Outline

OLS issues

There are few weaknesses with OLS regression:

- may be unduly influenced by outliers
- restriction to linear models
- for even moderate number of possible predictors, selecting the "best" model is not easy
  - there are $\sum_{r=1}^{p} \binom{p}{r}$ possible models
  - stepwise techniques are myopic
- if $p > n$ then OLS will fail
- multicollinearity makes interpretation difficult
- no "tuning" parameters

robust regression

We discussed different options for dealing with outliers. One option is to adjust our modeling method to be more “robust”.

robustness

insensitivity to small deviations from the assumptions the model imposes on the data

distributional robustness

robustness w.r.t. skewed distributions and residual outliers

“Downweighting” outliers is the basis behind robust techniques.

least absolute value regression

OLS is heavily influenced by large outliers in Y due to the squared term in the fitting objective function:

$$\min \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

Least absolute values (LAV) regression modifies the objective function, replacing the squared term with an absolute value:

$$\min \sum_{i=1}^{n} |y_i - \hat{y}_i|$$

least absolute value regression

LAV is a.k.a. L_1 regression or median-regression
(note: OLS is also called L_2 regression)

- more resistant in outliers in Y than OLS
- may perform worse than OLS if outliers are in X (e.g. if points have high leverage)
- less efficient than OLS (standard errors are higher)
- requires more complex computation

M-estimation

M-estimation is a general technique in which the estimates for β are determined by minimizing a function of the residuals: $\rho(e)$

$$\min \sum_{i=1}^{n} \rho(e_i) = \sum_{i=1}^{n} \rho(y_i - \hat{y}_i)$$

The function ρ should have the following properties:

- non-negativity: $\rho(e) > 0$
- equal to 0 when its argument is 0: $\rho(0) = 0$
- symmetric: $\rho(e) = \rho(-e)$
- monotone in $|e_i|$: $\rho(e_i) \geq \rho(e_j)$ for $|e_i| > |e_j|$

Huber loss function

A compromise of OLS and LAV is M-estimation with a Huber loss function.

For small residuals, it is like OLS; for larger residuals, it is like LAV.

$$\rho(e_i) = \begin{cases} \frac{1}{2}e_i^2 & \text{if } |e_i| \leq k \\ k|e_i| - \frac{1}{2}k^2 & \text{if } |e_i| > k \end{cases}$$

where k is a threshold value for "large" residuals expressed in terms of the "spread" of Y which is based on the median absolute deviation.

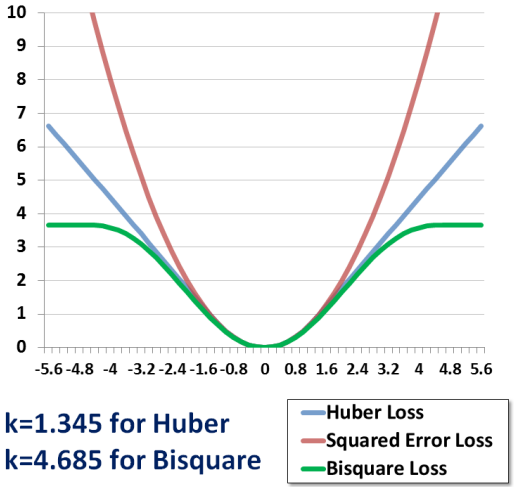
Tukey's bisquare loss function

Tukey's bisquare is especially resistant to extreme values.

If $|e_i| \leq k$,

$$\rho(e_i) = \frac{k^2}{6}\left(1 - \left(1 - \left(\frac{e_i}{k}\right)^2\right)^3\right)$$

Otherwise, $\rho(e_i) = \frac{k^2}{6}$



M-estimation parameter estimates

Determining the β parameters for M-estimation is more complex than OLS.

Minimizing the associated loss function (Huber or Bisquare) requires assigning a different weight to each observation depending on the size of the residual. However, the residuals are not known until after fitting an intial regression...

