Introduction to Data Science

REGULARIZATION

BRIAN D'ALESSANDRO

IMPLICIT FEATURE SELECTION

Regularization can be thought of as an implicit feature selection tool.

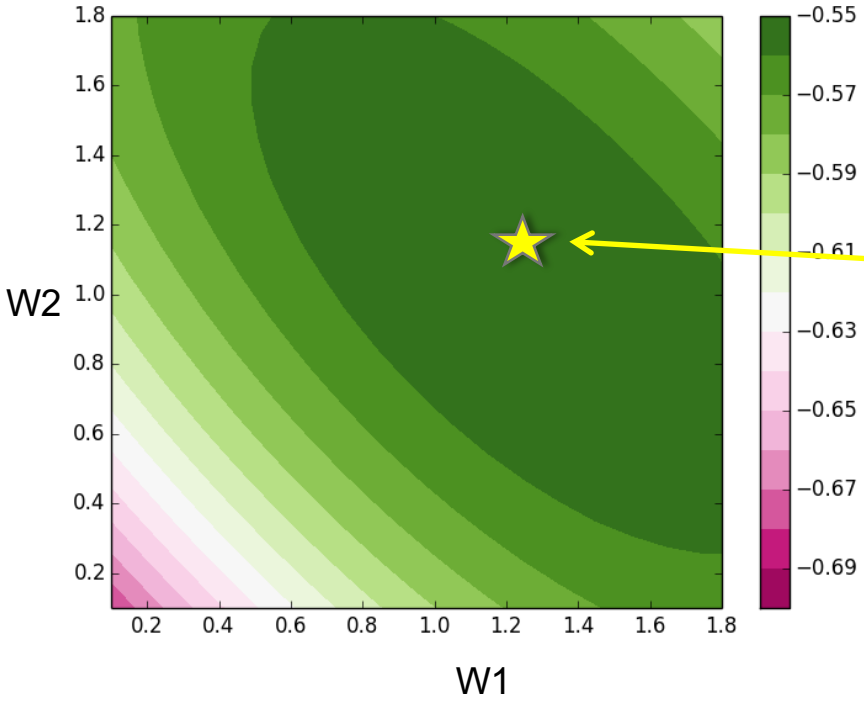
We control the complexity of the model by restricting how large the feature weights can grow.

Induces model bias as a means to reduce expected model variance.

Most useful when: n is small, dimensionality is high

OPTIMIZATION REVISITED

In standard ERM we look for the feature weights that minimize our loss function. We generally don't put any limits on what values the weights can take.



Objective

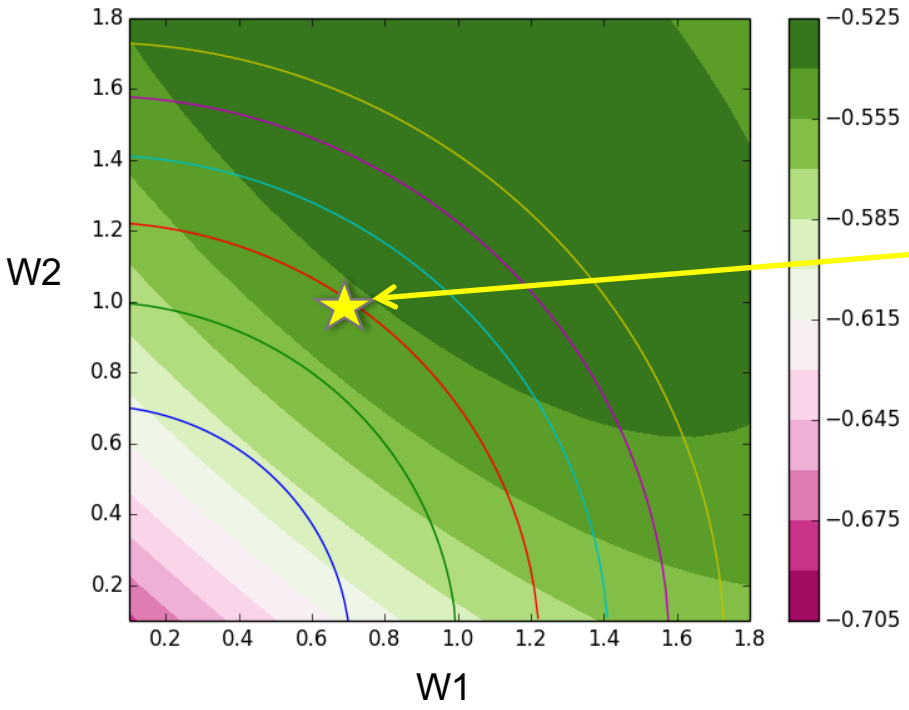
$$f^{opt} = \underset{f \in \mathbb{F}}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} \mathbb{L}(f(x_i), y_i)$$

