CSCE 633: Machine Learning

Lecture 8: Model Selection and Regularization

Texas A&M University

9-11-18

Last Time

- Logistic Regression
- Gradient Descent

Goals of this lecture

- Ridge Regression
- Lasso Regularization
- Measures of Model Performance

Shrinkage

- Train a model fitting all p predictors that constrains or regularizes the coefficients
- Two best methods for this, Ridge Regression and the Lasso

Ridge Regression

- Recall the least squares fit for $\beta_0, \beta_1, \cdots, \beta_p$ minimizes
- $RSS = \sum_{i=1}^{n}(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij})^2$

Ridge Regression

- Recall the least squares fit for $\beta_0, \beta_1, \cdots, \beta_p$ minimizes
- $RSS = \sum_{i=1}^{n}(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij})^2$
- With Ridge Regression, we modify the equation that we want to minimize by adding a penalty (paying a price) for using predictors

Ridge Regression

- Recall the least squares fit for $\beta_0, \beta_1, \cdots, \beta_p$ minimizes
- $RSS = \sum_{i=1}^{n}(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij})^2$
- With Ridge Regression, we modify the equation that we want to minimize by adding a penalty (paying a price) for using predictors

$$\sum_{i=1}^{n}(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij})^2 + \lambda \sum_{j=1}^{p} \beta_j^2 = RSS + \lambda \sum_{j=1}^{p} \beta_j^2$$

- Where $\lambda \geq 0$ is the tuning parameter

Ridge Regression

$$\sum_{i=1}^{n}(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij})^2 + \lambda \sum_{j=1}^{p} \beta_j^2 = RSS + \lambda \sum_{j=1}^{p} \beta_j^2$$

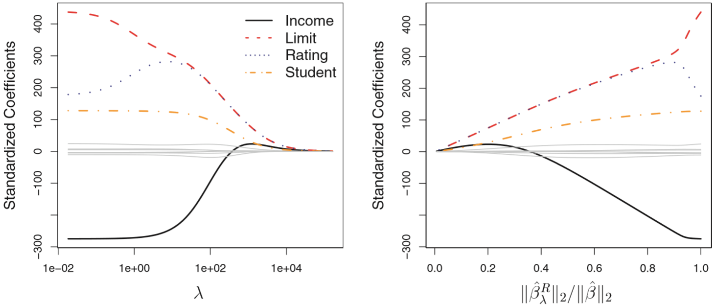
- Where $\lambda \geq 0$ is the tuning parameter
- Ridge Regression creates a tradeoff, we still want coefficients that reduce RSS but now we have a shrinkage penalty.
- This penalty is small if $\beta_0, \cdots, \beta_p$ are close to 0
- Where least squares creates a single set of coefficients, Ridge Regression creates a set $\hat{\beta}_\lambda^R$ for each λ

Ridge Regression

$$\sum_{i=1}^{n}(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij})^2 + \lambda \sum_{j=1}^{p} \beta_j^2 = RSS + \lambda \sum_{j=1}^{p} \beta_j^2$$

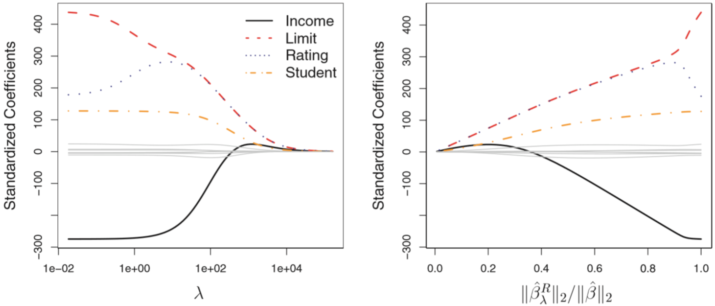
- Where least squares creates a single set of coefficients, Ridge Regression creates a set $\hat{\beta}_\lambda^R$ for each λ
- Selecting the right λ is key (done via cross-validation)
- Note, penalty is not assigned to the intercept β_0 since β_0 is the measure of the mean value of the response when $x_{i1} = x_{i2} = \cdots = x_{ip} = 0$
- If we assume columns of X have been centered (meaning each column has a mean of 0) then intercept is $\hat{\beta}_0 = \bar{y} = \sum_{i=1}^{n} \frac{y_i}{n}$

Ridge Regression and Credit Data



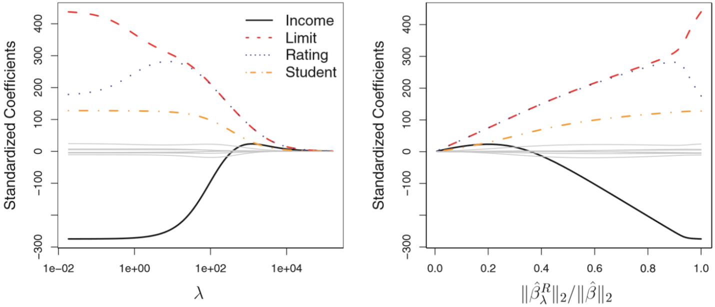
- Each line is one of the ten variables as a function of λ. Solid black line is the income variable
- We can see when $\lambda = 0$ we get standard least squares.
- When $\lambda = \infty$ we have the null model

