13 Shrinkage: Ridge Regression, Subset Selection, and Lasso

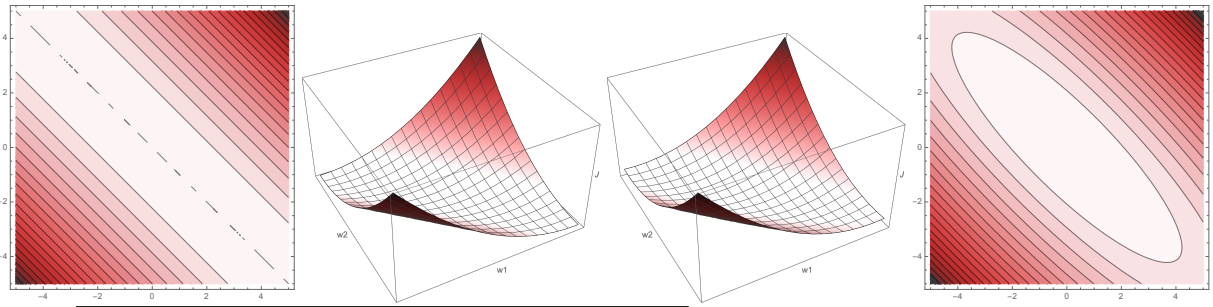
RIDGE REGRESSION aka Tikhonov Regularization

Least-squares linear regression + ℓ_2 penalized mean loss. (1) + (A) + (a) + (d).

Find w that minimizes $\|Xw - y\|^2 + \lambda \|w'\|^2 = J(w)$
where w' is w with component α replaced by 0.
X has fictitious dimension but we DON'T penalize α.

Adds a regularization term, aka a penalty term, for shrinkage: to encourage small $\|w'\|$. Why?

1. Guarantees positive definite normal eq'ns; always unique solution.
[Standard least-squares linear regression yields singular normal equations when the sample points lie on a common hyperplane in feature space—for example, when $d > n$.]



lslrcontour.pdf, lslr.pdf, ridge.pdf, ridgecontour.pdf [The cost function $J(w)$ without and with regularization. This plot ignores the dimension of the bias term α.]

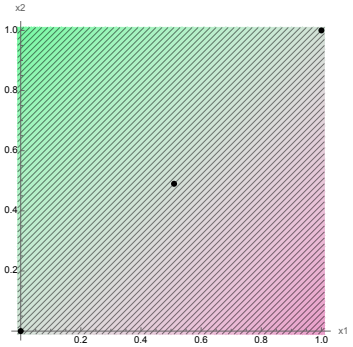
[At left, we see a cost function for least-squares regression, a positive semidefinite quadratic form. This cost function has many minima, and the regression problem is said to be ill-posed. By adding a small penalty term, we obtain a positive definite quadratic form (right), with one unique minimum. "Regularization" implies that we are turning an ill-posed problem into a well-posed problem.]

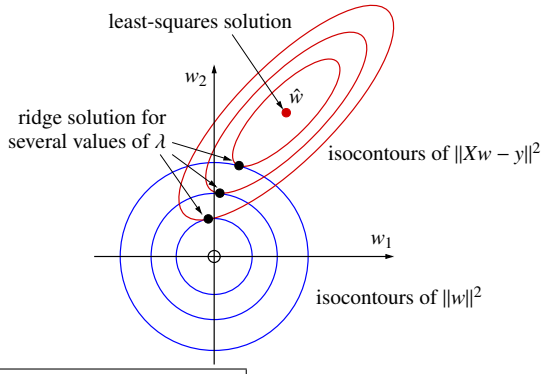
[That was the original motivation, but the next has become more important in machine learning . . .]

2. Reduces overfitting by reducing variance. Why?

Example: Input $X_1 = (0, 0)$ with label 0; $X_2 = (1, 1)$ with label 0; $X_3 = (0.51, 0.49)$ with label 1.
Linear regr. gives $50x_1 - 50x_2$. [This linear function fits all three points exactly.]
Big weights!