Solution

Use some convex optimization procedure to numerically solve.

CONSTRAINING THE SOLUTION

We're going to use the same objective function, but we're going to add a constraint. We specify that the norm of the weights can't grow beyond a certain value. Usually R(W) is a convex function, such as L2 or L1 norm of W.



Objective

$$f^{opt} = \underset{f \in \mathbb{F}}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} \mathbb{L}(f(x_i), y_i)$$

subject to $R(W) \leq t$

Solution

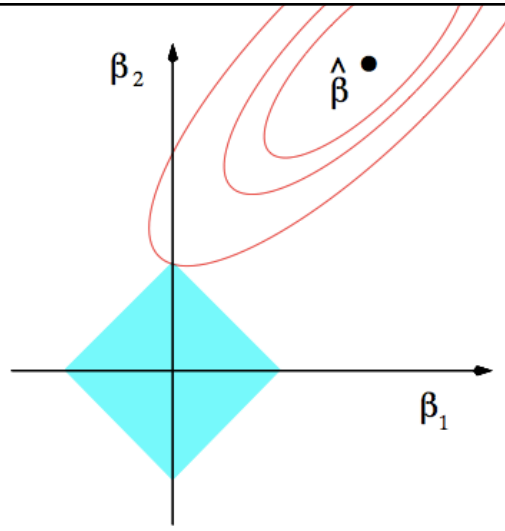
Now we pick the max value that lies within the constraint circle.

L1 AND L2 REGULARIZATION

We can take the Lagrange form of the constrained optimization problem, and set up two new objective functions.

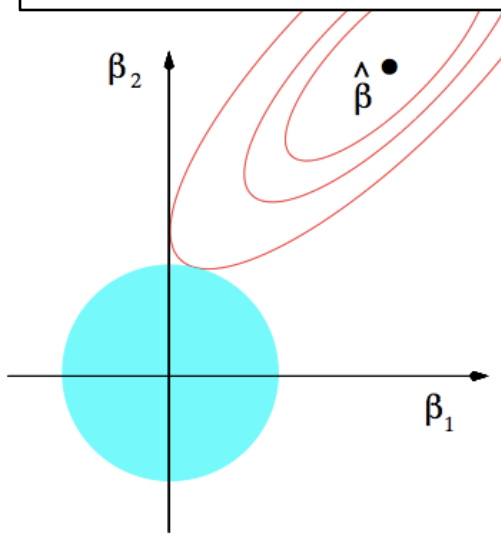
L1 (Lasso)

$$f^{opt} = \underset{f \in \mathbb{F}}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} \mathbb{L}(f(x_i), y_i) + \lambda \sum_{j=1}^{m} |W_j|$$



