CS 189/289

Today's lecture:

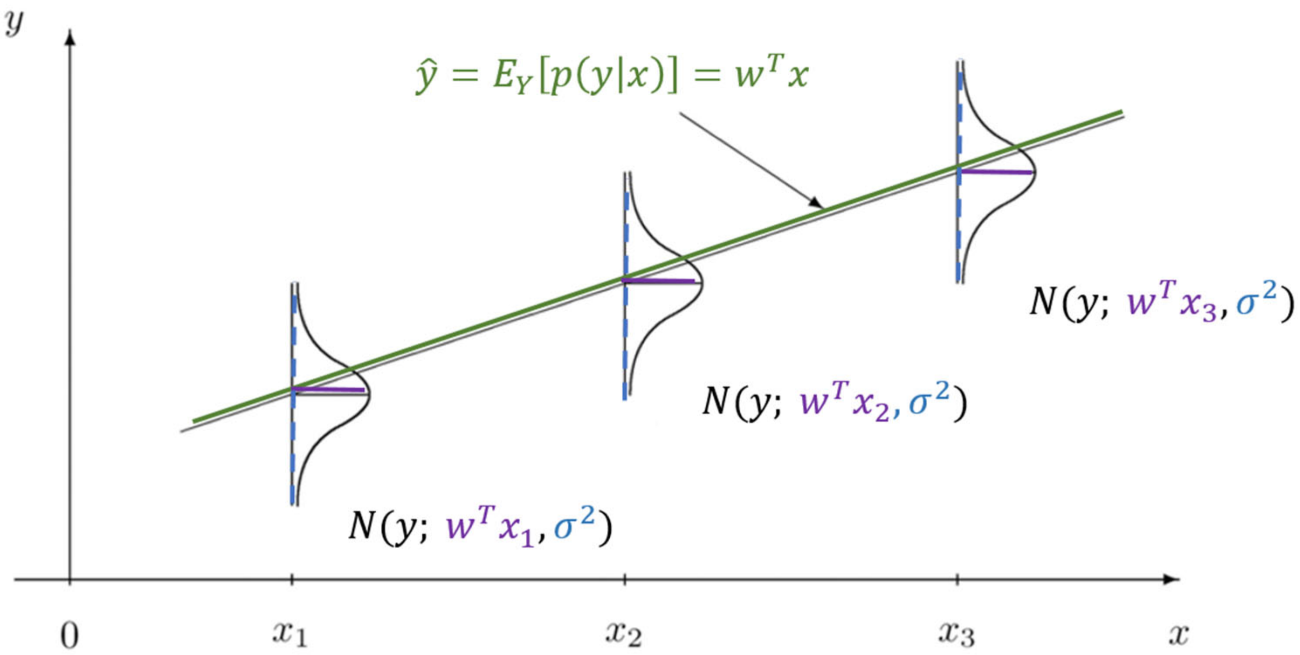
Linear regression part II (regularization)

Assigned reading:

- 1.25, 1.26 (regularization, model selection)
- 2.1.3 (Bayes Theorem)
- 2.6 (Bayesian modeling)
- 4.1.6 (Regularization for linear regression)
- 9.2.2 (Lasso regularization)

Reloading last lecture

Gaussian linear regression, $p(y|x) = N(y|w^T x, \sigma^2)$.



- $\theta_{MLE} = (w_{MLE}, \sigma^2_{MLE}) = \arg\max_{(w,\sigma^2)} \log p(D|\theta)$
- $\mathcal{L}_w = (y - Aw)^T (y - Aw)$, set $\frac{\partial \mathcal{L}}{\partial w} = 0$
- $A^T y = A^T A w \qquad (A \in \mathbb{R}^{N\times d})$
- $w_{MLE} = (A^T A)^{-1} A^T y$, $\sigma^2_{MLE} = \frac{1}{N}\sum_i (y_i - w^T x)^2$

careful!

- Not invertible if columns (features) in A are linearly dependent.
- Automatically happens when $d > N$, in which case, $y = Aw$ exactly.

- When not invertible, there are ∞ many equally good solutions for w_{MLE}.
- Called *underdetermined* linear regression.

$\begin{pmatrix} x_1^T \\ \vdots \\ x_n^T \end{pmatrix} = A \in \mathbb{R}^{n\times d}$

Fixing underdetermined models

Two main categories of fixes:

1. Remove features until the problem is well-behaved “feature selection”).
2. Leave the features as they are, but *add constraints to the system* to “tighten it up” (aka “regularization”).

Intuition of why ∞ # of $\boldsymbol{w_{MLE}}$ solutions

- Suppose we have 2 *linearly dependent features* in the training data such that $\alpha x_1 = x_2$.
- Suppose we found one MLE solution, $\widehat{\boldsymbol{w}}$.
- Then for any training data point, $\hat{y} = \boldsymbol{x}^T \widehat{\boldsymbol{w}}$.

$$= [x_1\ x_2] \begin{bmatrix} \widehat{w}_1 \\ \widehat{w}_2 \end{bmatrix}$$

$$= \widehat{w}_1 x_1 + \widehat{w}_2 x_2 = \widehat{w}_1 x_1 + \widehat{w}_2\, \alpha x_1$$

$$= (\widehat{w}_1 + \widehat{w}_2 \alpha) x_1 = (\widehat{w}_1 + \widehat{w}_2 \alpha + \beta - \beta) x_1 \text{ for any } \boldsymbol{\beta}.$$

$$= ((\widehat{w}_1 + \beta) + (\widehat{w}_2 \alpha - \beta)) x_1 \text{ for any } \boldsymbol{\beta}.$$

$$= (\widehat{w}_1 + \beta) x_1 + (\widehat{w}_2 \alpha - \beta) \frac{x_2}{\alpha} = (\widehat{w}_1 + \beta) x_1 + (\widehat{w}_2 \alpha - \beta) \frac{1}{\alpha} x_2$$

$$= [x_1\ x_2] \begin{bmatrix} \widehat{w}_1 + \beta \\ \frac{\widehat{w}_2 \alpha - \beta}{\alpha} \end{bmatrix} = \boldsymbol{x}^T \widetilde{\boldsymbol{w}} = \hat{y} \text{ for } \widetilde{\boldsymbol{w}} = \begin{bmatrix} \widehat{w}_1 + \beta \\ (\widehat{w}_2 \alpha - \beta)/\alpha \end{bmatrix}$$

Of all the $\{\boldsymbol{w_{MLE}}\}$ with zero error, is there one that intuitively might be generally be better?

