
$(m,b) = p_- - 0.5\mathbf{X}_n^{\perp}$
$(x_{30}, y_{30}) = (0.02, 5.39)$

We can explicitly compute Hessian of costs;

$$\lambda_n(\mathbf{p}) = \left(y_n - \mathbf{X}_n^T \mathbf{p}\right)^2$$

(λ_n is quadratic in $\mathbf{p}$); λ_n has Hessian

$$\underbrace{\mathbf{X}_n \mathbf{X}_n^T}_{Q_n} = \begin{pmatrix} x_n \\ 1 \end{pmatrix} \begin{pmatrix} x_n & 1 \end{pmatrix} = \begin{pmatrix} x_n^2 & x_n \\ x_n & 1 \end{pmatrix}$$

- Q_n has rank one ($\mathbf{X}_n^\perp$ is in kernel), although it is 2×2
- Q_n is symmetric
- $Q_n \geq 0$; $\mathbf{p}^T Q_n \mathbf{p} = \left(\mathbf{X}_n^T \mathbf{p}\right)^2 \geq 0$ for any $\mathbf{p} \in \mathbb{R}^2$.

Thus Λ has Hessian

$$Q \stackrel{\text{def}}{=} \frac{1}{N}\sum_{n=1}^{N} Q_n = \frac{1}{N}\sum_{n=1}^{N} \begin{pmatrix} x_n^2 & x_n \\ x_n & 1 \end{pmatrix}$$

- Q is full rank (unless all x_n's agree)
- Q is symmetric
- $Q_n \geq 0$; $\mathbf{p}^T Q \mathbf{p} = \frac{1}{N}\sum_{n=1}^{N} \mathbf{p}^T Q_n \mathbf{p} \geq 0$ for any $\mathbf{p} \in \mathbb{R}^2$.

Summary

- The cost function Λ can be written as an average of cost functions λ_n, where each λ_n is a cost between the model and a single record.
- Each λ_n is convex; Λ is thus convex (and typically strictly convex).

ILLINOIS

illinois.edu

Linear Regression 3: Minimal Loss

Richard Sowers

University of Illinois at Urbana-Champaign

ILLINOIS
ISE | Industrial & Enterprise Systems Engineering
GRAINGER COLLEGE OF ENGINEERING

ILLINOIS
Mathematics
COLLEGE OF LIBERAL ARTS & SCIENCES

Linear regression has an explicit solution, given by the Euler equations of optimality. Let's perturb the optimal $\mathbf{p}^*$ by amount $\delta \in \mathbb{R}$ in direction $\tilde{\mathbf{p}}$.

$$\varepsilon_n(\mathbf{p}^* + \delta\tilde{\mathbf{p}}) = y_n - \mathbf{X}_n^T \{\mathbf{p}^* + \delta\tilde{\mathbf{p}}\} = \underbrace{y_n - \mathbf{X}_n^T \mathbf{p}^*}_{\varepsilon_n(\mathbf{p}^*)} - \delta \mathbf{X}_n^T \tilde{\mathbf{p}}$$

so

$$\Lambda(\mathbf{p}^* + \delta\tilde{\mathbf{p}}) = \frac{1}{N}\sum_{n=1}^N \left(\varepsilon_n(\mathbf{p}^*) - \delta \mathbf{X}_n^T \tilde{\mathbf{p}}\right)^2.$$

Taking derivatives with respect to δ (at $\delta = 0$),

$$\underbrace{D\Lambda(\mathbf{p}^*)\tilde{\mathbf{p}}}_{\text{derivative at } \mathbf{p}^* \text{ in direction } \tilde{\mathbf{p}} \in \mathbb{R}^2} = -2\frac{1}{N}\sum_{n=1}^N \varepsilon_n(\mathbf{p}^*)\mathbf{X}_n^T \tilde{\mathbf{p}}$$

$$\underbrace{D\Lambda(\mathbf{p}^*)}_{\text{linear map from } \mathbb{R}^2 \text{ to } \mathbb{R}} = -2\frac{1}{N}\sum_{n=1}^N \varepsilon_n(\mathbf{p}^*)\mathbf{X}_n^T$$

The *Euler condition of optimality* is that if

$$\Lambda(\mathbf{p}^*) = \min_{\mathbf{p}\in\mathbb{R}^2} \Lambda(\mathbf{p})$$

then

$$0 = D\Lambda(\mathbf{p}^*) = -2\frac{1}{N}\sum_{n=1}^{N} \underbrace{\left\{y_n - \mathbf{X}_n^T\mathbf{p}^*\right\}}_{\varepsilon_n(\mathbf{p}^*)} \mathbf{X}_n^T$$

which can be rearranged ($\mathbf{X}_n^T\mathbf{p}^* = \mathbf{p}^{*,T}\mathbf{X}_n$ is a scalar) as

$$\underbrace{\frac{1}{N}\sum_{n=1}^{N} y_n\mathbf{X}_n}_{\mathbf{v}} = \underbrace{\frac{1}{N}\sum_{n=1}^{N} \mathbf{X}_n\mathbf{X}_n^T}_{Q} \mathbf{p}^*$$

so

$$\mathbf{p}^* = Q^{-1}\mathbf{v}.$$

This agrees with standard formulæ.