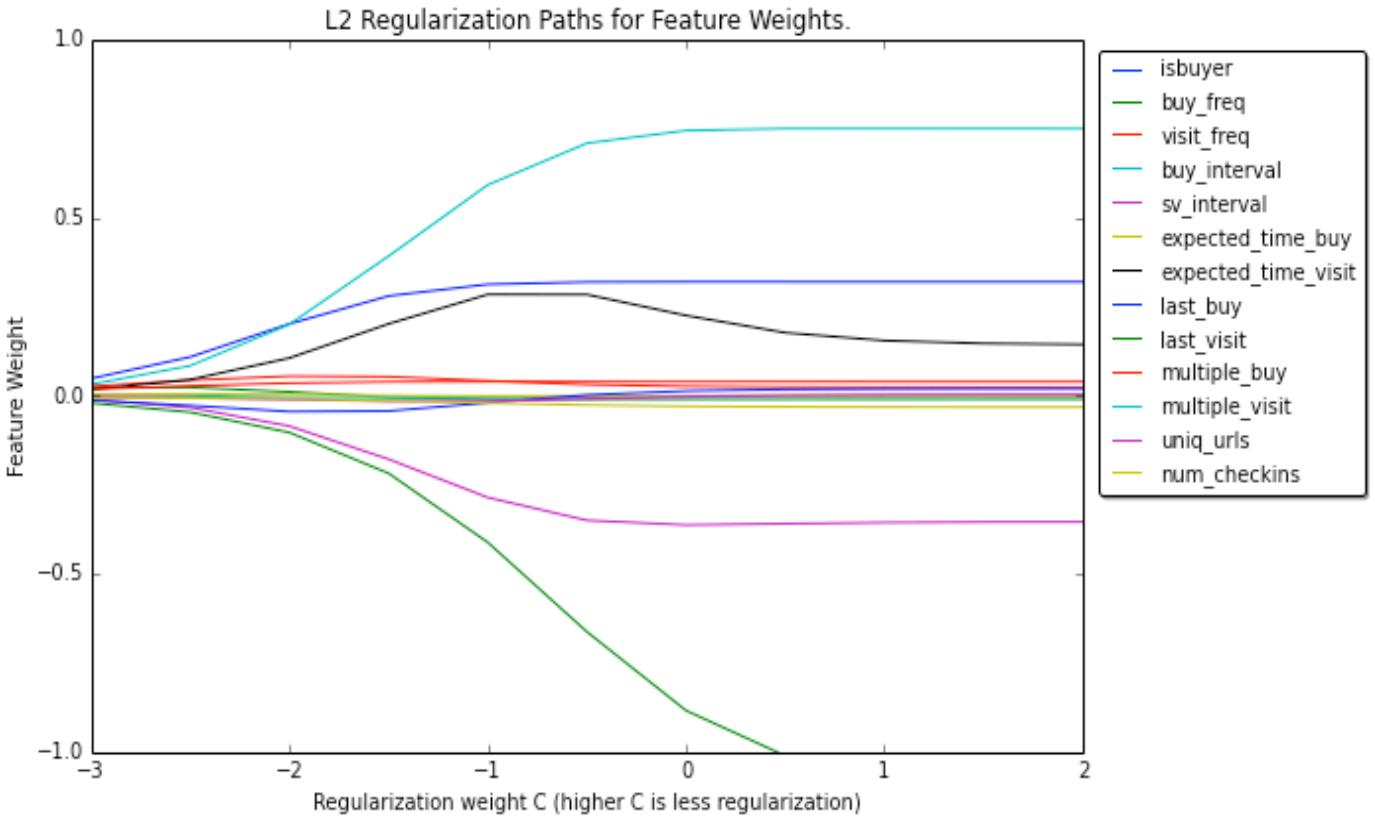
L2 (Ridge)

$$f^{opt} = \underset{f \in \mathbb{F}}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} \mathbb{L}(f(x_i), y_i) + \lambda \sum_{j=1}^{m} W_j^2$$