Ridge Regression and Credit Data



- Income, limit, rating, and student have the largest coefficients.
- Note, in some steps individual estimates might actually grow because of relative important
- Right hand side of figure, we scale the x-axis as $\frac{\|\hat{\beta}^R_\lambda\|_2}{\|\hat{\beta}\|_2}$ where $\|\hat{\beta}\|_2$ is the ℓ_2 norm of the least squares coefficient estimates

Ridge Regression and Credit Data



- Right hand side of figure, we scale the x-axis as $\frac{\|\hat{\beta}^R_\lambda\|_2}{\|\hat{\beta}\|_2}$ where $\|\hat{\beta}\|_2$ is the ℓ_2 norm of the least squares coefficient estimates
- That value ranges from 1 when $\lambda = 0$ to 0 when $\lambda = \infty$
- Thus, the x-axis represents amounts coefficient estimates have been shrunk to 0

Scaling

- Scaling is now an important part of data we need to consider.
- Whereas, with least squares, if X_j was scaled by some constant c then least squares β would have been scaled by $\frac{1}{c}$, this is not true for Ridge Regression
- $x_j \hat{\beta}^R_{j,\lambda}$ will depend on λ and scaling of x_j and perhaps even the scaling of other predictors.
- To avoid scaling issues, we should standardize predictors

$$\tilde{x}_{ij} = \frac{x_{ij}}{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_{ij} - \bar{x}_{ij})^2}}$$

- Where the denominator is the estimated standard deviation of j

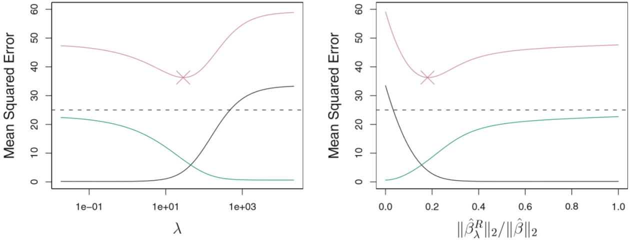
Normalization

- One important step in numeric data is normalization of the data to help these techniques
- one common technique is to center and scale each predictor X_j
- This causes all the predictors to have mean 0 and standard deviation of 1

$$\tilde{x}_j = \frac{x_j - \bar{x}_j}{\sigma_j^2}$$

Why does this work?

- Rooted in the bias-variance trade-off
- As λ increases, flexibility of ridge regression fit decreases, decreasing variance but increasing bias.



- Simulated data of $p = 45$, $n = 50$, black is bias, green is variance, purple is test error
- $\lambda = 30$ is the optimal solution and MSE of least squares is almost as high as null-model

LASSO

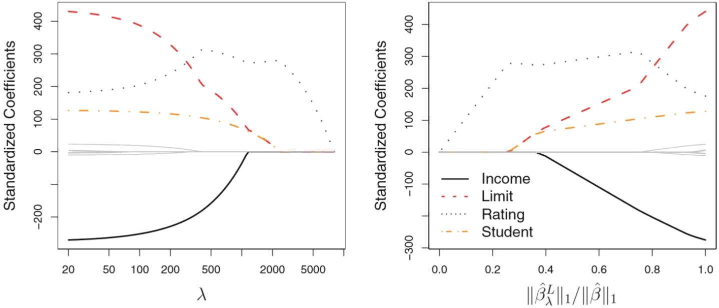
- Ridge Regression has one obvious disadvantage. Unlike subset methods, ridge regression still fits all p predictors.
- The penalty $\lambda \sum_j \beta_j^2$ will shrink all coefficients but none will hit 0 exactly
- This may not be a problem for accuracy but is for interpretability
- For example with our credit data set, Ridge Regression will still use all 10 predictors, even if it finds that income, limit, rating, and student are the most important.

LASSO Regularization

$$\sum_{i=1}^{n}(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij})^2 + \lambda\sum_{j=1}^{p}|\beta_j| = RSS + \lambda\sum_{j=1}^{p}|\beta_j|$$

- Creates a set $\hat{\beta}_\lambda^L$ for each λ
- We use the ℓ_1 norm instead of ℓ_2
- Lasso shrinks coefficients but the ℓ_1 penalty drives coefficients to 0 when λ is sufficiently large
- This means Lasso performs variable selection!

