Regularization for Regression: Ridge & Lasso

Mehrdad Mahdavi  
CMPSC 448  
Spring 2023
Why should I regularize?

Consider the following dataset and 3 linear regression models:

Which line should we choose?
Why should I regularize?

Consider the following dataset and 3 linear regression models:

What if you are forced to choose between RED and GREEN?
Key idea of regularization:

- **Red** line is much more sensitive to this feature, we should pick **green**.
- If we don’t get the slope right, **red** line causes more harm than **green**.
In essence, for functions which are too “wiggly”, the chance that the model will overfit the data is high which results in high prediction (test) error!
We need to impose some constraints on the hypothesis space when training:

\[
\arg \min_{\mathbf{w} \in \mathbb{R}^d} \frac{1}{2} \sum_{i=1}^{n} (\mathbf{w}^\top \mathbf{x}_i - y_i)^2
\]

\[\Omega(\mathbf{w}) \leq r\]

Each \( r \) corresponds to a different hypothesis spaces.

This leads to a constrained optimization problem for fitting/training.
Complexity measures

For different function classes (hypothesis spaces) we have different complexity measures:

- Depth of a decision tree
- Number of neurons in a neural network
- Degree of polynomial
- ...

How about for linear decision functions as we used in linear regression (OLS)?
Regularization for linear regression

**Intuition**: large slopes tend to lead to overfitting.

Penalize the coefficients $w_1, w_2, \ldots, w_d$ if they take large values!

$$\arg \min_{w \in \mathbb{R}^d} \frac{1}{2} \sum_{i=1}^{n} (w^\top x_i - y_i)^2$$

such that: some function of coefficients $\leq \mu$
Choice for function of coefficients

such that: some function of coefficients $\leq \mu$

Ridge regression: OLS + L2 regularization ($\ell_2$ norm of coefficients)

Lasso: OLS + L1 regularization ($\ell_1$ norm of coefficients)
Duplicate features

- Suppose we have two features with same values: \( x_1, x_2 \in \mathbb{R} \) and \( x_1 = x_2 \)

- Consider linear prediction functions: \( f(w_1, w_2) = w_1 x_1 + w_2 x_2 \)

- Recall in this setting all the functions with \( w_1 + w_2 = c \) is a solution!

- OLS can select any of these functions as final solution!

- What function will we select if we do \( \ell_1 \) or \( \ell_2 \) regularization?
## Duplicate features: Ridge vs Lasso

- Assume \( w_1 + w_2 = 4 \)
- Consider the \( \ell_1 \) and \( \ell_2 \) norm of various solutions:

| \( w_1 \) | \( w_2 \) | \( |w_1| + |w_2| \) | \( w_1^2 + w_2^2 \) |
|---|---|---|---|
| 4  | 0  | 4  | 16 |
| 2  | 2  | 4  | 8  |
| 1  | 3  | 4  | 10 |
| -1 | 5  | 6  | 26 |

\( \ell_1 \) does not discriminate, as long as all have same sign

\( \ell_2 \) minimized when weight is spread equally
OLS has infinitely many solutions (all these solutions lead to same training error)!

\[ w_1 + w_2 = c \]
Duplicate features

Training error increase as we move away from OLS parameter settings

$\|w\|_2 \leq 1$

$l_2$ minimized when weight is spread equally
Duplicate features

\[ \ell_1 \text{ spreads weight arbitrarily (all weights same sign) as long as all have same sign} \]
Equivalent formulation

- Penalize the coefficients $w_1, w_2, \ldots, w_d$ if they take large values!

- Change the training goal: minimize squared error plus penalty on coefficients

$$\arg\min_{\mathbf{w} \in \mathbb{R}^d} \frac{1}{2} \sum_{i=1}^{n} (\mathbf{w}^\top \mathbf{x}_i - y_i)^2 + \lambda \text{(penalization of coefficients)}$$

- OLS objective (training error)
- Regularization function (constraint on coefficients)
- Regularization parameter

- Basically the unconstrained version of the first formulation!

- For any choice of $\mu$ there is a unique value for $\lambda$ such that both formulations result in the same optimal solution! Can get conditions for equivalence from Lagrangian duality theory!