[Weights this big would be justified if there were big differences between labels, or if there were small distances between points, but neither is true. Large weights imply that tiny changes in x can cause huge changes in y. Consider that the labels don't differ by more than 1 and the points are separated by distances greater than 0.7. So these disproportionately large weights are a sure sign of overfitting.]
So we penalize large weights.
[This use of regularization is closely related to the first one. When you have large variance and a lot of overfitting, it implies that your problem is *close to* being ill-posed, even though technically it might be well-posed.]





ridgeterms2.pdf (redrawing of ISL, Figure 6.7) [In this plot of weight space, for simplicity, we're not using a bias term α (we set it to zero). $\hat{w}$ is the least-squares solution. The red ellipses are isocontours of $\|Xw - y\|^2$. The blue circles are isocontours of $\|w\|^2$, centered at the origin. The ridge regression solution lies where a red isocontour just touches a blue isocontour tangentially. As λ increases, the solution will occur at a more outer red isocontour and a more inner blue isocontour. This shrinks w and helps to reduce overfitting.]

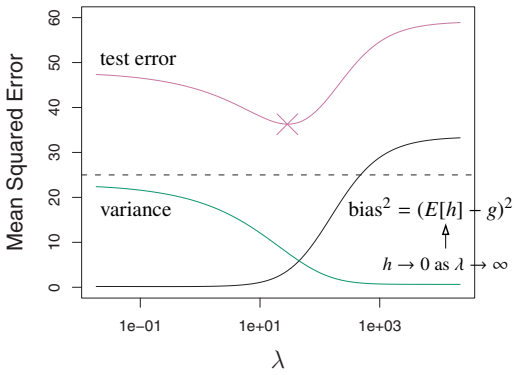
Setting $\nabla J = 0$ gives normal eq'ns

$$(X^\top X + \lambda I')\, w = X^\top y$$

where I' is identity matrix w/bottom right set to zero. [Don't penalize the bias term α.]
[Don't worry; $X^\top X + \lambda I'$ is always positive definite for $\lambda > 0$, assuming X ends with a column of 1's.]

Algorithm: Solve for w. Return $h(z) = w^\top z$.

Increasing λ $\Rightarrow$ more regularization; smaller $\|w'\|$
Recall [from the previous lecture] our data model $y = Xv + e$, where e is noise.
Variance of ridge regr. at test pt z is $\mathrm{Var}(z^\top (X^\top X + \lambda I')^{-1} X^\top e)$.
As $\lambda \to \infty$, variance $\to 0$, but bias increases.



ridgebiasvar.pdf (ISL, Figure 6.5) [Plot of bias^2 & variance as λ increases.]

[The test error as a function of λ is a U-shaped curve. We find the bottom by validation. Regularization is intended to reduce the variance, but this method of regularization also increases the bias.]

λ is a hyperparameter; tune by (cross-)validation.

Ideally, features should be "normalized" to have same variance.
Alternative: use asymmetric penalty by replacing I' w/other diagonal matrix. [For example, if you use polynomial features, you could use different penalties for monomials of different degrees.]

Bayesian Justification for Ridge Reg.

Assign a prior probability on w': $w' \sim \mathcal{N}(0, \varsigma^2)$, with PDF $f(w') \propto e^{-\|w'\|^2/(2\varsigma^2)}$
[This prior probability says that we think weights close to zero are more likely to be correct.]
Apply MLE to maximize the posterior prob.

$$\begin{aligned}
\text{Bayes' Theorem: posterior } f_{W|X,Y}(w) &= \frac{f_{Y|X,W}(y)\, f(w')}{f_{Y|X}(y)} \\
\text{Maximize log posterior} &= \ln f_{Y|X,W}(y) + \ln f(w') - \text{const} \\
&= -\text{const}\, \|Xw - y\|^2 - \text{const}\, \|w'\|^2 - \text{const} \\
&\Rightarrow \text{Minimize } \|Xw - y\|^2 + \lambda \|w'\|^2
\end{aligned}$$

[We are treating w and y as random variables, but X as a fixed constant—it's not random.]
This method (using MLE, but maximizing posterior) is called maximum *a posteriori* (MAP).
[A prior probability on the weights is another way to understand regularizing ill-posed problems.]