Lasso and Credit Data



- Lasso picks rating, then student and limit together, then income. Eventually all others would enter as you approach least squares fit
- Where ridge selects coefficients/shrinkage, lasso produces models with any number of variables

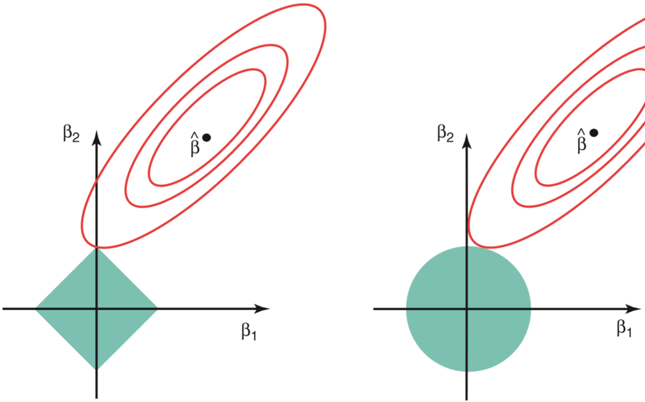
Another Formulation

$$\min_{\beta} \sum_{i=1}^{n} (y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij})^2$$

Subject to $\sum_{j=1}^{p} |\beta_j| \leq s$ for Lasso
and Subject to $\sum_{j=1}^{p} \beta_j^2 \leq s$ for Ridge Regression

- If $p = 2$ Lasso solution falls within the diamond $|\beta_1| + |\beta_2| \leq s$
- Circle for Ridge $\beta_1^2 + \beta_2^2 \leq s$

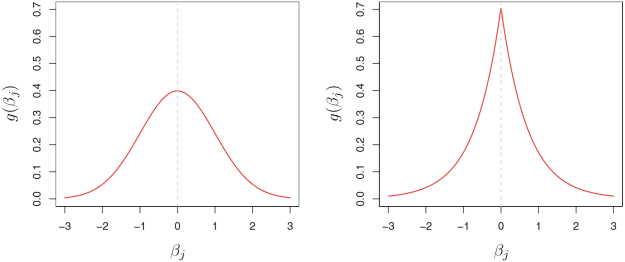
Another Formulation



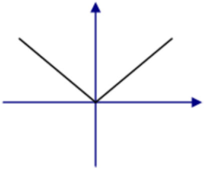
- Ellipses are increasing *RSS* from the least squares solution
- if the λ allows enough to include *RSS* that is the fit found
- Because Lasso will intersect at a corner, while Ridge somewhere in between on circle, is why Lasso sets some coefficients to 0 while Ridge just shrinks them

Final Notes

- Lasso is better if small set of predictors dominates response
- Ridge is better if all predictors contribute somewhat equally
- Cannot tell in advance, need cross-validation to give us an idea
- Lasso shrinks very differently than Ridge, known as soft thresholding
- Ridge assumes the density function of the posterior probabilities of β are Gaussian (most coefficients are somewhere near 0), while Lasso assumes Laplacian (most coefficients centered at 0)



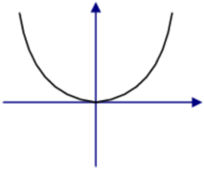
More about sparsity



$0.5\times(x-v)^2 + \lambda|x|$

If $v \geq \lambda$, $x=v-\lambda$
If $v \leq -\lambda$, $x=v+\lambda$
Else, $x=0$

Nondifferentiable at 0

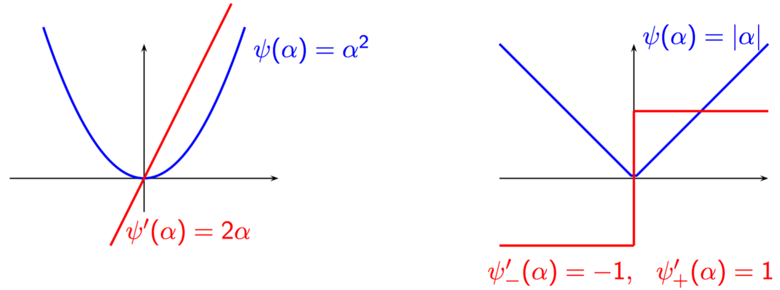


$0.5\times(x-v)^2 + \lambda x^2$

$x=v/(1+2\lambda)$

Differentiable at 0

More about sparsity



The gradient of the ℓ_2-norm vanishes when α get close to 0. On its differentiable part, the norm of the gradient of the ℓ_1-norm is constant.