An iterative solution is required (*iterative reweighted least squares*)

As the absolute residual goes up, the weight goes down.

e.g., Zambia (from the `LifeCycleSavings` data) is "downweighted" by a factor of 0.4720 (Huber) or 0.3756 (bisquare)

Cases not shown have a weight of 1 for Huber; the weights increase towards 1 for the bisquare weights.

In OLS, all cases (always) have a weight of 1.

	Huber weight	bisquare weight
Zambia	0.4720	0.3756
Chile	0.5846	0.5743
Philippines	0.6890	0.6728
Peru	0.7092	0.6903
Iceland	0.7611	0.7283
Paraguay	0.7886	0.7523
Korea	0.8086	0.7677
Japan	0.8794	0.7838
Costa Rica	0.8828	0.7916
Denmark	0.8910	0.8000
Sweden	1.0000	0.8568
Brazil	1.0000	0.8887

comparing techniques

When to use OLS, M-estimation with Huber loss, or M-estimation with bisquare loss?

- if robust regression results very different from OLS; use robust regression
- Huber weights may have difficulty with severe outliers
- bisquare weights may have difficulty converging

PCR

OLS has a problem if:

- the number of predictors p is large; especially if p is close to n;
- predictors are correlated

One immediate solution to this is to use PCA

- to reduce the dimension of the input data X
- to eliminate multicollinearity

This is called principal component regression (PCR).

PCR

A couple of issues:

- can be difficult to interpret the meaning of the model
- PCA is unsupervised technique – it is not associated with the outcome variable Y at all!
- i.e., dimension reduction via PCA does not necessarily produce new predictors that explain the response.

PLS

Partial least squares (PLS) is basically a supervised version of PCA – designed specifically for regression.

- PLS constructs a set of linear combinations of the inputs X called *components*
- it uses both X and Y in the iterative construction of the components
- the $m \leq p$ components are chosen to maximally summarize the covariance with Y

Similar to PCA, in PLS, the inputs should be centered and scaled.

The complexity tuning parameter is the number of components, m.

PLS algorithm

1. standardize each $X_j \in X$; set $\hat{y}^{(0)} = \mathbf{1}\bar{y}$;
   set $X_j^{(0)} = X_j$ for $j = 1 \ldots p$
2. For iterations $m = 1, 2, \ldots, p$
   - $Z_m = \sum_{j=1}^{p} \hat{\beta}_{mj} X_j^{(m-1)}$, where $\hat{\beta}_{mj}$ is from fitting $Y = f\left(X_j^{(m-1)}\right)$
   - fit $Y = f(Z_m)$ to find $\hat{B}_m$
   - Update $\hat{y}$: $\hat{y}^{(m)} \leftarrow \hat{y}^{(m-1)} + \hat{B}_m Z_m$
   - Orthogonalize each $X_j^{(m-1)}$ w.r.t. Z_m: $X_j^{(m-1)} \leftarrow X_j^{(m-1)} - \hat{\gamma}_m Z_m$
     where $\hat{\gamma}_m$ is coefficient from fit: $X_j^{(m-1)} = f(Z_m)$
3. Output the sequence of fitted vectors $\{\hat{y}^{(m)}\}_1^p$;
   recover $\hat{\beta}^{\text{pls}}$ from $\hat{y}^{(m)} = \mathbf{X}\hat{\beta}^{\text{pls}}$

Adapted from Algorithm 3.2 in Hastie et al., *The Elements of Statistical Learning*, Chapter 3, Section 3.4.4

PCR vs. PLS

- `solubility` data from `AppliedPredictiveModeling`
- 1,267 molecular compounds described by predictors: 208 binary, 16 count descriptors, 4 continuous
- goal: predict “solubility”
- at right: PCA and PLS component’s correlation with outcome