FEATURE SUBSET SELECTION

[Some of you may have noticed as early as Homework 1 that you can sometimes get better performance on a spam classifier simply by dropping some useless features.]

All features increase variance, but not all features reduce bias.
Idea: Identify poorly predictive features, ignore them (weight zero).
Less overfitting, smaller test errors.
2nd motivation: Inference. Simpler models convey interpretable wisdom.
Useful in all classification & regression methods.
Sometimes it's hard: Different features can partly encode same information.
Combinatorially hard to choose best feature subset.

Alg: Best subset selection. Try all $2^d - 1$ nonempty subsets of features. [Train one classifier per subset.]
Choose best classifier by (cross-)validation. Slow.

[Obviously, best subset selection isn't feasible if we have a lot of features. But it gives us an "ideal" algorithm to compare practical algorithms with. If d is large, there is no algorithm that's guaranteed to find the best subset and that runs in acceptable time. But heuristics often work well.]

Heuristic 1: Forward stepwise selection.
Start with null model (0 features); repeatedly add best feature until validation errors start increasing (due to overfitting) instead of decreasing. At each outer iteration, inner loop tries every feature & chooses the best by validation. Requires training $O(d^2)$ models instead of $O(2^d)$.
Not perfect: e.g., won't find the best 2-feature model if neither of those features yields the best 1-feature model. [That's why it's a heuristic.]

Heuristic 2: Backward stepwise selection.
Start with all d features; repeatedly remove feature whose removal gives best reduction in validation error.
Also trains $O(d^2)$ models.

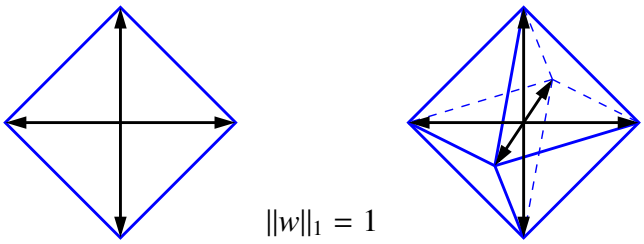
[Forward stepwise is a better choice when you suspect only a few features will be good predictors; e.g., spam. Backward stepwise is better when most features are important. If you're lucky, you'll stop early.]

LASSO (Robert Tibshirani, 1996)

Least-squares linear regression + ℓ_1 penalized mean loss. (1) + (A) + (a) + (e).
"Least absolute shrinkage and selection operator"
[This is a regularized regression method similar to ridge regression, but it has the advantage that it often naturally sets some of the weights to zero.]

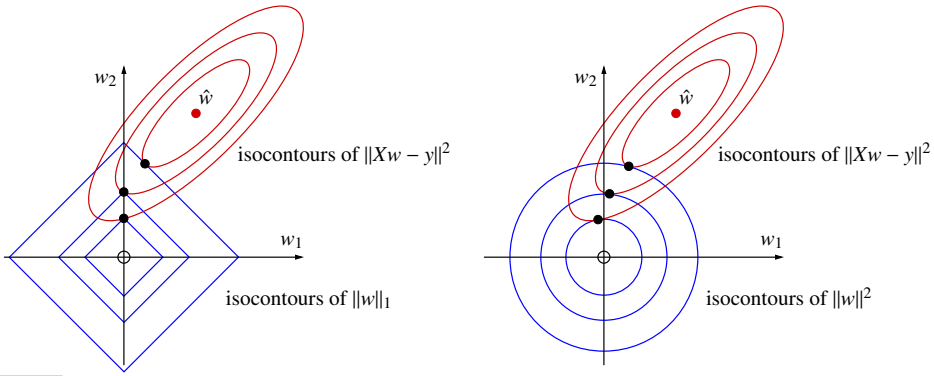
Find w that minimizes $\|Xw - y\|^2 + \lambda \|w'\|_1$ where $\|w'\|_1 = \sum_{i=1}^{d} |w_i|$ (Don't penalize α.)