- The standard formulation you usually see is the unconstrained version, but there is no difference (we will use the constrained version for illustration purposes only).
Regularized training/fitting

- Penalize the coefficients $w_1, w_2, \ldots, w_d$ if they take large values!

- Change the objective function minimize squared error plus penalty on coefficients

$$\arg \min_{w \in \mathbb{R}^d} \frac{1}{2} \sum_{i=1}^{n} \left( w^\top x_i - y_i \right)^2 + \lambda \text{ (penalization of coefficients)}$$

- This objective makes a **tradeoff** between low training error (fit to data) vs. having small **weights (parameters)**

- Nearly-always reduces overfitting

- Regularization parameter $\lambda > 0$ controls “**strength**” of regularization.

  Large $\lambda$ puts large penalty on slopes.
The Ridge regression is a regularization technique that uses L2 regularization (length of vector of parameters) to impose a penalty on the size of coefficients.

\[
\arg \min_{\mathbf{w} \in \mathbb{R}^d} \frac{1}{2} \sum_{i=1}^{n} (\mathbf{w}^\top \mathbf{x}_i - y_i)^2 + \lambda \| \mathbf{w} \|_2^2
\]

OR

\[
\arg \min_{\mathbf{w} \in \mathbb{R}^d} \frac{1}{2} \sum_{i=1}^{n} (\mathbf{w}^\top \mathbf{x}_i - y_i)^2
\]

such that \( \| \mathbf{w} \|_2^2 \leq \mu \)
Ridge solution

We would like to find the optimal solution of following convex problem:

$$\mathbf{w}_{RR}^* = \arg \min_{\mathbf{w} \in \mathbb{R}^d} \frac{1}{2} \sum_{i=1}^{n} (\mathbf{w}^\top \mathbf{x}_i - y_i)^2 + \lambda \| \mathbf{w} \|_2^2$$

Let’s take the derivative of function and set it to zero:

$$\frac{f_{\text{OLS}}(\mathbf{w})}{\partial \mathbf{w}} = \sum_{i=1}^{n} (\mathbf{w}^\top \mathbf{x}_i - y_i) \mathbf{x}_i + 2\lambda \mathbf{w}$$

$$= \sum_{i=1}^{n} (\mathbf{w}^\top \mathbf{x}_i) \mathbf{x}_i - \sum_{i=1}^{n} y_i \mathbf{x}_i + 2\lambda \mathbf{w}$$

$$= \sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^\top \mathbf{w} - \sum_{I=1}^{n} y_i \mathbf{x}_i + 2\lambda \mathbf{w}$$

$$= \left( \sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^\top \right) \mathbf{w} - \sum_{i=1}^{n} y_i \mathbf{x}_i + 2\lambda \mathbf{w} = 0$$
Ridge solution

\[
\begin{align*}
\left( \sum_{i=1}^{n} x_i x_i^\top \right) w - \sum_{i=1}^{n} y_i x_i + 2\lambda w &= 0 \\
\left( \sum_{i=1}^{n} x_i x_i^\top + 2\lambda I \right) w - \sum_{i=1}^{n} y_i x_i &= 0 \\
\left( \sum_{i=1}^{n} x_i x_i^\top + 2\lambda I \right) w &= \sum_{i=1}^{n} y_i x_i \\
\boldsymbol{w}_{RR}^* &= \left( \sum_{i=1}^{n} x_i x_i^\top + 2\lambda I \right)^{-1} \sum_{i=1}^{n} y_i x_i
\end{align*}
\]

Red terms are those different from OLS!
Similar to OLS, we can derive the optimal solution of RR in matrix form:

\[
\mathbf{w}_{RR}^* = \left( \mathbf{X}^\top \mathbf{X} + 2\lambda \mathbf{I} \right)^{-1} \sum_{i=1}^{n} y_i \mathbf{x}_i
\]

An apple-to-apple comparison of OLS versus RR solutions:

\[
\mathbf{w}_{OLS}^* = \left( \mathbf{X}^\top \mathbf{X} \right)^{-1} \sum_{i=1}^{n} y_i \mathbf{x}_i
\]
Gradient descent (GD) for Ridge regression

Let solve the optimization with GD:

\[ w_{RR}^* = \arg \min_{w \in \mathbb{R}^d} \frac{1}{2} \sum_{i=1}^{n} (w^T x_i - y_i)^2 + \lambda \|w\|^2 \]