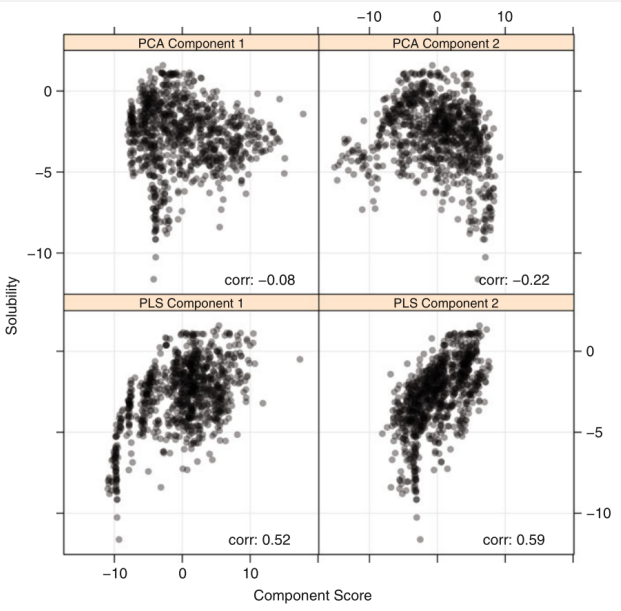


Table 1
Summary of the literature review

Reference	Topic	Conclusions and *quotations*
[7]	Theoretical-Simulations	PCR and PLS gave similar optimal prediction results. PLS uses fewer LVs than PCR.
[3]	Theoretical	"PLS is one of a continuum of variations similar to PCR making specific implicit assumptions about the importance of each principal component for describing the dependant variable. It is inherently no better than those methods formed by selecting individual eigenvectors."
[4]	Theoretical-Simulations	PLS often reached its minimal mean squared error of prediction using fewer factors than PCR.
[8]	Simulations	"PCR and PLS are very similar"..."PLS seems to predict better than PCR in the cases when there are random linear baselines or independently varying major spectral components which overlap with the spectral features of the analyte."
[9]	Theoretical	"It has been proven that the intuitive inequality $R^2_{PLS} \geq R^2_{PCR}$ holds for any given dimensionality...it should be emphasized that a better goodness of fit, as expressed by R^2, does not necessarily transcend into superior prediction performance."

penalized regression

- When a model overfits the data, or when there are issues with collinearity, the parameter estimates may become inflated.
- When retaining only a subset of predictors (e.g. through stepwise techniques), a model's variance tends to increase.
- Certain penalized regression techniques (specifically *regularization* methods) can mitigate these issues.
- These methods "control" the magnitude of the parameter estimates – penalizing models with larger values.

ridge regression

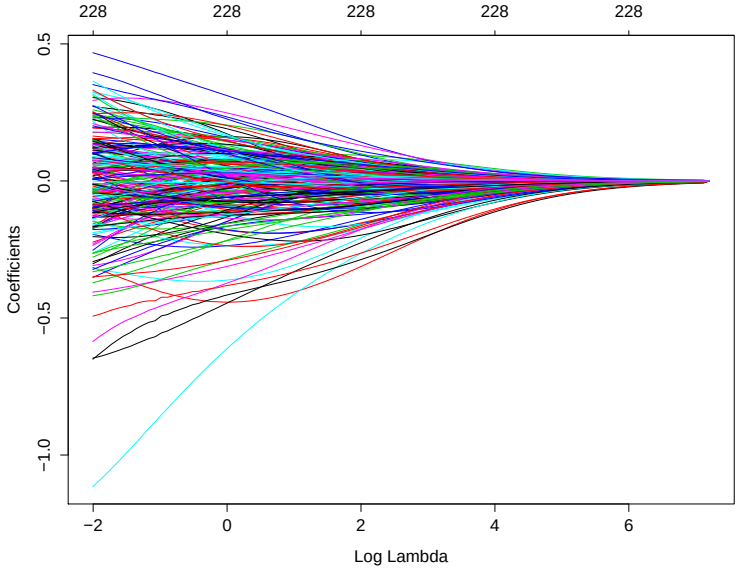
In OLS, to estimate β we minimize:

$$\sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

In *ridge regression*, we minimize:

$$\sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \lambda \sum_{j=1}^{p} \hat{\beta}_j^2$$

where $\lambda > 0$ is a tuning parameter to scale the penalty.