How to Solve LASSO

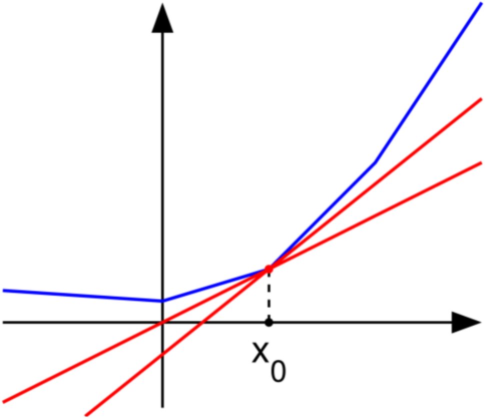
Rewrite the optimization problem:

$$\min_{\beta} \frac{1}{2}\|y - X\beta\|_2^2 + \lambda\|\beta\|_1$$

Challenges:

- The optimization if non-smooth.
- Subgradient Method
  - Subgradients are easy to derive and implement
  - Convergence needs carefully chosen step sizes
  - Convergence is weak theoretically

Subgradient Method



How to Solve LASSO

Fast l_1 minimization algorithms:

- Iterative Shrinkage Thresholding Algorithm (ISTA)
- Proximal Gradient Method (PGM)
- Alternating Direction Methods of Multpliers

Iterative Shrinkage Thresholding Algorithm (ISTA)

ISTA considers the LASSO model as a special case of the composite objective function:

$$\min_{\beta} F(\beta) = f(\beta) + g(\beta),$$

where f is a smooth and convex function, and g is the regularization term that is not necessarily smooth nor convex. Here $f(\beta) = \frac{1}{2}\|y - X\beta\|_2^2$.

- If $g(\beta) = \lambda\|\beta\|_2^2$: Ridge regression.
- If $g(\beta) = \lambda\|\beta\|_1$: LASSO.

ISTA using Hessian Matrix Approximation

Estimate $f(\beta)$ using its Taylor expansion to the second order around β^k:

$$\beta^{k+1} = argmin_\beta\{f(\beta^k)+\nabla f(\beta^k)^T(\beta-\beta^k)+\frac{1}{2}(\beta-\beta^k)^T\nabla^2 f(\beta^k)(\beta-\beta^k)+g(\beta)\}$$

$$\approx argmin_\beta\{\nabla f(\beta^k)^T(\beta-\beta^k)+\tfrac{\alpha^k}{2}\|\beta-\beta^k\|_2^2+g(\beta)\}$$
$$= \text{argmin}_\beta\{\tfrac{\alpha^k}{2}\|\beta-\gamma^k\|_2^2+g(\beta)\}$$

where

$$\gamma^k = \beta^k - \frac{1}{\alpha^k}\nabla f(\beta^k).$$

ISTA using Hessian Matrix Approximation

Specifically for LASSO, where $g(\beta) = \|\beta\|_1$, the last optimization step is seperable:

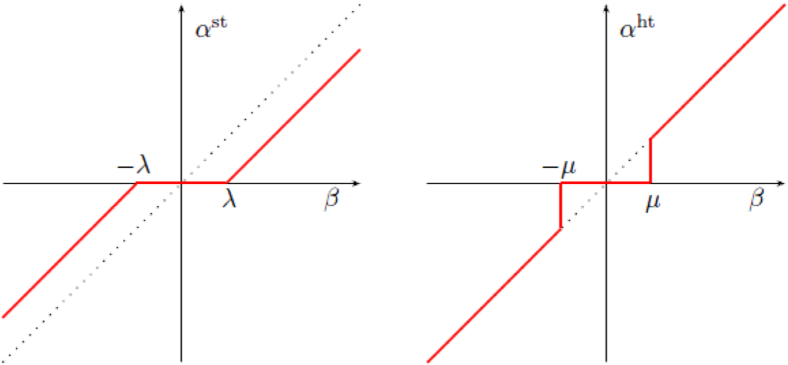
$$\beta^{k+1} = argmin_\beta\{\sum_i \frac{\alpha^k}{2}(\beta_i - \gamma_i^k)_2^2 + \lambda|\beta_i|\}\}$$

The problem consists of multiple independent 1-D problems that have explicit solution:

$$\beta_i^{k+1} = soft(\gamma_i^k, \frac{\lambda}{\alpha^k})$$

$$\begin{aligned} \text{soft}(u,a) &\doteq \text{sgn}(u)\max\{|u| - a, 0\} \\ &= \begin{cases} \text{sgn}(u)(|u| - a) & \text{if |u|>a} \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

Soft Thresholding Function



(a) Soft-thresholding operator, $\alpha^{\mathrm{st}} = \mathrm{sign}(\beta)\max(|\beta| - \lambda, 0)$.

(b) Hard-thresholding operator $\alpha^{\mathrm{ht}} = \mathbf{1}_{|\beta| \geq \mu}\beta$.

ISTA using Proximal Gradient Method

At each step, perform a gradient descent step on $f(\beta)$ without considering the non-smooth regularization:

$$\gamma^k = \beta^k - \alpha^k \nabla f(\beta^k) = \beta^k + \alpha^k X^T (y - X\beta^k)$$

Then combine the regularization by solving the following proximity problem:

$$\beta^{k+1} = argmin_\beta \{ \frac{1}{2\alpha^k} \|\beta - \gamma^k\|_2^2 + \lambda \|\beta\|_1 \}$$

which will induce exactly the same solution:

$$\beta^{k+1} = soft(\beta^k + \alpha^k X^T (y - X\beta^k), \alpha^k \lambda).$$

Need select a small enough step size α^k.