Let solve the optimization with GD:

\[ w_0 = 0 \]

\textbf{for} \ t = 1, 2, \ldots, T \ \textbf{do:}

\textbf{Update:}

\[ w_{t+1} = w_t - \eta_t \left( \sum_{i=1}^{n} (w_t^T x_i - y_i) x_i + 2\lambda w_t \right) \]

\textbf{end for}

\textbf{return} \ w_T

\footnotesize{Only change compared to OLS:}

\footnotesize{The time complexity is:} \( O(Tnd) \)
The Ridge regression tries to **shrink the parameters**:

\[
\arg \min_{\mathbf{w} \in \mathbb{R}^d} \frac{1}{2} \sum_{i=1}^{n} (\mathbf{w}^\top \mathbf{x}_i - y_i)^2 \]

such that \( \|\mathbf{w}\|^2 \leq \mu \)

For simplicity assume \( \mu = 1 \)

If the solution lies outside the circle, we just need to divide every coefficient by the length of vector to have it inside the circle.

\[
\left[ \frac{w_1}{w_1^2 + w_2^2}, \frac{w_2}{w_1^2 + w_2^2} \right]
\]
Regularization parameter

- Regularization methods introduce bias into the regression solution that can reduce variance considerably relative to the ordinary least squares (OLS) solution.

- In terms of fundamental trade-off:
  - Regularization increases training error (why?).
  - Regularization decreases variance.

- How should you choose $\lambda$?
  - **Theory**: as $n$ grows, $\lambda$ should be in the range $O(1)$ to $\sqrt{n}$ (note that regularization parameters is multiplied by regularization).
  - **Practice**: optimize validation set or cross-validation error.

- This almost always decreases the TEST error.
I am not convinced :-(

● It’s a weird thing to do, but Mehrdad says “always use regularization”.

  “Almost always decreases test error” should already convince you.

● But here are more reasons:

  1. The optimal solution is **unique** (why ?).
  2. $X^\top X$ does not need to be invertible. (no collinearity issues).
  3. Less sensitive to changes in $X$ or $y$.
  4. Gradient descent converge faster (bigger $\lambda$ means fewer iterations).
  5. Stein’s paradox: if $d \geq 3$, ‘shrinking’ moves us closer to ‘true’ $w$.
  6. Worst case: just set $\lambda$ small and get the same performance.

    [ let’s delegate it to cross-validation to decide, if regularization is not needed, we will get tiny or even zero value for regularization parameter ]
A geometric interpretation

By increasing the correlation between features, ellipsoids keep stretching out, contours getting closer to two parallel lines.

\[ \frac{1}{2} \| Xw - y \|^2 \]

Contour plot of objective

The concentric ellipse indicates sum of square term. When we move towards smaller ellipse from outer ellipse, the sum of square term will decrease.

OLS solution

RR solution

The RR optimal solution is a point which is common point (intersection) between ellipse and circle as well as gives a minimum value for RR optimization problem.

The constraint ball

\[ \| w \|^2 \leq \mu \]

Ridge regression reduces the variance of estimate because it limits the space that the solution vector can lie in.

Show that the contours of OLS objective are indeed ellipsoid?
Ridge regression in reality

ridge regression, lambda=0.1

ridge regression, lambda=0

ridge regression, lambda=1

ridge regression, lambda=1e-5
Instead of using the squared of the weights to impose the penalty, we take the sum of absolute value of all coefficients as regularization function:

$$\arg\min_{w \in \mathbb{R}^d} \frac{1}{2} \sum_{i=1}^{n} (w^\top x_i - y_i)^2 + \lambda \|w\|_1$$

OR

$$\arg\min_{w \in \mathbb{R}^d} \frac{1}{2} \sum_{i=1}^{n} (w^\top x_i - y_i)^2$$

such that $\|w\|_1 \leq \mu$
**Lasso solution**

- The loss function is not differentiable, so no analytical (closed-form) solution.
- Consider the 1 dimensional setting:

\[
\arg\min_{w \in \mathbb{R}} \frac{1}{2} \sum_{i=1}^{n} (wx_i - y_i)^2 + \lambda |w|
\]

Can be optimized using **Subgradient Descent** algorithm that will be covered later in the course!
A geometric interpretation of Lasso

There is a high probability that optimum point falls in the corner point of diamond, leading to a solution with zero value for subset of coefficients (sparsity).