Intuition for choosing one specific $\hat{\boldsymbol{w}}$.

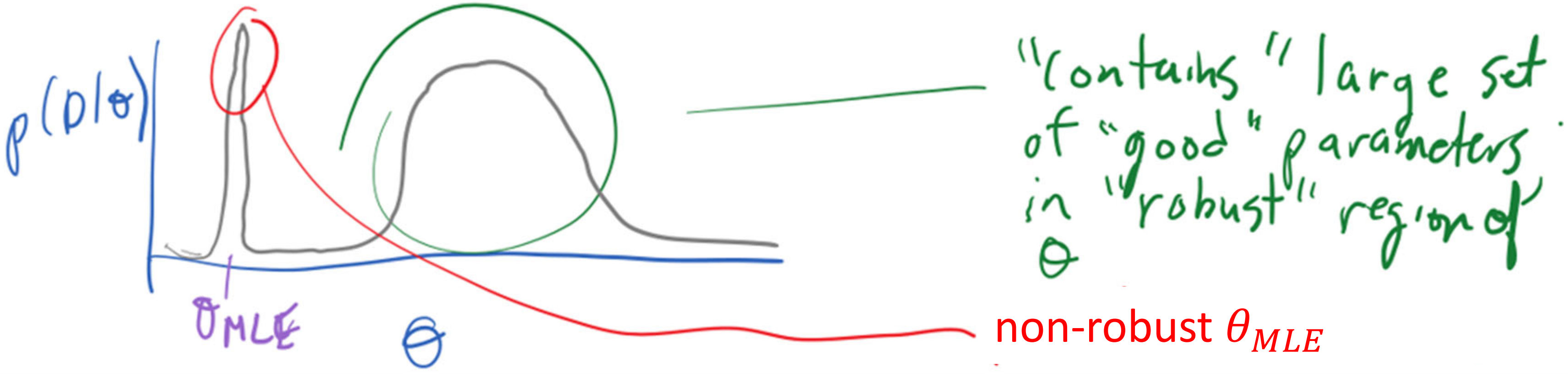
Of the ∞ solutions for $\boldsymbol{w_{MLE}}$, <u>choose the one with the least norm</u>, $\|\boldsymbol{w_{MLE}}\|_2$.
Why might this be a good idea? $\hat{y} = x^T\hat{w}$

- We would like each feature to have as little effect on the outcome as possible while still predicting well, so that model behaves "gracefully".
- Consider prediction, $\hat{y} = \boldsymbol{w}^T\boldsymbol{x}$. How much does the prediction change when we perturb, $\boldsymbol{x}' = \boldsymbol{x} + \boldsymbol{\delta}$, for different norm $\boldsymbol{w}$?
- With smaller coefficients the model is less sensitive to noisy data.
- What about for non-degenerate linear regression ($\boldsymbol{A}^T\boldsymbol{A}$ is invertible)?
- Yes! For <u>many</u> problems (and models), small param norm is a good idea.
- This is one e.g. of *regularization*: in effect, reduce # free parameters, while keeping the same set of parameters!

Recall how MLE can go wrong?

MLE yields a "point estimate" of our parameter

- When we perform MLE, we get just one single estimate of the parameter, $\boldsymbol{\theta}$, rather than a distribution over it which captures uncertainty.
- In Bayesian statistics, we obtain a (posterior) distribution over $\boldsymbol{\theta}$. We will touch more on this in a few lectures.



L2 regularized linear regression

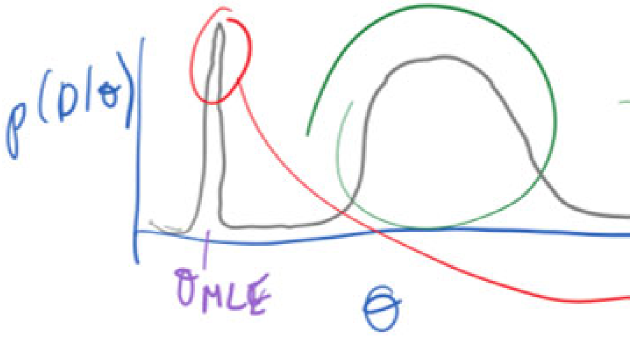
To *shrink* $\boldsymbol{w}$ to have smaller norm than the MLE solution, we add a "penalty" term to the loss function:

$$\mathcal{L} = (y - Aw)^T (y - Aw) + \lambda \|w\|_2{}^2$$
$$w_{L_2} = \underset{w}{\operatorname{argmin}}\ (y - Aw)^T (y - Aw) + \lambda \|w\|_2{}^2$$

Also called "Ridge" regression, or L2 linear regression.

Related to *Bayesian* modeling (next).

The Bayesian modelling approach



- Bayesians put a *prior distribution* on the parameters, $\boldsymbol{p(\theta)}$.
- Then they seek to compute the *posterior distribution*, $\boldsymbol{p(\theta|D)}$.
- Then, predictive distribution is given by

$$p(y|x) = \int_\theta p(y|x,\theta)p(\theta|D)d\theta = E_\theta[p(y|x,\theta)]$$

