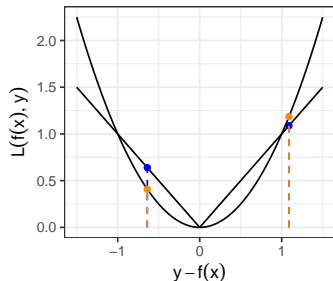


Introduction to Machine Learning

Supervised Regression: Linear Models with L_1 Loss

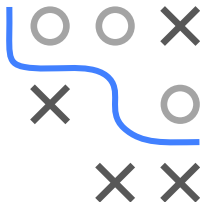
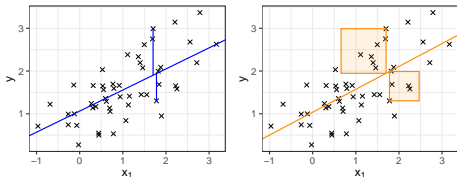


Learning goals

- Understand difference between L_1 and L_2 regression
- See how choice of loss affects optimization & robustness

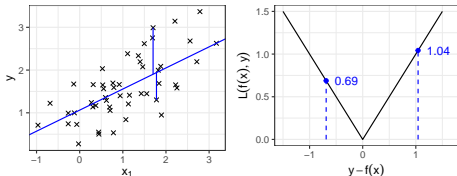
ABSOLUTE LOSS

- L_2 regression minimizes quadratic residuals – wouldn't **absolute** residuals seem more natural?

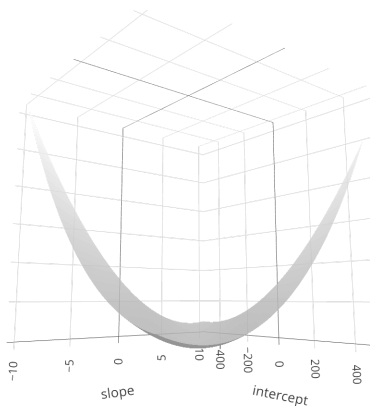
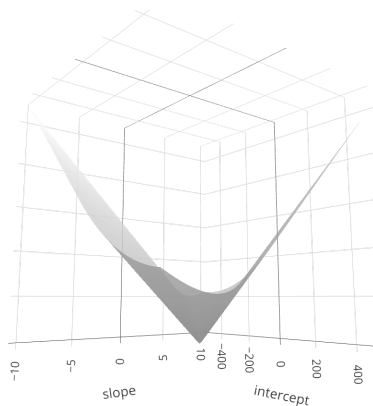


- L_1 loss / absolute error / least absolute deviation (LAD)

$$L(y, f(\mathbf{x})) = |y - f(\mathbf{x})|$$



L1 VS L2 – LOSS SURFACE



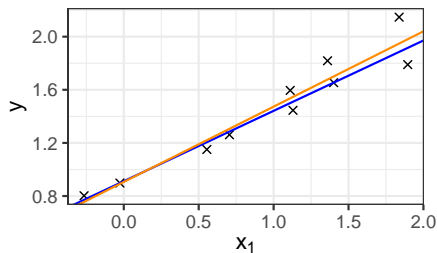
L1 loss (left) harder to optimize than L2 loss (right)

- Convex but **not differentiable** in $y - f(\mathbf{x}) = 0$
- No analytical solution

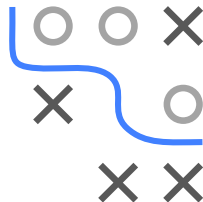
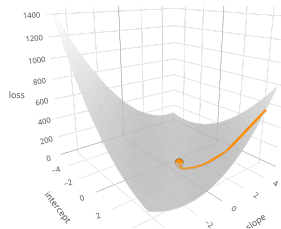
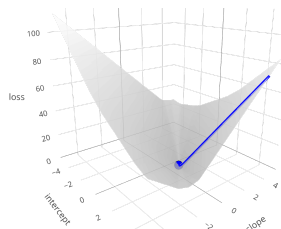
L1 VS L2 – ESTIMATED PARAMETERS

- Results of $L1$ and $L2$ regression often not that different
- Simulated data: $y^{(i)} = 1 + 0.5x_1^{(i)} + \epsilon^{(i)}$, $\epsilon^{(i)} \stackrel{i.i.d}{\sim} \mathcal{N}(0, 0.01)$

	intercept	slope
$L1$	0.91	0.53
$L2$	0.91	0.57

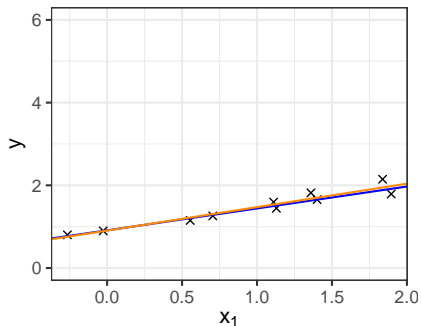


absolute quadratic

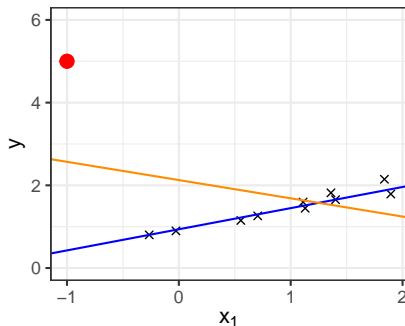


L1 VS L2 – ROBUSTNESS

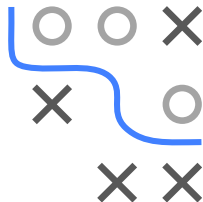
- L2 quadratic in residuals \rightsquigarrow outlying points carry lots of weight
- E.g., $3\times$ residual $\Rightarrow 9\times$ loss contribution
- L1 more **robust** in presence of outliers (example ctd.):



absolute quadratic

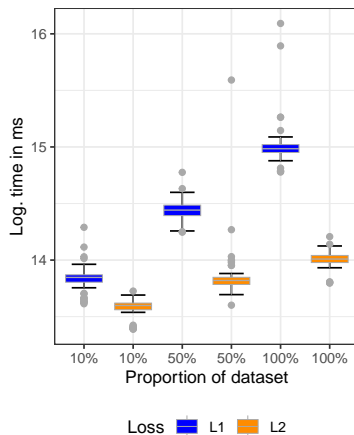
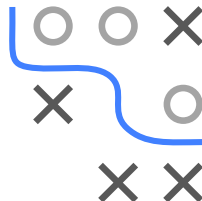


absolute quadratic



L1 VS L2 – OPTIMIZATION COST

- Real-world weather problem \rightsquigarrow predict mean temperature
- Compare **time** to fit L1 (`quantreg::rq()`) vs L2 (`lm::lm()`) for different dataset proportions (repeat 50 \times)



Loss

	Fitted: <i>L1</i>	Fitted: <i>L2</i>
Total <i>L1</i> loss	8.98×10^4	8.99×10^4
Total <i>L2</i> loss	5.83×10^6	5.81×10^6

Estimated coefficients

x_j	<i>L1</i> : $\hat{\theta}_j$	<i>L2</i> : $\hat{\theta}_j$
Max_temperature	0.553	0.563
Min_temperature	0.441	0.427
Visibility	0.026	0.041
Wind_speed	0.002	0.010
Max_wind_speed	-0.026	-0.039
(Intercept)	-0.380	-0.102

L1 slower to optimize!