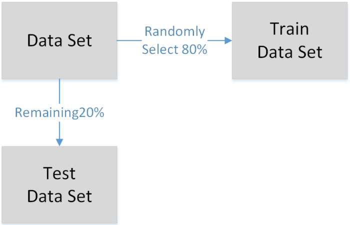
Lasso: Continued

- Selecting λ
- Further Lasso Solvers and Elastic Net

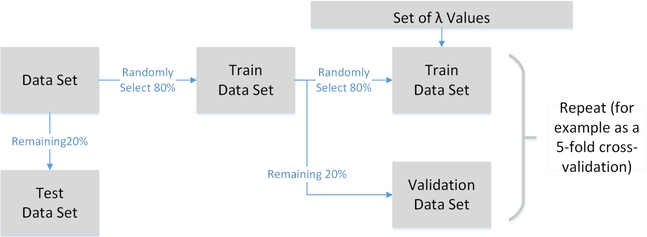
LASSO: Picking λ

- Need to pick best λ (or s in the alternative formulation) for best estimation
- We can run a cross-validation over a grid of λ values
- We pick the *lambda* with the smallest error

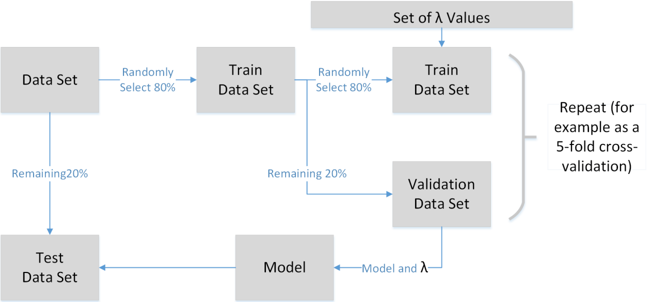
LASSO: Picking λ



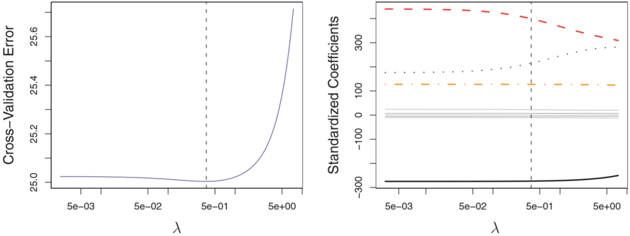
LASSO: Picking λ



LASSO: Picking λ

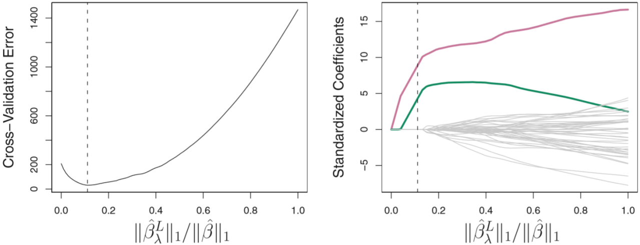


LASSO: Our Credit Example



- Sometimes Lasso does not do better than Least Squares Solution
- Small λ selected here

LASSO: Synthetic Example



- Sometimes Lasso does a lot better than Least Squares Solution

LASSO: Elastic Net

- Last week, we discussed the differences between Ridge Regression and Lasso
- We also discussed how it is not immediately obvious which would be better, sometimes need cross-validation to test
- if $n > p$ but variables are correlated, ridge empirically does better than lasso
- if $p > n$ lasso cannot select more than n variables before it saturates.
- A mix between Lasso and Ridge exists, called Elastic Net

Vanilla Elastic Net

New Objective Function is

$$J(\beta, \lambda_1, \lambda_2) = \|y - X\beta\|^2 + \lambda_2\|\beta\|_2^2 + \lambda_1\|\beta\|_1$$

- The objective now has a penalty that is from ridge regression and a penalty that is from lasso
- It turns out this doesn't predict really well, unless the optimal solution is found by ridge or by lasso
- This is because some solution in the middle has coefficients penalized by both λ_1 and λ_2

Vanilla Elastic Net

New Objective Function is

$$J(\beta, \lambda_1, \lambda_2) = \|y - X\beta\|^2 + \lambda_2\|\beta\|_2^2 + \lambda_1\|\beta\|_1$$

- The objective now has a penalty that is from ridge regression and a penalty that is from lasso
- It turns out this doesn't predict really well, unless the optimal solution is found by ridge or by lasso
- This is because some solution in the middle has coefficients penalized by both λ_1 and λ_2
- To fix it, we adjust the optimal solution. So, first, we solve the vanilla version