- Procedurally, this is done using Bayes' rule:

$$p(\theta|D) = \frac{p(D|\theta)p(\theta)}{p(D)}.$$

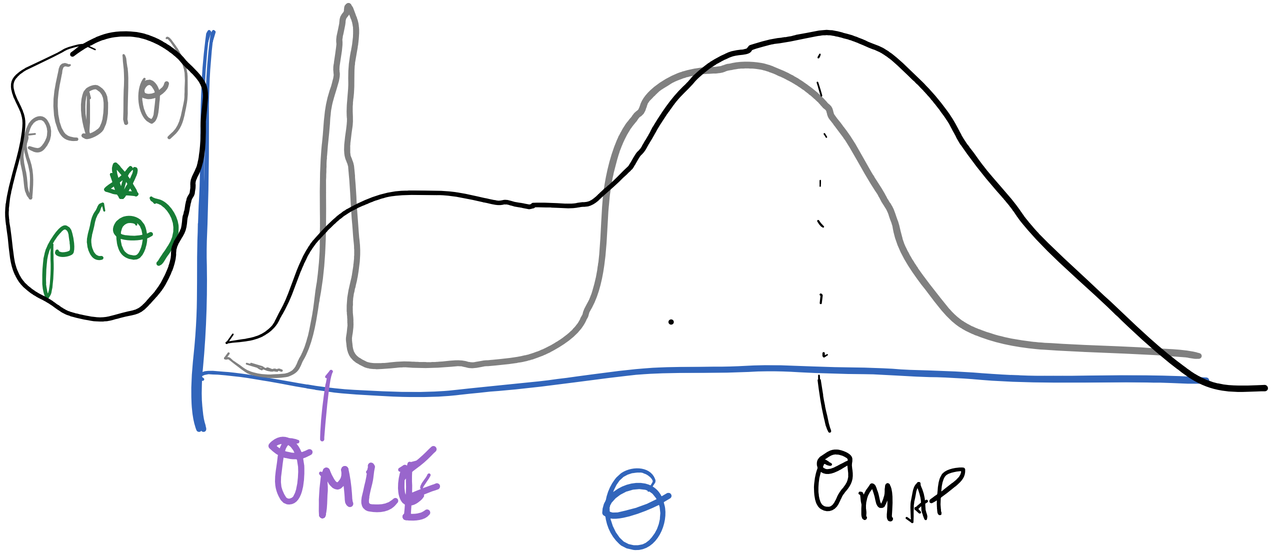
- Difficult in practice! $p(D) = \int_\theta p(D,\theta)d\theta = \int_\theta p(D|\theta)p(\theta)d\theta$
- We will be lazy, instead being pseudo Bayesians, yielding L2 regression:
- $\theta_{lazy} = \operatorname{argmax}_\theta p(\theta|D)$ *Maximum A Posteriori* (MAP) estimation.

MAP: the lazy Bayesian (*Maximum A Posteriori*)

- Still use a prior over parameters, $\boldsymbol{p(\theta)}$.
- Finds point estimate of the parameter that maximizes the posterior.
- $\theta_{MAP} = \underset{\theta}{\operatorname{argmax}}\, p(\theta|D)$

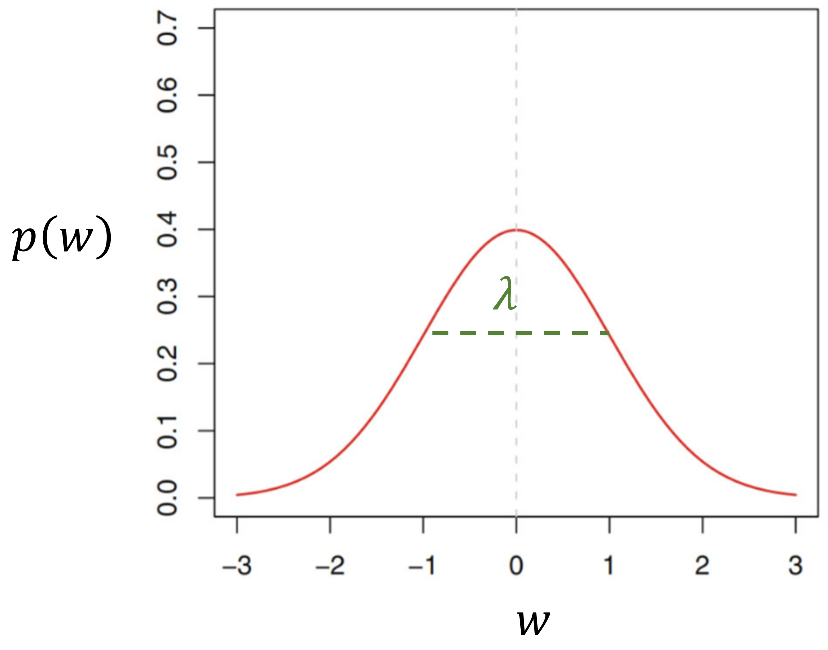
$$= \underset{\theta}{\operatorname{argmax}} \frac{p(D|\theta)p(\theta)}{\cancel{p(D)}}$$

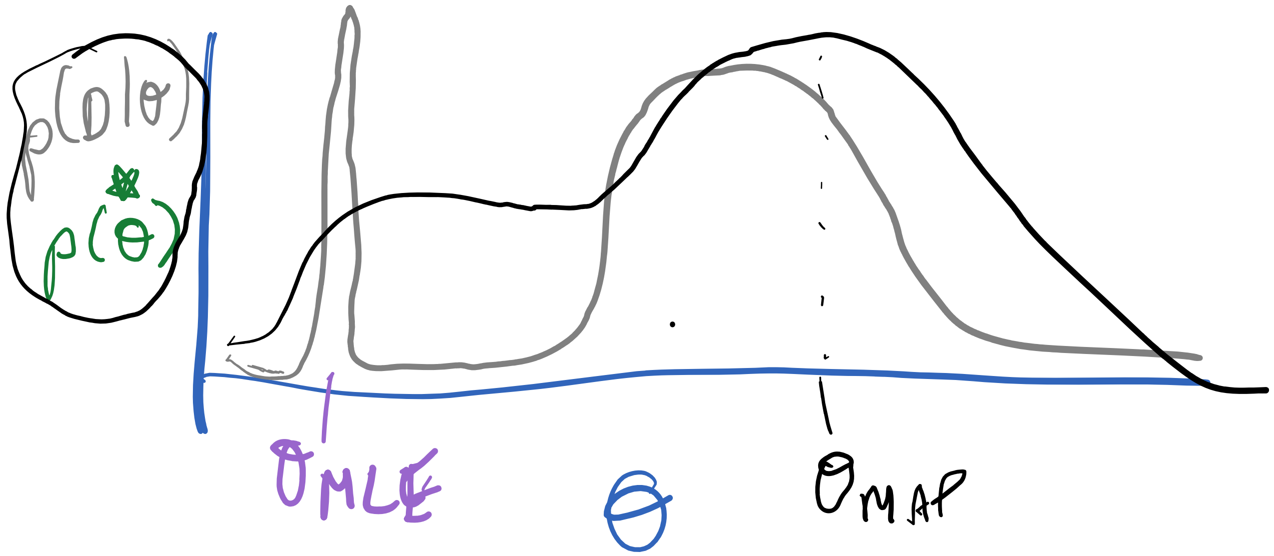
$$= \underset{\theta}{\operatorname{argmax}}\, p(D|\theta)p(\theta)$$



A prior for small weights yields L2 regression!