Shrinkage of coefficients in ridge regression

ridge regression

more importantly...

- for $\lambda = 0$, ridge = OLS
- the CV RMSE error is reduced for $\lambda > 0$!

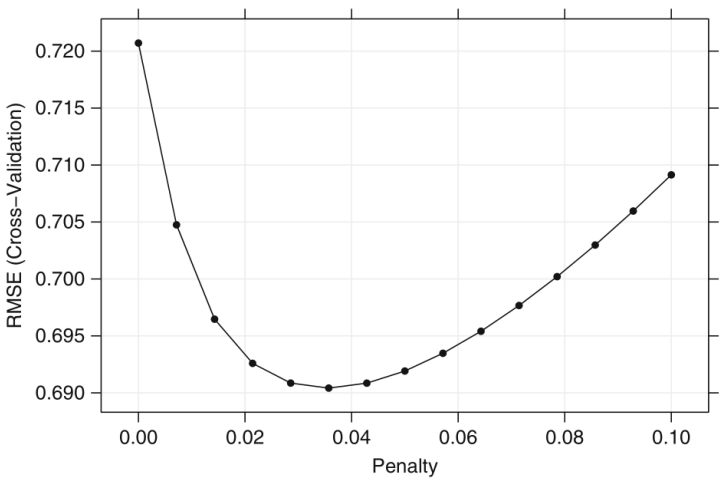


Fig. 6.16: The cross-validation profiles for a ridge regression model

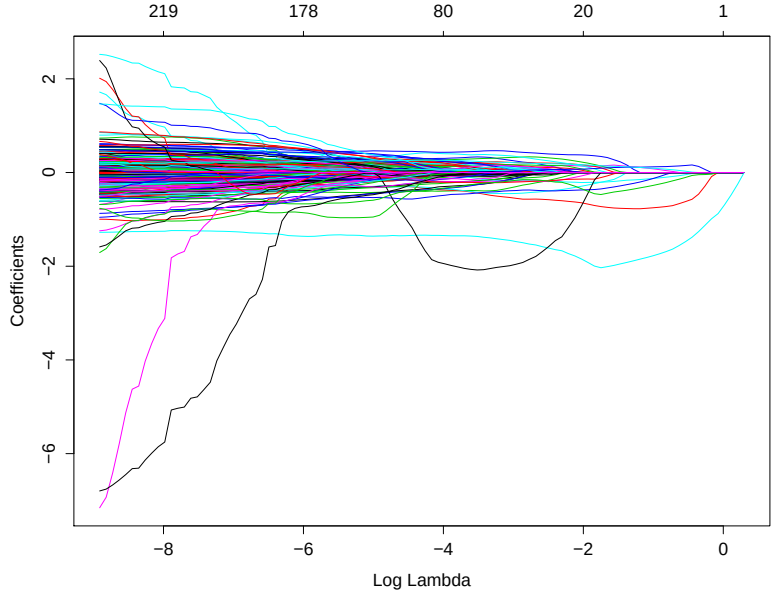
LASSO

Ridge regression does not help with feature selection; however, the *least absolute shrinkage and selection operator* (LASSO) model does help.

In LASSO, we minimize:

$$\sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \lambda \sum_{j=1}^{p} |\hat{\beta}_j|$$

where $\lambda > 0$ is a tuning parameter to scale the penalty.