LARS-Elastic Net

First we re-write X as

$$\tilde{X} = \frac{1}{\sqrt{1+\lambda_2}} \begin{pmatrix} X \\ \sqrt{\lambda_2} I_p \end{pmatrix}$$

Where I_p is the identity matrix and

$$\tilde{y} = \begin{pmatrix} y \\ 0_{p \times 1} \end{pmatrix}$$

Then we solve for β like a normal lasso problem

$$\tilde{\beta} = argmin_{\tilde{\beta}} \|\tilde{y} - \tilde{X}\tilde{\beta}\|^2 + \frac{\lambda_1}{\sqrt{1+\lambda_2}} \|\tilde{\beta}\|_1$$

So $\beta = \frac{\tilde{\beta}}{\sqrt{1+\lambda_2}}$

Improved Elastic Net

Then we solve for β like a normal lasso problem

$$\tilde{\beta} = argmin_{\tilde{\beta}} \|\tilde{y} - \tilde{X}\tilde{\beta}\|^2 + \frac{\lambda_1}{\sqrt{1+\lambda_2}}\|\tilde{\beta}\|_1$$

So $\beta = \frac{\tilde{\beta}}{\sqrt{1+\lambda_2}}$

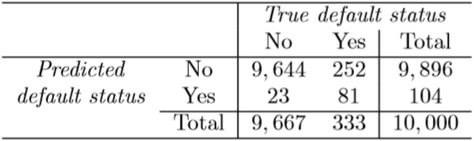
- So now we want to undo one of the penalties so coefficients aren't double penalized
- for simplicity we undo the λ_2 penalty (ℓ_2)

$$\hat{\beta} = \sqrt{1+\lambda_2}\tilde{\beta}$$

Measures of Classification Accuracy

- Confusion Matrices
- Class labels versus Probabilities
- Measurements

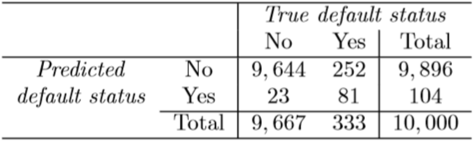
Confusion Matrix



		True default status		
		No	Yes	Total
Predicted	No	9,644	252	9,896
default status	Yes	23	81	104
	Total	9,667	333	10,000

- If we denote true positives as TP, false positives as FP, true negatives as TN, and false negatives as FN then we can say Accuracy is
- $ACC = \frac{TP+TN}{TP+FN+TN+FP}$
- What is the accuracy here?

Confusion Matrix



		True default status		
		No	Yes	Total
Predicted	No	9,644	252	9,896
default status	Yes	23	81	104
	Total	9,667	333	10,000

- If we denote true positives as TP, false positives as FP, true negatives as TN, and false negatives as FN then we can say Accuracy is
- $ACC = \frac{TP+TN}{TP+FN+TN+FP}$
- What is the accuracy here? 97.25%

Confusion Matrix: Problems with Accuracy?

		True	
		No	Yes
Predicted	No	50	50
	Yes	0	0

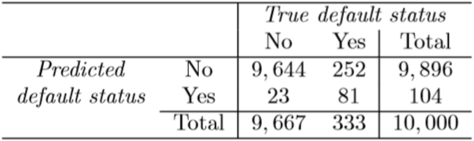
- If we denote true positives as TP, false positives as FP, true negatives as TN, and false negatives as FN then we can say Accuracy is
- $ACC = \frac{TP+TN}{TP+FN+TN+FP}$

Confusion Matrix with Class Imbalance: Problems with Accuracy?

		True	
		No	Yes
Predicted	No	900	100
	Yes	0	0

- If we denote true positives as TP, false positives as FP, true negatives as TN, and false negatives as FN then we can say Accuracy is
- $ACC = \frac{TP+TN}{TP+FN+TN+FP}$

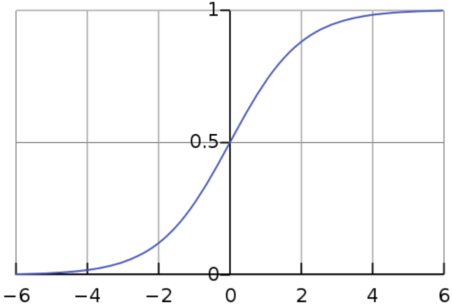
Confusion Matrix: Measurements in Addition to Accuracy



		True default status		
		No	Yes	Total
Predicted default status	No	9,644	252	9,896
	Yes	23	81	104
	Total	9,667	333	10,000

- If we denote true positives as TP, false positives as FP (Type I Error), true negatives as TN, and false negatives as FN (Type II Error) then we can say Accuracy is
- $ACC = \frac{TP+TN}{TP+FN+TN+FP}$
- Recall $=$ Sensitivity $=$ True Positive Rate $= \frac{TP}{TP+FN}$
- Specificity $=$ True Negative Rate $= \frac{TN}{TN+FP} = 1 - FPR$
- False Positive Rate (FPR) $= \frac{FP}{FP+TN}$
- Precision $=$ Positive Predictive Value $= \frac{TP}{TP+FP}$
- F1 Score $= 2 \times \frac{Precision \times Recall}{Precision + Recall}$
- Can calculate these per-class and average together or total across all classes