$$\mathbf{v} = \frac{1}{N}\sum_{n=1}^{N}\begin{pmatrix} y_n x_n \\ y_n \end{pmatrix} = \begin{pmatrix} \overline{xy} \\ \overline{y} \end{pmatrix}$$

$$Q = \frac{1}{N}\sum_{n=1}^{N}\begin{pmatrix} x_n \\ 1 \end{pmatrix}\begin{pmatrix} x_n & 1 \end{pmatrix} = \frac{1}{N}\sum_{n=1}^{N}\begin{pmatrix} x_n^2 & x_n \\ x_n & 1 \end{pmatrix}$$

$$= \begin{pmatrix} \overline{x^2} & \overline{x} \\ \overline{x} & 1 \end{pmatrix}$$

where $\overline{\cdot}$ empirically averages.

Then

$$\begin{pmatrix} m^* \\ b^* \end{pmatrix} = \underbrace{\mathbf{p}^*}_{Q^{-1}\mathbf{v}} = \frac{1}{\overline{x^2} - \overline{x}^2} \begin{pmatrix} 1 & -\overline{x} \\ -\overline{x} & \overline{x^2} \end{pmatrix} \begin{pmatrix} \overline{yx} \\ \overline{y} \end{pmatrix}$$

$$= \begin{pmatrix} \frac{\overline{xy} - \overline{x}\cdot\overline{y}}{\overline{x^2} - \overline{x}^2} \\ \frac{-\overline{x}\cdot\overline{xy} + \overline{x^2}\,\overline{y}}{\overline{x^2} - \overline{x}^2} \end{pmatrix} = \begin{pmatrix} \frac{\overline{xy} - \overline{x}\cdot\overline{y}}{\overline{x^2} - \overline{x}^2} \\ \frac{-\overline{x}\{\overline{xy} - \overline{x}\cdot\overline{y}\} + \left\{\overline{x^2} - \overline{x}^2\right\}\overline{y}}{\overline{x^2} - \overline{x}^2} \end{pmatrix}$$

$$= \begin{pmatrix} \frac{\overline{xy} - \overline{x}\cdot\overline{y}}{\overline{x^2} - \overline{x}^2} \\ -\overline{x}\frac{\overline{xy} - \overline{x}\cdot\overline{y}}{\overline{x^2} - \overline{x}^2} + \overline{y} \end{pmatrix}$$

The regression line is thus the well-known formula

$$\ell_{m^*,b^*}(x) = m^* x + b^* = \frac{\overline{xy} - \overline{x}\cdot\overline{y}}{\overline{x^2} - \overline{x}^2} x - \overline{x}\frac{\overline{xy} - \overline{x}\cdot\overline{y}}{\overline{x^2} - \overline{x}^2} + \overline{y}$$

$$= \overline{y} + \underbrace{\frac{\overline{xy} - \overline{x}\cdot\overline{y}}{\overline{x^2} - \overline{x}^2}}_{=\frac{\text{Covariance}(x,y)}{\text{Variance}(x)}} (x - \overline{x})$$

Z-scores

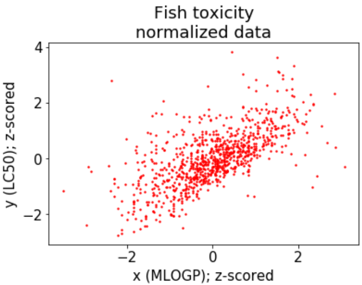
By first subtracting the mean and then dividing by standard deviation (i.e., taking *z-scores*), we can normalize datasets to be mean zero and unit standard deviation, leading to better numerical stability.

$$\sigma_{X,X} \stackrel{\text{def}}{=} \overline{(x-\overline{x})^2}; \quad \sigma_{Y,Y} \stackrel{\text{def}}{=} \overline{(y-\overline{y})^2}; \quad \sigma_{X,Y} \stackrel{\text{def}}{=} \overline{(x-\overline{x})(y-\overline{y})}.$$

then

$$x_n^z \stackrel{\text{def}}{=} \frac{x_n-\overline{x}}{\sqrt{\sigma_{X,X}}} \qquad y_n^z \stackrel{\text{def}}{=} \frac{y_n-\overline{y}}{\sqrt{\sigma_{Y,Y}}}$$

Then $\{(x_n^z, y_n^z)\}_{n=1}^N$ is centered at the origin and has unit standard deviation in the x and y directions.