Shrinkage of coefficients in LASSO

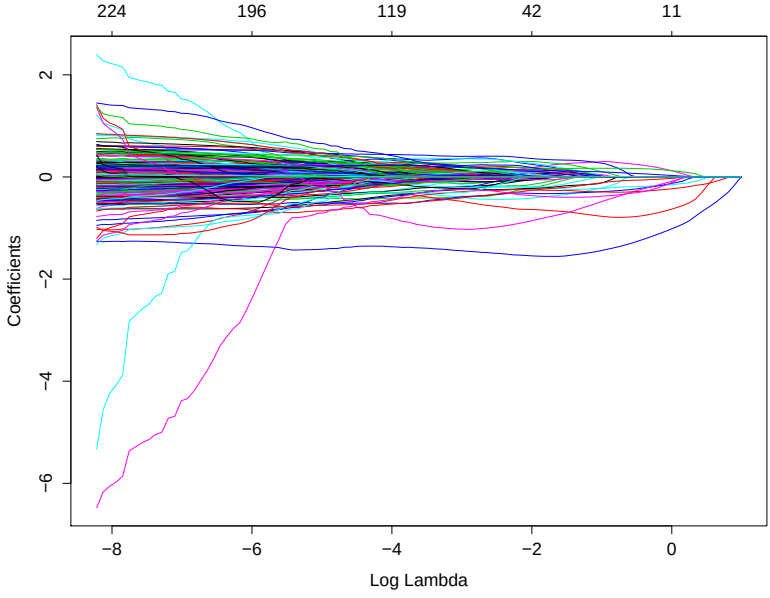
elastic net regression

A generalization of LASSO is the *elastic net* which combines the two types of penalties:

In elastic net, we minimize:

$$\sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \lambda_1 \sum_{j=1}^{p} |\hat{\beta}_j| + \lambda_2 \sum_{j=1}^{p} \hat{\beta}_j^2$$

where $\lambda_1, \lambda_2 > 0$ are tuning parameters to scale the penalties.



Shrinkage of coefficients in elastic net regression

comparison

- ridge regression does not select variables
- LASSO does, but the selection between two highly correlated variables in almost random
- elastic net enables the regularization of ridge *and* selects variables – so it useful in case of highly correlated predictors
- for the `solubility` data, LASSO had best CV RMSE

The following slides regarding **support vector machine regression** is additional and entirely optional material – it is not required for the class.

The material is adapted from Chapter 7, section 7.3 from *Applied Predictive Modeling*.

SVM

Support vector machines (SVM) are a class of very powerful, and highly flexible modeling techniques.

- originally developed for classification; extended to predictive regression
- not limited to only linear models
- robust to outliers
- different type of penalized regression
- we will look a type of SVM called ϵ-*insensitive regression*

SVM objective

$$\text{minimize: } c \sum_{i=1}^{n} L_\epsilon (y_i - \hat{y}_i) + \sum_{j=1}^{p} \hat{\beta}_j^2$$

where c is a cost parameter and $L_\epsilon(\cdot)$ is a loss function.

The cost penalty parameter is associated with the residuals; not the regression coefficients (like ridge, LASSO, and elastic net).

SVM ϵ-*insensitive* loss function

$$L_\epsilon(y_i - \hat{y}_i) = \begin{cases} 0 & \text{if } |y_i - \hat{y}_i| \leq \epsilon \\ |y_i - \hat{y}_i| - \epsilon & \text{otherwise} \end{cases}$$

With this loss function:

- points with small residuals do not contribute to regression fit!
- larger residuals contribute a linear amount (like Huber loss)

SVM prediction

With OLS, to predict a new data point **u**, the equation is:

$$\hat{y} = \hat{\beta}_0 + \sum_{j=1}^{p} \hat{\beta}_j u_j$$

SVM prediction

SVM prediction is similar.

The parameter estimates can be written as a function of a set of unknown parameters α_i and the training samples.

$$\begin{aligned}\hat{y} &= \hat{\beta}_0 + \sum_{j=1}^{p} \hat{\beta}_j u_j \\ &= \hat{\beta}_0 + \sum_{j=1}^{p} \sum_{i=1}^{n} \alpha_i x_{ji} u_j \\ &= \hat{\beta}_0 + \sum_{i=1}^{n} \alpha_i \left(\sum_{j=1}^{p} x_{ji} u_j \right)\end{aligned}$$