Logistic Regression: Decision Threshold

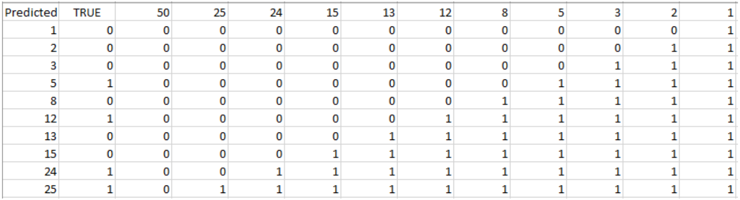


Logistic Regression: Decision Threshold

If we pick a decision threshold of $p(x) > 0.5$ what happens?

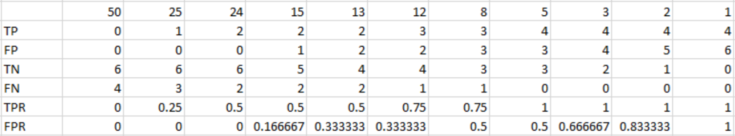
Predicted	Ground Truth
2%	0
3%	0
5%	1
1%	0
15%	0
25%	1
24%	1
13%	0
8%	0
12%	1

Logistic Regression: Decision Threshold



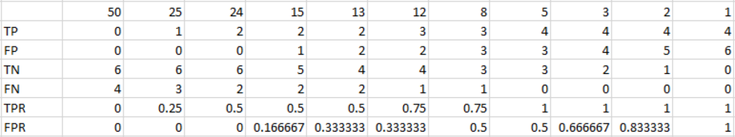
Predicted	TRUE	50	25	24	15	13	12	8	5	3	2	1
1	0	0	0	0	0	0	0	0	0	0	0	1
2	0	0	0	0	0	0	0	0	0	0	1	1
3	0	0	0	0	0	0	0	0	0	1	1	1
5	1	0	0	0	0	0	0	0	1	1	1	1
8	0	0	0	0	0	0	0	1	1	1	1	1
12	1	0	0	0	0	0	1	1	1	1	1	1
13	0	0	0	0	0	1	1	1	1	1	1	1
15	0	0	0	0	1	1	1	1	1	1	1	1
24	1	0	0	1	1	1	1	1	1	1	1	1
25	1	0	1	1	1	1	1	1	1	1	1	1

Logistic Regression: Decision Threshold

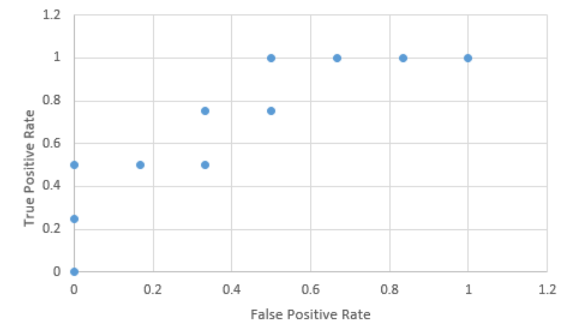


	50	25	24	15	13	12	8	5	3	2	1
TP	0	1	2	2	2	3	3	4	4	4	4
FP	0	0	0	1	2	2	3	3	4	5	6
TN	6	6	6	5	4	4	3	3	2	1	0
FN	4	3	2	2	2	1	1	0	0	0	0
TPR	0	0.25	0.5	0.5	0.5	0.75	0.75	1	1	1	1
FPR	0	0	0	0.166667	0.333333	0.333333	0.5	0.5	0.666667	0.833333	1