- Zero-mean prior, $p(w) = N(w; 0, \lambda I)$.
- Bayesian posterior, $p(w|D) = \dfrac{p(D|w)p(w)}{p(D)}$ is then "nice" in that everything is Gaussian (can work it out using MVGs).





MAP for linear regression w Gaussian prior

$$w_{MAP} = \underset{w}{\operatorname{argmax}} \log p(D|w)\, p(w) = \underset{w}{\operatorname{argmax}} \log p(D|w) + \log N(w; 0, \lambda I)$$

$$= \underset{w}{\operatorname{argmax}} \textstyle\sum_{i=1}^{N} \log N(y_i | w^T x_i, \sigma^2) + \log N(w | 0, \lambda I)$$

$$= \underset{w}{\operatorname{argmin}} + \frac{1}{2\sigma^2}(y - Aw)^T(y - Aw) - \textstyle\sum_{i=1}^{d} \log \left[\frac{1}{\sqrt{2\pi\lambda}} \exp\left(-\frac{(w_i - 0)^2}{2\lambda} \right) \right]$$

$$= \underset{w}{\operatorname{argmin}} \frac{1}{2\sigma^2}(y - Aw)^T(y - Aw) + \textstyle\sum_{i=1}^{d} \left[-\log \frac{1}{\sqrt{2\pi\lambda}} + \frac{w_i^2}{2\lambda} \right]$$

$$= \underset{w}{\operatorname{argmin}} \frac{1}{2\sigma^2}(y - Aw)^T(y - Aw) + \textstyle\sum_{d} \frac{1}{2\lambda} w_i^2$$

$$= \underset{w}{\operatorname{argmin}} \frac{1}{2\sigma^2}(y - Aw)^T(y - Aw) + \frac{1}{2\lambda} \|w\|_2{}^2$$

$$= \underset{w}{\operatorname{argmin}}\ (y - Aw)^T(y - Aw) + \cancel{2}\sigma^2 \frac{1}{\cancel{2}\lambda} \|w\|_2{}^2$$

$$= \underset{w}{\operatorname{argmin}} (y - Aw)^T(y - Aw) + \lambda' \|w\|_2{}^2 \quad \text{for } \lambda' = \frac{\sigma^2}{\lambda}.$$

Equivalence between MAP w Gaussian prior and L2 regression!

Obtaining the MAP/L2 solution

$$
\begin{aligned}
w_{L_2} &= \underset{w}{\operatorname{argmin}} \; (y - Aw)^T (y - Aw) + \lambda \|w\|_2{}^2 \\
&= \underset{w}{\operatorname{argmin}} \; (y - Aw)^T (y - Aw) + \lambda w^T w
\end{aligned}
$$

Take partial derivative and set to zero:

$$\nabla_w \mathcal{L}_{MAP} = -2A^T y + 2A^T A w + 2\lambda I w$$

$$\rightarrow \quad 0 = -A^T y + A^T A w + \lambda I w$$

$$\rightarrow A^T y = (A^T A + \lambda I) w$$

$$\rightarrow \left(A^T A + \lambda I\right)^{-1} A^T y = w$$

So $w_{L_2} = \left(A^T A + \lambda I\right)^{-1} A^T y$.

If $\lambda > 0$, we can invert $\left(A^T A + \lambda I\right)$.