SVM prediction

$$\hat{y} = \hat{\beta}_0 + \sum_{i=1}^{n} \alpha_i \left(\sum_{j=1}^{p} x_{ji} u_j \right)$$

A few things worth noting from this equation:

- there are as many α_i parameters as there are data points!
- the individual training data **X** required to make a prediction
- only points with larger residuals are used for the prediction!
- training points with $\alpha \neq 0$ are called the support vectors

SVM prediction

$$\hat{y} = \hat{\beta}_0 + \sum_{i=1}^{n} \alpha_i \left(\sum_{j=1}^{p} x_{ji} u_j \right)$$

This can be re-written more generally as:

$$f(\mathbf{u}) = \hat{\beta}_0 + \sum_{i=1}^{n} \alpha_i K(X_{.i}, \mathbf{u})$$

where $K(\cdot)$ is called a *kernel function*.

The linear kernel is: $K(X_{.i}, \mathbf{u}) = \sum_{j=1}^{p} x_{ji} u_j = X'_{.i}\mathbf{u}$

SVM kernels

There are other possible kernel functions.

- training data can enter prediction model in a non-linear way
- leads to non-linear models (and thus called the “kernel trick”)

Non-linear kernels:

$$\begin{aligned} \text{polynomial:} &\quad (\phi \mathbf{x}'\mathbf{u} + 1)^{degree} \\ \text{radial basis function:} &\quad \exp\left(-\sigma ||\mathbf{x} - \mathbf{u}||_2^2\right) \\ \text{hyperbolic tangent:} &\quad \tanh(\phi \mathbf{x}'\mathbf{u} + 1) \end{aligned}$$

where ϕ and σ are scaling parameters.

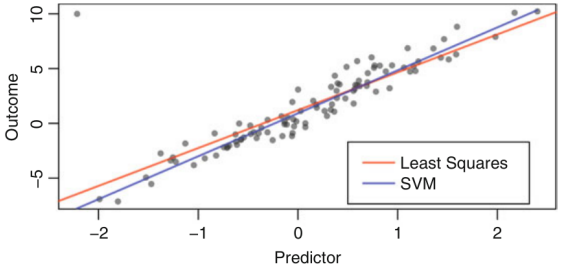
SVM example

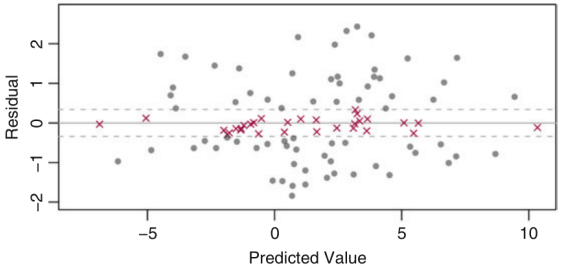
top graph:

- OLS model (red) is impacted by one outlier
- SVM model (blue) less sensitive

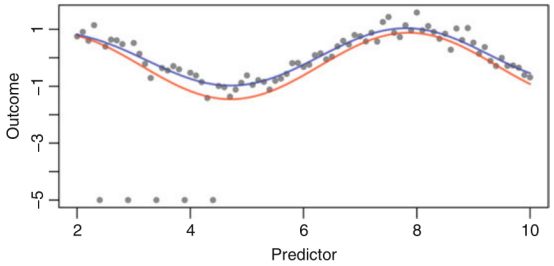
bottom graph:

- support vectors indicated in gray circles
- points not used in fit in red crosses





SVM example



In this example the data is non-linear data (sine function) with some outliers.

- an OLS model (red) was fit with a sine transformed predictor
- the SLM model (blue) was fit without an transformation only using the "kernel trick" with the radial basis function

related R code and packages

approach	model type	package	function
robust	least absolute value	quantreg	rq
	M-estimation	MASS	rlm
component-based	PCR	pls	pcr
	PLS	pls	plsr
penalized	ridge regression	MASS	lm.ridge
	LASSO	lars	lars
	elastic net	elasticnet	enet
non-linear	SVM regression	kernlab	ksvm