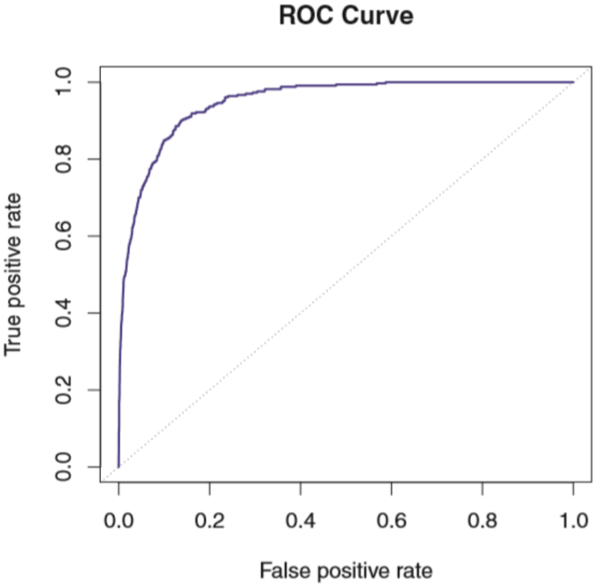
Logistic Regression: Decision Threshold



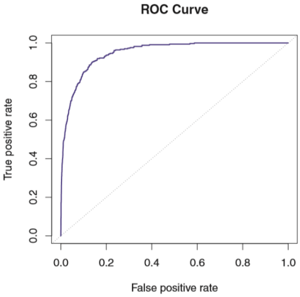
	50	25	24	15	13	12	8	5	3	2	1
TP	0	1	2	2	2	3	3	4	4	4	4
FP	0	0	0	1	2	2	3	3	4	5	6
TN	6	6	6	5	4	4	3	3	2	1	0
FN	4	3	2	2	2	1	1	0	0	0	0
TPR	0	0.25	0.5	0.5	0.5	0.75	0.75	1	1	1	1
FPR	0	0	0	0.166667	0.333333	0.333333	0.5	0.5	0.666667	0.833333	1



Receiver Operating Characteristic Curve



ROC and Measurements



- By predicting probability instead of label can generate ROC Curve
- Can choose threshold on ROC curve that optimizes some threshold-specific Measurement
- Can also plot Precision-Recall Curve
- Area Under ROC Curve (AUROC) is a measurement of how well your model separates classes (without taking decision threshold into account)
- Can also calculate AUPRC

Problem Solving: Gradient Descent

$$MSE(\beta_0, \beta_1) = J(\beta_0, \beta_1) = J(\boldsymbol{\beta}) = \frac{1}{n}\sum_{i=1}^{n}(y_i - (\beta_0 + \beta_1 x_i))^2$$

- Optimize β_0 and β_1 by Gradient Descent
- Remember, Update Rules $\beta_0^{t+1} = \beta_0^t - \alpha \frac{\partial}{\partial \beta_0} J(\beta_0, \beta_1)$
- Remember, Update Rules $\beta_1^{t+1} = \beta_1^t - \alpha \frac{\partial}{\partial \beta_1} J(\beta_0, \beta_1)$
- if $\boldsymbol{\beta} = (\beta_0, \beta_1)$ then
- $\boldsymbol{\beta}^{t+1} = \boldsymbol{\beta}^k - \alpha \nabla J(\boldsymbol{\beta})$

Problem Solving: Jacobian

$$\nabla J(\boldsymbol{\beta})$$

- If we have a series of functions f that map input x to an output y (such as linear regression)
- The Jacobian is a matrix such that $J_{ij} = \frac{\partial f_i}{\partial x_j}$
- The Hessian is a matrix such that $H_{ij} = \frac{\partial^2 f}{\partial x_i \partial x_j}$

Problem Solving: Jacobian

$$f(x, y) = \begin{bmatrix} x^2 y \\ 5x + \sin(y) \end{bmatrix}$$

$$\nabla f = \begin{bmatrix} \frac{\partial f_1}{\partial x} & \frac{\partial f_1}{\partial y} \\ \frac{\partial f_2}{\partial x} & \frac{\partial f_2}{\partial y} \end{bmatrix} = \begin{bmatrix} 2xy & x^2 \\ 5 & \cos(y) \end{bmatrix}$$

Problem Solving: Gradient Descent

$$MSE(\beta_0, \beta_1) = J(\beta_0, \beta_1) = J(\boldsymbol{\beta}) = \frac{1}{2n}\sum_{i=1}^{n}((\beta_0 + \beta_1 x_i) - y_i)^2$$

Our Objective

$$\min_{\beta_0, \beta_1} J(\beta_0, \beta_1) = \min_{\beta} J(\beta)$$

Then our update rule at time $t+1$, with stopping criteria ϵ is

$$\beta^{t+1} = \beta^t - \alpha \nabla J(\beta)$$

$$\nabla J(\beta) = \begin{bmatrix} \frac{\partial J(\beta)}{\partial \beta_0} & \frac{\partial J(\beta)}{\partial \beta_1} \end{bmatrix}$$

$$\|\nabla J(\beta)\| < \epsilon$$

Problem Solving: Gradient Descent

$$\nabla J(\beta) = \left[\frac{\partial J(\beta)}{\partial \beta_0} \quad \frac{\partial J(\beta)}{\partial \beta_1} \right]$$

$$= \left[\frac{1}{n} \sum_{i=1}^{n} (\beta_0 + \beta_1 x_i - y_i) \quad \frac{1}{n} \sum_{i=1}^{n} (\beta_0 + \beta_1 x_i - y_i) x_i \quad \right]$$

Takeaways and Next Time

- Model selection finds optimal solutions by using a subset of predictors
- Regularization minimizes RSS subject to a price for the predictors used
- Model Performance and Problem Solving
- Next Time: Measures of Performance with Logistic Regression