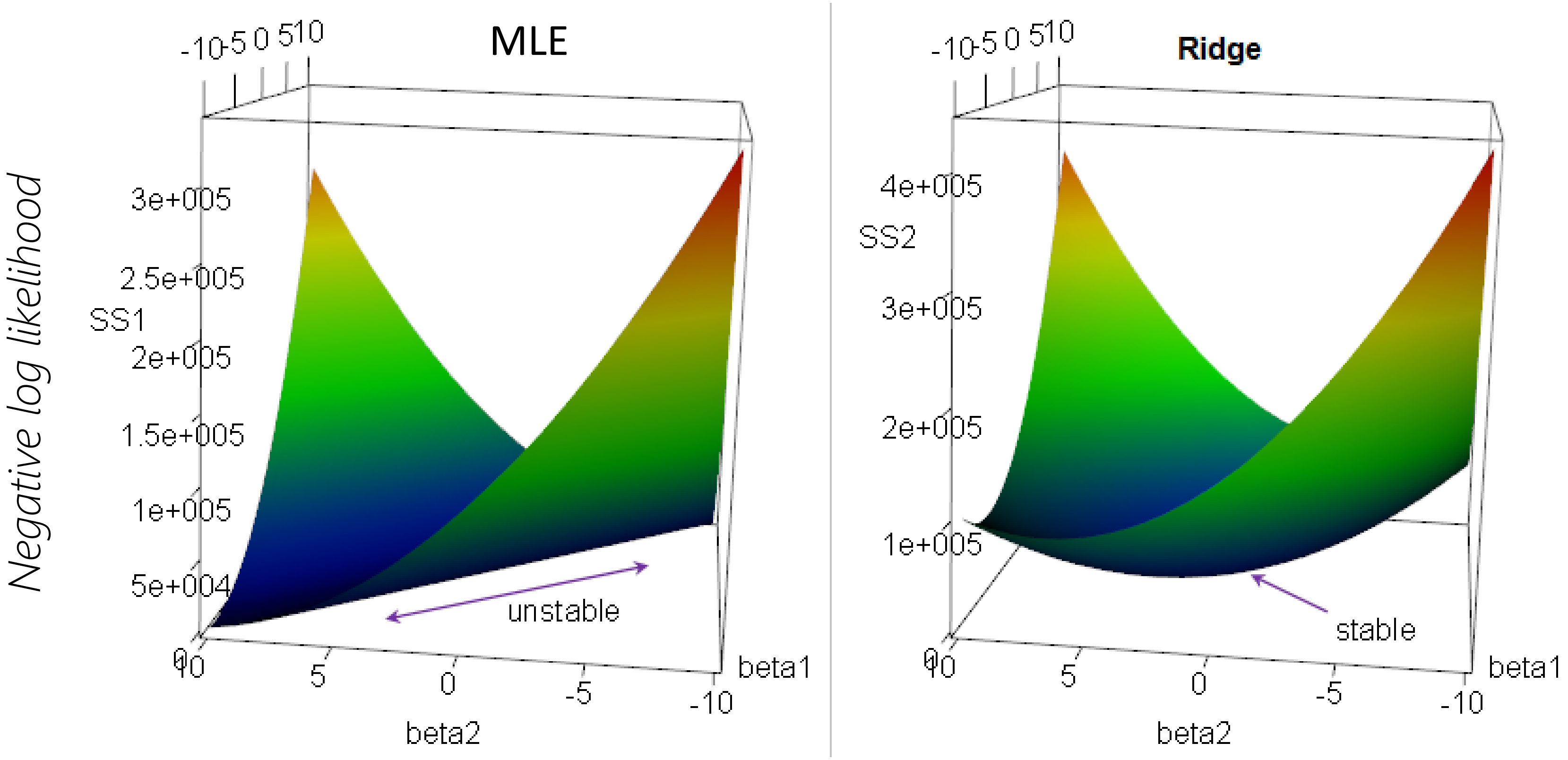
$A = \Phi D \Phi^T$

$A^{-1} = \Phi D^{-1} \Phi^T$

where $D^{-1} = \begin{bmatrix} \lambda_1^{-1} & & & \\ & \lambda_2^{-1} & & \\ & & \ddots & \\ & & & \lambda_n^{-1} \end{bmatrix}$

Aside, why is this called “Ridge” regression?

When some features are linearly dependent (can’t invert $A^T A$), we have ∞ many equally good solutions that form a ridge.