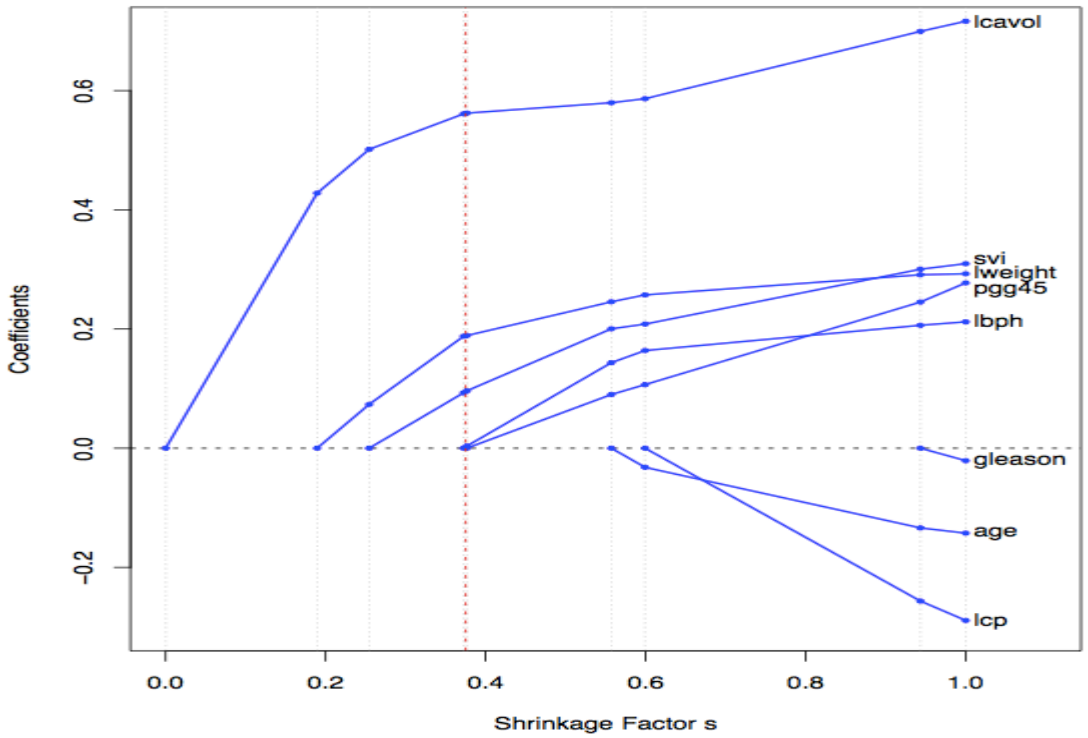
REGULARIZATION PATHS

We can visualize what happens to the feature weights as we change the regularization strength.



AN L1 EXAMPLE

This is an example from ESL, we can see here how coefficients are frozen at 0 before they grow. This is a good example of how regularization is a form of implicit feature selection.



WHY DOES IT WORK?

Regularization is a tool to reduce a model's complexity, where 'complexity' represents how sensitive a model is to small levels of noise in the data.

Take a standard linear model: $F(X) = W^*X + b$

Define: $X^* = X + \varepsilon$, s.t. $|\varepsilon|_2$ is reasonably small

Since X and X^* are reasonably close, we hope that $F(X)$ and $F(X^*)$ are reasonably close. We can measure this by:

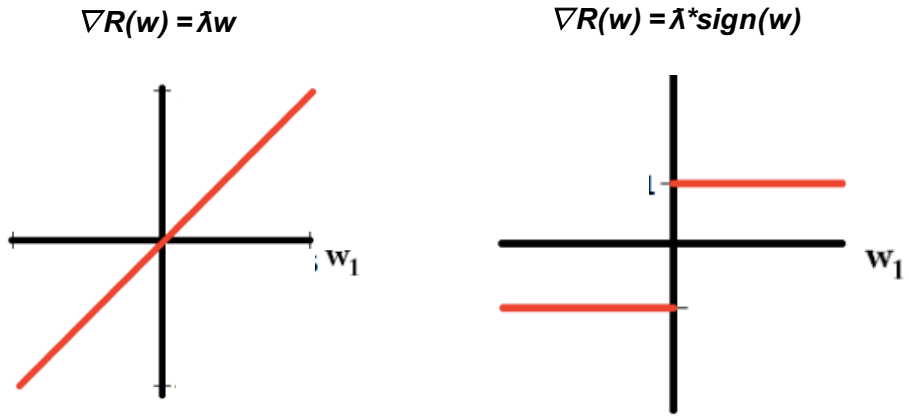
$$|F(X) - F(X^*)| = |W^*X - W^*X^*| = |W^*(X-X^*)| = |W^* \varepsilon| <= |W|_2 * |\varepsilon|_2$$

So by bounding $|W|_2$ we can limit how much small perturbations of X changes our prediction.

In other words, regularization ensures that the nearest neighbors of X receive a similar prediction as X.

WHY DOES L1 LEAD TO SPARSITY?

To understand L1 & L2 regularization better, let's start with our total Cost function gradient: ∇ **Cost =** ∇***Loss(y,*** $\hat{y}$**)** **+** ∇***R(w)***. Assuming ***w=0*** to start, in order for ***w+δ*** to move us closer to the minimum of our convex cost function we need for ***δ*** * ∇ **Cost < 0.**



Let's assume that moving in the direction of ***δ*** improves our loss (otherwise, why we would even consider moving), which means ***δ*** * ∇ ***Loss(y,*** $\hat{y}$**) < 0.** Putting these together leads to the condition that in order to justify moving our weight from 0, we need for |∇***Loss(y,*** $\hat{y}$**)|> |** ∇***R(w)***|. With L1 there is a constant cost (***λ***)** whereas for for L2 our cost is proportional to ***w (λw).*** When ***w*** is small and close to zero, the L1 penalty is usually much higher, making it harder to meet the condition that |∇***Loss(y,*** $\hat{y}$**)|> |** ∇***R(w)***|. If the condition isn't met then it is more optimal to keep ***w=0***. Thus, L1 will keep the weight at 0. To escape 0 we need to either reduce ***λ*** or receive so much benefit that the benefit exceeds the loss.

THE BAYESIAN INTERPRETATION

We can think of regularization as a
Maximum a Posteriori estimation problem.

Using Bayes rule, the posterior of β is:

$$P(\beta|X,Y) \propto P(X,Y|\beta) * P(\beta) = Likelihood * Prior$$

The MAP estimate of β is the value that maximizes the posterior distribution.

$$\hat{\beta}_{MAP} = \underset{\beta}{\text{argmax}}\ L(\beta|X,Y)\ P(\beta)$$

BAYESIAN LOGISTIC REGRESSION

For L2 regularization, we assume that P(β) is drawn from a normal distribution with 0 mean and variance τ.

$$P(\beta_j) = N(0, \tau_j) = \frac{1}{\sqrt{2\pi\tau_j}} exp(\frac{-\beta_j^2}{2\tau_j}), \quad j = 1, 2, \ldots d$$

The posterior of logistic regression feature weights can then be defined as:

$$\hat{\beta}_{MAP} = \underset{\beta}{\text{argmax}} \prod_{i=1}^{n} p_i^{y_i}(1 - p_i)^{1-y_i} \prod_{j=1}^{d} \frac{1}{\sqrt{2\pi\tau_j}} exp(\frac{-\beta_j^2}{2\tau_j})$$

If we take the negative log of the above posterior, we end up with the ERM form of logistic regression.

(See iPython notebook for similar treatment of L1).

REGULARIZATION WEIGHT

Both the regularization weight and style of regularization are hyper-parameters. So we can use standard model selection methodologies (i.e., cross-validation) to choose the optimal regularization strength.

We can use the same method for L1 vs L2, but sometimes we choose L1 on principal alone. Its often desired to have the implicit feature selection.