The Lasso optimal solution is a point which is common point (intersection) between ellipse and diamond as well as gives a minimum value for the optimization problem.

By increasing the correlation between features, ellipsoids keep stretching out, contours getting closer to two parallel lines.

\[ \frac{1}{2} \| Xw - y \|_2^2 \]

Contour plot of objective

OLS solution

Lasso solution

The constraint ball

\[ \| w \|_1 \leq \mu \]
Lasso in practice
If you have prior knowledge that only a few variables affect the prediction (the optimal solution is sparse) use Lasso!
Suppose we have two features with same values: \( x_1, x_2 \in \mathbb{R} \) and \( x_2 = 2x_1 \).

Consider linear prediction functions: \( f(w_1, w_2) = w_1 x_1 + w_2 x_2 \).

In this setting all the functions with \( w_1 + 2w_2 = c \) is a solution!

(i.e., provide same predictions and have same training error)

What function will we select if we do \( \ell_1 \) or \( \ell_2 \) regularization?
OLS has infinitely many solutions (all these solutions lead to same training error)!

\[ w_1 + 2w_2 = c \]
Linearly related features

\[ \|w\|_2 \leq 1 \]

\( \ell_2 \) prefers variables with larger scale – spreads weight proportional to scale (note the difference when we had duplicate features)
**Linearly related features**

$\ell_1$ regularization chooses variable with larger scale, 0 weight to others (note the difference when we had duplicate features)
Lasso vs Ridge

- Ridge has a unique solution but Lasso might have multiple optimal solutions!

- Optimization Ridge objective can be done more efficiently (differentiable vs non-differentiable)

- Sparsity feature of Lasso can also be used as feature selection (finding the most important features among all features), an advantage compared to Ridge

- The best of both worlds (shrinkage of Ridge + sparsity of Lasso): the Elastic Net Regression which sets the regularization as:

\[
\alpha \|w\|_1 + (1 - \alpha) \|w\|_2^2
\]
Regression in scikit-learn

```python
>>> from sklearn import linear_model
>>> reg = linear_model.LinearRegression()
>>> reg.fit([[0, 0], [1, 1], [2, 2]], [0, 1, 2])
...  
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
>>> reg.coef_
array([0.5, 0.5])
```

```python
>>> from sklearn import linear_model
>>> reg = linear_model.Ridge(alpha=.5)
>>> reg.fit([[0, 0], [0, 0], [1, 1]], [0, 1, 1])
Ridge(alpha=0.5, copy_X=True, fit_intercept=True, max_iter=None, normalize=False, random_state=None, solver='auto', tol=0.001)
>>> reg.coef_
array([0.34545455, 0.34545455])
>>> reg.intercept_
0.13636
```

```python
>>> from sklearn import linear_model
>>> reg = linear_model.Lasso(alpha=0.1)
>>> reg.fit([[0, 0], [1, 1]], [0, 1])
Lasso(alpha=0.1, copy_X=True, fit_intercept=True, max_iter=10000, normalize=False, positive=False, precompute=False, random_state=None, selection='cyclic', tol=0.0001, warm_start=False)
>>> reg.predict([[1, 1]])
array([0.8])
```
Suppose we want to model the following data with a single feature ($d = 1$).

- Obviously linear model suffers from a high approximation error (underfitting).
- One option: fit a low-degree polynomial; this is known as polynomial regression.

Do we need to derive a whole new algorithm?
With a simple trick, **feature mapping**, we get polynomial regression for free!

Assume we have one feature. Note that the Polynomial regression model, say degree $p$, is expressed as:

$$f(w_0, w_1, w_2, \ldots, w_p) = w_0 + w_1 x + w_2 x^2 + \ldots + w_p x^p$$

We can write this equivalently as: $f(w_0, w_1, w_2, \ldots, w_p) = \mathbf{w}^\top \mathbf{x}$

where

$$\mathbf{x} = \begin{bmatrix} 1 \\ x \\ x^2 \\ \vdots \\ x^p \end{bmatrix} \quad \text{and} \quad \mathbf{w} = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ \vdots \\ w_p \end{bmatrix}$$
Feature mapping

- We can still use linear regression by **mapping** the input feature(s) to another space using polynomial feature mapping and treat the mapped feature as the input of a linear regression procedure.