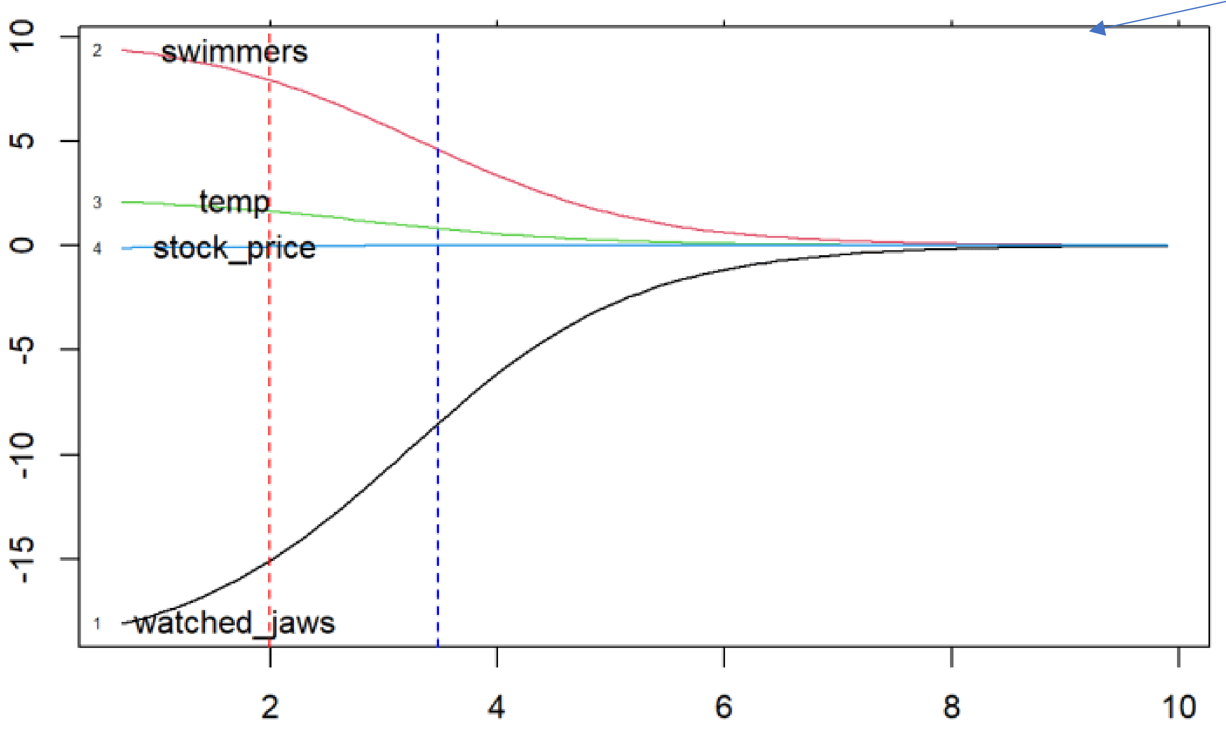


Effect of value of λ

$$\mathcal{L}_{MAP} = (y - Aw)^T(y - Aw) + \lambda \|w\|_2{}^2$$

# of non-zero features

w



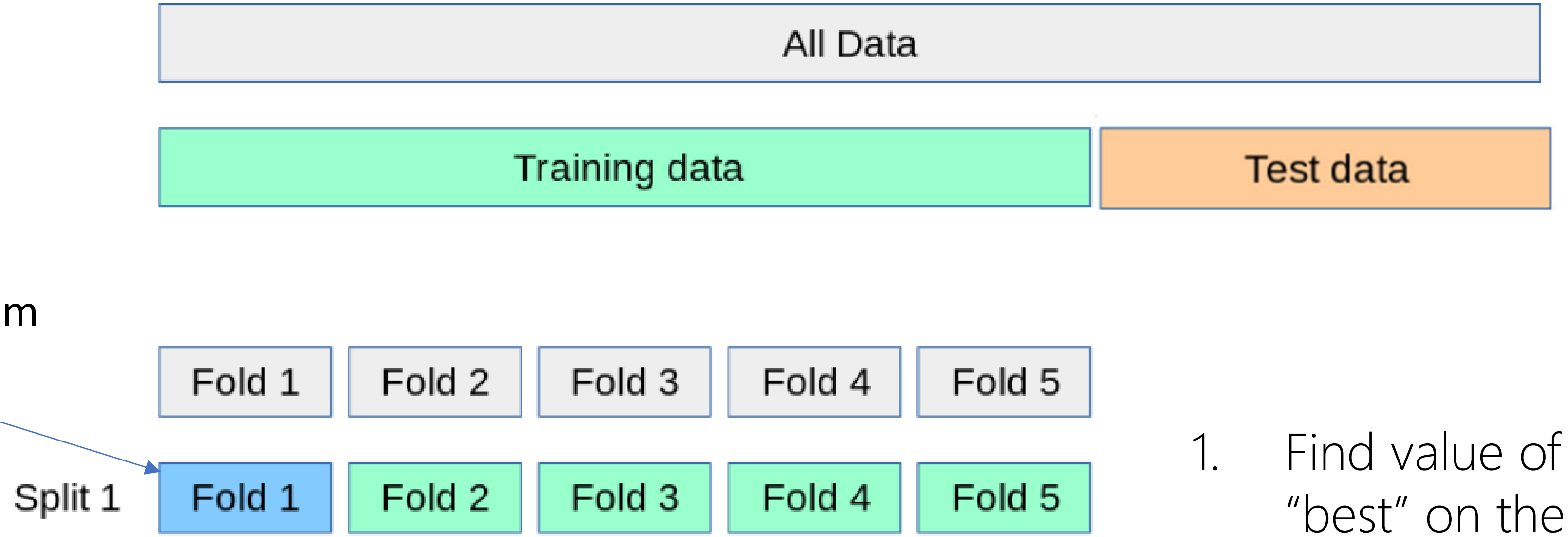
$\log \lambda$

https://www.r-bloggers.com/2020/06/un...g-lasso-and-ridge-regression/

- Practically, how should we set λ?
- Can we treat it as a parameter in the loss, and minimize wrt it?
- No: cannot use MLE!
- Need independent data, a *validation set* on which to evaluate the loss.

Train/validation/test split

Validation set from the training data



1. Find value of hyperparameter that is "best" on the validation set.
2. <u>Quantify</u> performance on the test data.

"Best"/quantify require computing the log likelihood on the held out data, and this is our metric.

https://scikit-learn.org/stable/modules/cross_validation.html

K-fold cross-validation

1. Find value of hyperparameter that is "best" on the validation set.
2. Quantify performance on the test data.

"Best"/quantify require computing the log likelihood on the held out data, and this is our metric.

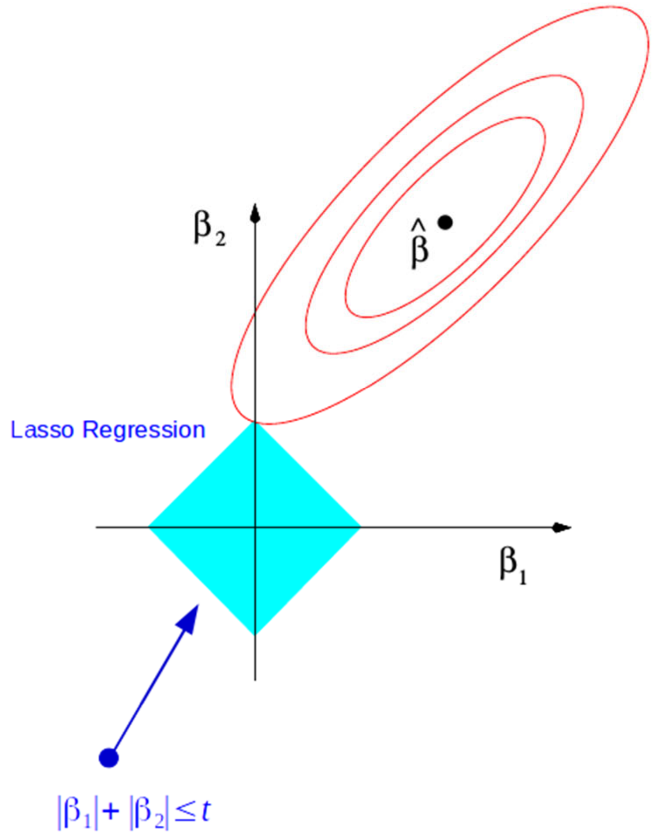
https://scikit-learn.org/stable/modules/cross_validation.html

Other priors for MAP in linear regression?

- What if wanted to have the linear function only depend on a few features (*i.e.*, the coefficients of $\boldsymbol{w}$ are sparse)?
- Add a penalty that counts the # of non-zero weights—an $\boldsymbol{L_0}$ penalty, $\lambda||w||_{\mathbf{0}}$.
- Not differentiable! (becomes combinatorial optimization—nasty).
- But…the $\boldsymbol{L_1}$ norm penalty, $\lambda||w||_{\mathbf{1}} = \lambda \sum_d |w_d|$, tends to induce sparse $\boldsymbol{w}$—differentiable everywhere except at $\boldsymbol{w_i} = \mathbf{0}$ which can be worked around.
- This is called *Lasso* regression, or $\boldsymbol{L_1}$-penalized regression.