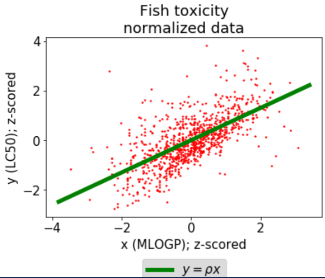
Our original formula for regression was

$$\ell_{m^*,b^*}(x) = \overline{y} + \frac{\sigma_{X,Y}}{\sigma_{X,X}}(x - \overline{x})$$

which we can rewrite as

$$\underbrace{\frac{\ell_{m^*,b^*}(x) - \overline{y}}{\sqrt{\sigma_{Y,Y}}}}_{\text{z-score of response}} = \underbrace{\frac{\sigma_{X,Y}}{\sqrt{\sigma_{Y,Y}}\sqrt{\sigma_{X,X}}}}_{\substack{=\rho_{X,Y} \\ \text{correlation coefficient between } X \text{ and } Y}} \underbrace{\left(\frac{x - \overline{x}}{\sqrt{\sigma_{X,X}}}\right)}_{\text{z-score of feature}} \qquad (+0)$$

regressed offset b for z-scored data: 1.01e-16
regressed slope m for z-scored data: 0.652
correlation coefficient for z-scored data: 0.652



ILLINOIS

illinois.edu

Linear Regression 4. Gradient Descent

Richard Sowers

University of Illinois at Urbana-Champaign

ILLINOIS
ISE | Industrial & Enterprise Systems Engineering
GRAINGER COLLEGE OF ENGINEERING

ILLINOIS
Mathematics
COLLEGE OF LIBERAL ARTS & SCIENCES

$$D\Lambda(\mathbf{p}) = -2\frac{1}{N}\sum_{n=1}^{N}\varepsilon_n(\mathbf{p})\mathbf{X}_n^T.$$

can convert derivative to gradient through duality on $\mathbb{R}^2$ with standard inner product $\langle x, y\rangle_{\mathbb{R}^2} = x^T y$;

$$\langle \nabla\Lambda(\mathbf{p}), \tilde{\mathbf{p}}\rangle_{\mathbb{R}^2} = \underbrace{D\Lambda(\mathbf{p})}_{(\nabla\Lambda(\mathbf{p}))^T}\tilde{\mathbf{p}}$$

so

$$\nabla\Lambda(\mathbf{p}) = -2\frac{1}{N}\sum_{n=1}^{N}\varepsilon_n(\mathbf{p})\mathbf{X}_n$$

For $\delta > 0$ small,

$$\Lambda\left(\mathbf{p} - \delta\nabla\Lambda(\mathbf{p})\right) \approx \Lambda(\mathbf{p}) - \delta \overbrace{D\Lambda(\mathbf{p})\nabla\Lambda(\mathbf{p})}^{\langle\nabla\Lambda(\mathbf{p}),\nabla\Lambda(\mathbf{p})\rangle_{\mathbb{R}^2}}$$
$$= \Lambda(\mathbf{p}) - \delta\|\nabla\Lambda(\mathbf{p})\|_{\mathbb{R}^2}^2 \leq \Lambda(\mathbf{p})$$

Gradient descent is

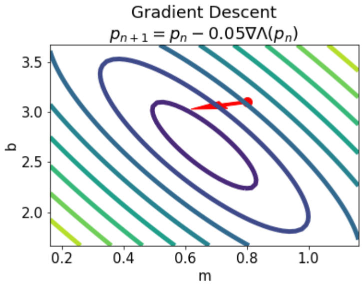
$$\mathbf{p}_{n+1} = \mathbf{p}_n - \delta\nabla\Lambda(\mathbf{p}_n)$$

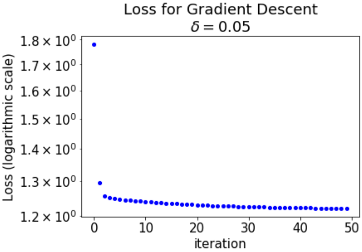
At minimum $\mathbf{p}^*$,

$$D\Lambda(\mathbf{p}^*) = 0 \Leftrightarrow \nabla\Lambda(\mathbf{p}^*) = 0 \Leftrightarrow \text{Euler conditions}$$

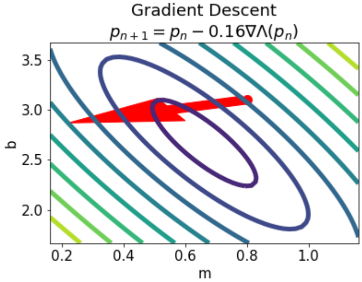
(if not at extremal, $\nabla\Lambda(\mathbf{p}) \neq 0$).

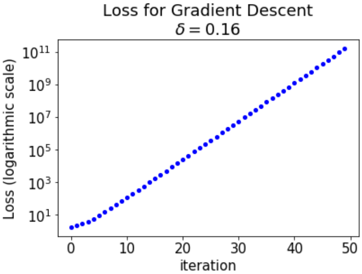
Converges for $\delta \ll 1$ (but perhaps slowly)





Diverges for $\delta \gg 1$ (overshoots)





More advanced algorithms modify gradient descent to improve performance. Python scipy.optimize.minimize has

- Nelder-Mead
- Conjugate-gradient
- Broyden, Fletcher, Goldfarb, and Shanno (BFGS)
- Newton-Conjugate-Gradient
- dog-leg trust-region
- Newton conjugate gradient trust-region
- Newton Generalized Lanczos trust-region

ILLINOIS

illinois.edu