- Basically, **a linear model in mapped space corresponds to a non-linear model in original feature spaces**.

- All of the derivations and algorithms so far in this lecture remain exactly the same!
Feature mapping

- How about if we have more than one feature $d > 1$?

- Consider two features and assume you wanna fit a polynomial degree 2, what would the mapping?

- The number of mapped features exponentially grows by $p$ and $d$!

- Later in the course we will learn about kernel trick to overcome this issue where we show we do not even need to explicitly map features!
Appendix
I am not convinced yet!

Let’s look at the effect of regularization from the linear algebraic viewpoint.

We saw that the eigenvalues of covariance matrix \( X^T X \) has a lot to say about the solution:

The optimal solution exists if this matrix is full rank
or
All the eigenvalues are non-zero

We now ask how regularization affects the optimal solution, in particular transformation of eigenvalues?
I am not convinced yet!

Let $X = U \Sigma V^\top$ be the SVD decomposition of the data matrix.

We can compute the optimal OLS solution in terms of above decomposition:

$$w_{\text{OLS}}^* = \left( X^\top X \right)^{-1} X^\top y$$

$$= \left( \left( U \Sigma V^\top \right)^\top U \Sigma V^\top \right) \left( U \Sigma V^\top \right)^\top y$$

$$= \left( V \Sigma U^\top U \Sigma V^\top \right)^{-1} V \Sigma U^\top y$$

$$= \left( V \Sigma^2 V^\top \right)^{-1} V \Sigma U^\top y$$

$$= \left( V^\top \right)^{-1} \left( \Sigma^2 \right)^{-1} V^{-1} V \Sigma U^\top y$$

$$= \left( V^\top \right)^{-1} \left( \Sigma^2 \right)^{-1} V^{-1} V \Sigma U^\top y$$

$$= \left( \sum_{i=1}^d \frac{\sigma_i}{\sigma_i^2} v_i u_i^\top \right) y$$
Similarly, we can compute the optimal Ridge solution in terms of SVD decomposition of data matrix $X = U\Sigma V^\top$:

$$w^*_{RR} = \left( X^\top X + 2\lambda I \right)^{-1} X^\top y$$

$$= \left( V \text{diag}(\sigma_1^2 + \lambda, \sigma_2^2 + \lambda, \ldots, \sigma_d^2 + \lambda) V^\top \right)^{-1} X^\top y$$

$$= V \text{diag}(\sigma_1^2 + \lambda, \sigma_2^2 + \lambda, \ldots, \sigma_d^2 + \lambda)^{-1} V^\top V \Sigma U^\top y$$

$$= V \text{diag}(\sigma_1^2 + \lambda, \sigma_2^2 + \lambda, \ldots, \sigma_d^2 + \lambda)^{-1} \Sigma U^\top y$$

$$= V \text{diag}\left( \frac{\sigma_1}{\sigma_1^2 + \lambda}, \frac{\sigma_2}{\sigma_2^2 + \lambda}, \ldots, \frac{\sigma_d}{\sigma_d^2 + \lambda} \right) U^\top y$$

$$= \left( \sum_{i=1}^{d} \frac{\sigma_i}{\sigma_i^2 + 2\lambda} v_i u_i^\top \right) y$$
Let’s compare the solutions:

\[
\mathbf{w}_{\text{OLS}}^* = \left( \sum_{i=1}^{d} \frac{\sigma_i}{\sigma_i^2} \mathbf{v}_i \mathbf{u}_i^\top \right) \mathbf{y}
\]

\[
\mathbf{w}_{\text{RR}}^* = \left( \sum_{i=1}^{d} \frac{\sigma_i}{\sigma_i^2 + 2\lambda} \mathbf{v}_i \mathbf{u}_i^\top \right) \mathbf{y}
\]

Regularization makes the solution not sensitive to very small eigenvalues (coming from noise)
THIS FEATURE IS OVERFITTING
WE NEED TO USE DEEP LEARNING
WE NEED TO CLEAN THE DATA
THIS IS BIG DATA
HAVE YOU GUYS FED THE ELEPHANT?