L_1-penalized linear regression, aka *Lasso*

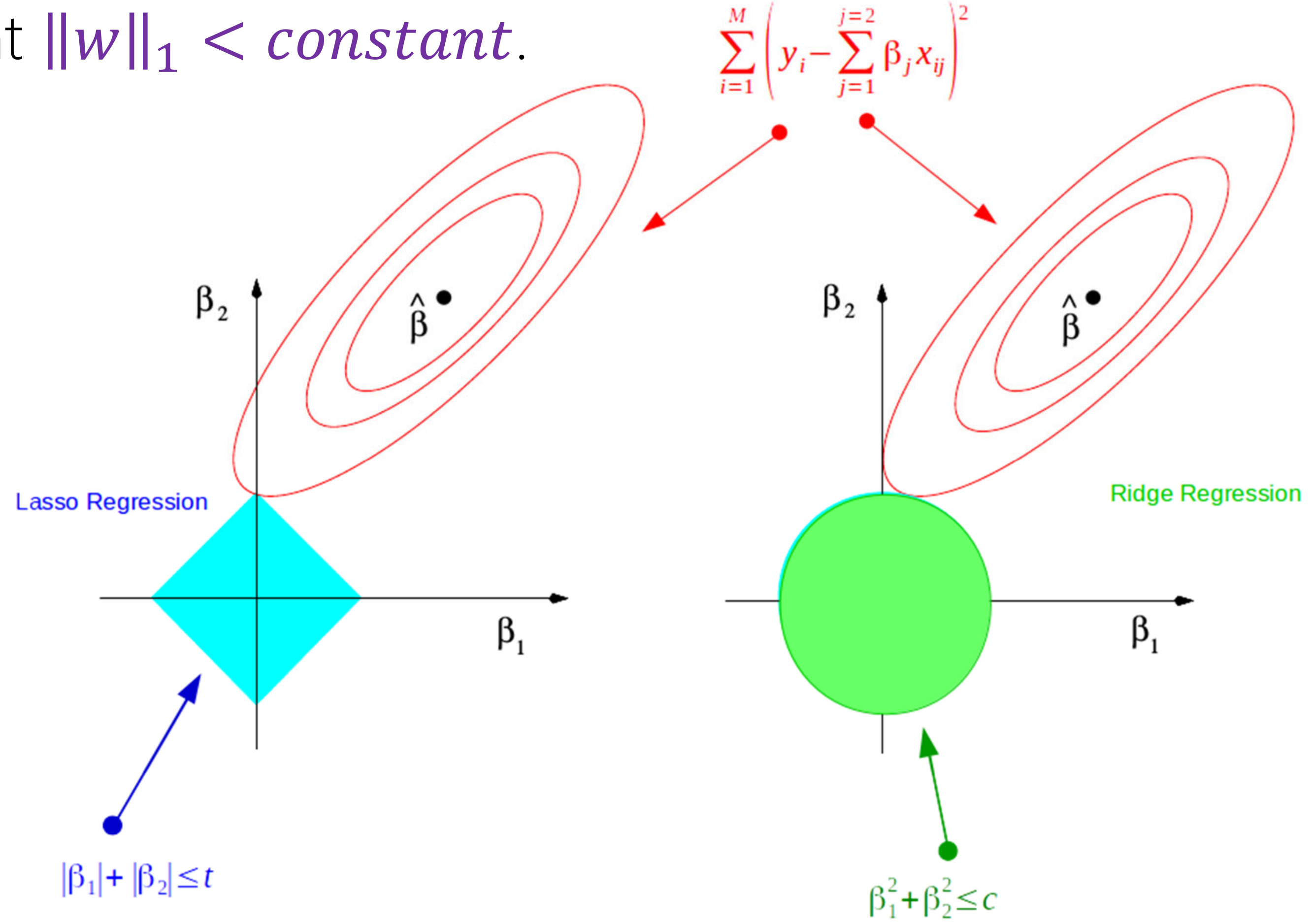
- $w_{L_1} = \underset{w}{\operatorname{argmin}} \ (y - Aw)^T(y - Aw) + \lambda\|w\|_1$
- Why does the L_1 norm penalty tends to induce sparse w?
- Equivalent to MLE with constraint $\|w\|_1 < constant$.
- "Pointy" constraint surface is jutting out along the axes.
- In many cases, the L_1 norm constraint will cause the unconstrained solution to intersect the constraint at a corner.
- The corners are where some coefficients are 0, which is a sparse solution..



https://towardsdatascience.com/ridge-and-lasso-regression-a-complete-guide-with-python-scikit-learn-e20e34bcbf0b

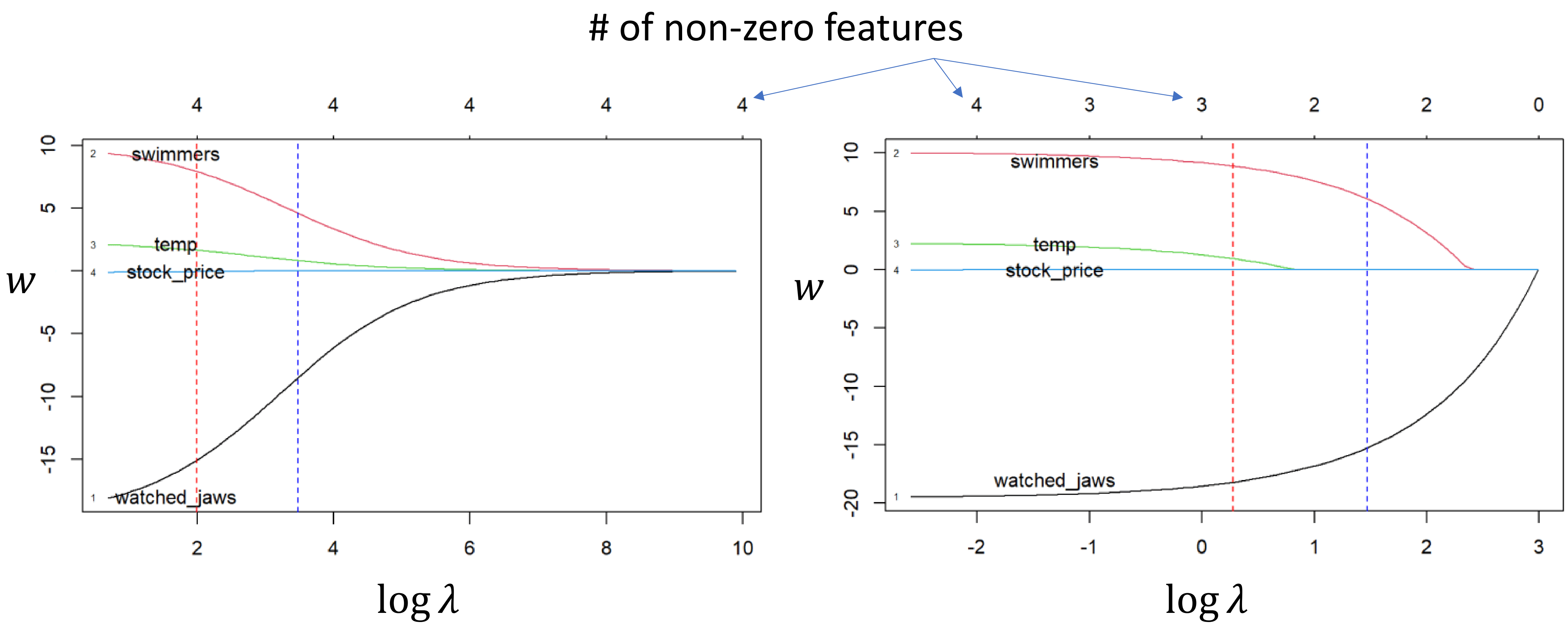
L_1-penalized linear regression, aka *Lasso*