Recall ridge regr.: isosurfaces of $\|w'\|^2$ are hyperspheres.
The isosurfaces of $\|w'\|_1$ are cross-polytopes.
The unit cross-polytope is the convex hull of all the positive & negative unit coordinate vectors.



[Draw this figure by hand crosspolys.pdf]

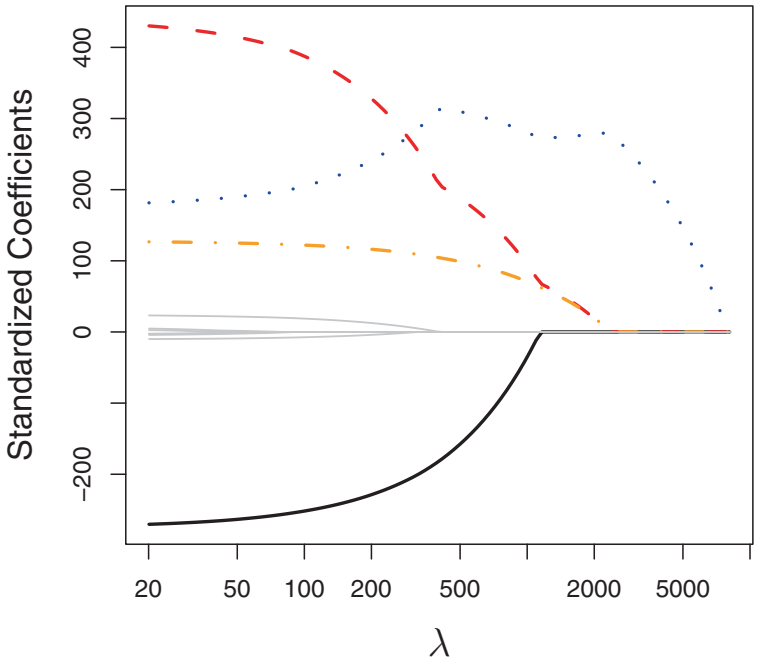
[You get larger and smaller cross-polytope isosurfaces by scaling these.]



lassoridge2.pdf [Isocontours of the terms of the objective function for the Lasso appear at left. Compare with the ridge regression isocontours at right.]

[The red ellipses are the isocontours of $\|Xw - y\|^2$, and the least-squares solution lies at their center. The isocontours of $\|w'\|_1$ are diamonds centered at the origin (blue). The solution lies where a red isocontour just touches a blue diamond. What's interesting is that for large values of λ, the red isocontour touches just the tip of a diamond. Then the weight w_1 gets set to zero. That's what we want to happen to features that don't have enough predictive power. For small values of λ, the red isosurface just barely touches a side of a diamond instead of a tip of the diamond, and no weight gets set to zero.]

[When you go to higher dimensions, you might have several weights set to zero. For example, in 3D, if the red isosurface just touches a sharp vertex of a cross-polytope, two of the three weights get set to zero. If it just touches a sharp edge of a cross-polytope, one weight gets set to zero. If it just touches a flat side of a cross-polytope, no weight is zero.]



lassoweights.pdf (ISL, Figure 6.6) [Weights as a function of λ.]

[This shows the weights for a typical linear regression problem with about 10 variables. You can see that as lambda increases, more and more of the weights become zero. Only four of the weights are really useful for prediction; they're in color. Statisticians used to choose λ by looking at a chart like this and trying to eyeball a spot where there aren't too many predictors and the weights aren't changing too fast. But nowadays they prefer validation.]

Sometimes sets some weights to zero, especially for large λ.
Algs: subgradient descent, least-angle regression (LARS), forward stagewise

[Lasso can be reformulated as a quadratic program, but it's a quadratic program with 2^d constraints, because a d-dimensional cross-polytope has 2^d facets. In practice, special-purpose optimization methods have been developed for Lasso. I'm not going to teach you one, but if you need one, look up the last two of these algorithms. LARS is built into the R Programming Language for statistics.]

[As with ridge regression, you should probably normalize the features first before applying Lasso.]