- $w_{L_1} = \underset{w}{\operatorname{argmin}}\ (y - Aw)^T(y - Aw) + \lambda\|w\|_1$
- Why does the L_1 norm penalty tends to induce sparse w?
- Equivalent to MLE with constraint $\|w\|_1 < constant$.
- "Pointy" constraint surface is jutting out along the axes.
- In many cases, the L_1 norm constraint will cause the unconstrained solution to intersect the constraint at a corner.
- The corners are where some coefficients are 0, which is a sparse solution..



https://towardsdatascience.com/ridge-and-lasso-regression-a-complete-guide-with-python-scikit-learn-e20e34bcbf0b

Ridge vs Lasso: shrinkage vs sparsity

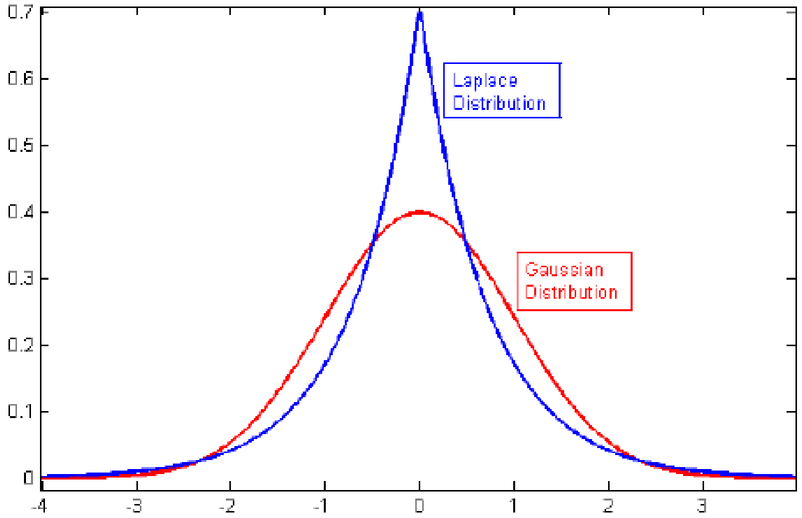


https://www.r-bloggers.com/2020/06/understanding-lasso-and-ridge-regression/

MAP interpretation for Lasso/L_1-penalized linear regression?

- L_2 regression arose from a $N(0, \lambda I)$ prior.
- Is there a prior corresponding to L_1?
- Yes, the Laplace prior!

$$p(w) = \exp(-\lambda' \|w\|_1).$$



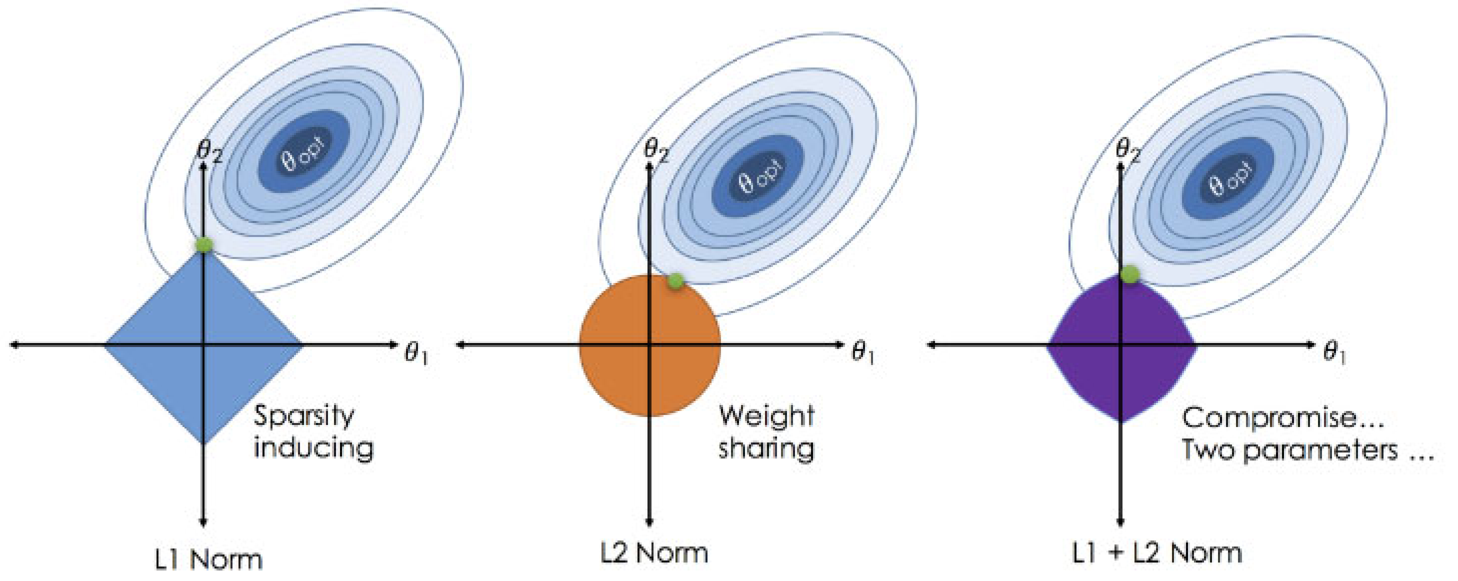
Issues with LASSO:

- If highly correlated features, tends to ignore all but one.

https://www.researchgate.net/figure/Comparison-of-Gaussian-and-Laplace-distributions_fig3_237843019

Combine L_1 and L_2 penalties?

Yes, “elastic net regression”.



https://www.researchgate.net/figure/Comparison-of-Gaussian-and-Laplace-distributions_fig3_237843019