STAT540-Lecture 18: Supervised Learning II

Dr. Gabriela Cohen Freue
Department of Statistics, UBC

March 4th, 2019

Recall: ”Supervised Learning Machine”

- We have a **training set** of n cases in which we know the values of both y and $\mathbf{X}$.
- Based on the training data, we will produce a model or a **rule** to predict y. These can be parametric or non-parametric.
- We want to predict y on new samples of a **test set** for which only $\mathbf{X}$ is known.

How can we find the best-trained rule?

Goal: prediction!

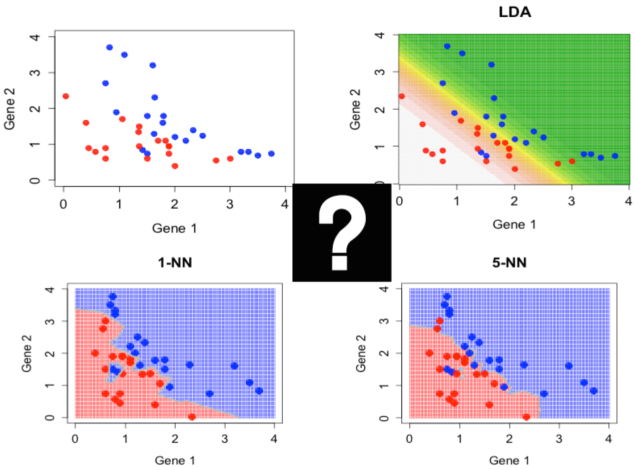
of which samples??

How do we define best?

and based on what information??

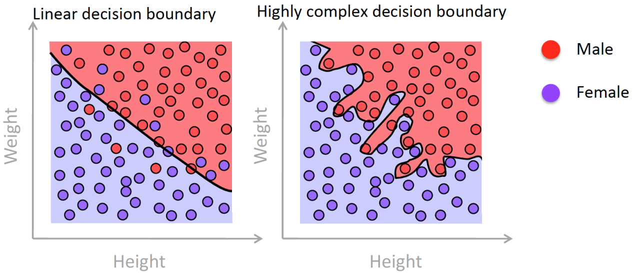
Many methods can be used to build a classifier or a model, each probably depending on tuning constants.

Which one is better?

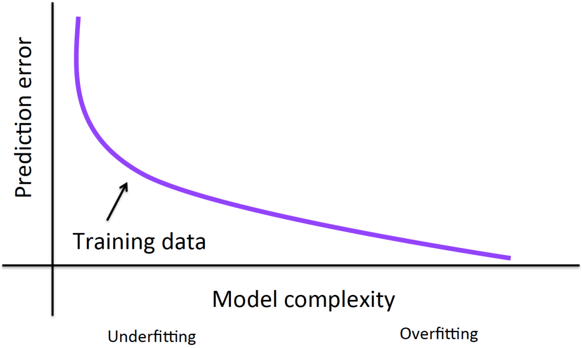


Overfitting

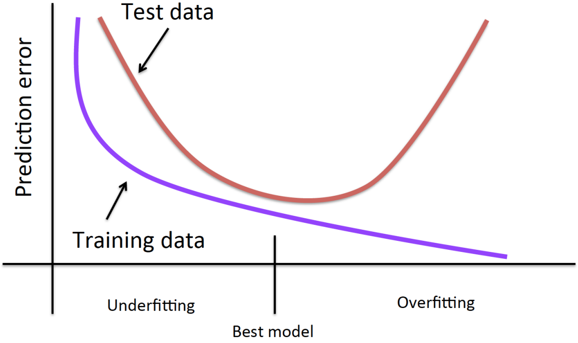
If we allow very complicated predictors, we could overfit the training data:



Error and model complexity



Error and model complexity



But ...

But ...

How do we find *a* test set?

But ...

How do we find *a* test set?

We can not use the same test set to select a model and test it!!

Ideally, we want to train the model on some samples and use other samples to test the model:

- Training set
- Validation set
- Test set

If an independent test set is not available, we need to use part of our available data to create one. How?

But ...

How do we find *a* test set?

We can not use the same test set to select a model and test it!!

Ideally, we want to train the model on some samples and use other samples to test the model:

- Training set
- Validation set
- Test set

If an independent test set is not available, we need to use part of our available data to create one. How?

cross validation

k-fold cross validation (k-fold CV)

Partition the training data T into k separate sets of (almost) equal size (k folds): $T_1, T_2, \ldots T_k$

1. Set aside one folds (T_i) and use it as a ”validation” (test) set. Use all remaining folds to build and train your model
2. Compute the fitted values for the observations in T_i, based on the model built excluding this fold in 1). Call these $\hat{y}_{j(-i)}$
3. Compute the CV error for the ith fold

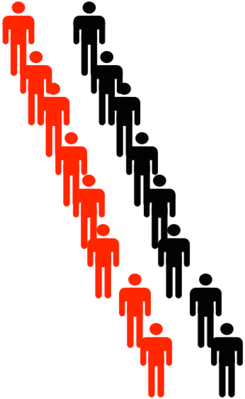
$$\text{CV Error}_i = \frac{1}{n_i} \sum_{j \in T_i} L(y_j, \hat{y}_{j(-i)})$$

   How ”far” are our predictions, $\hat{y}_{j(-i)}$, from the true response values, y_j in this test set, T_i

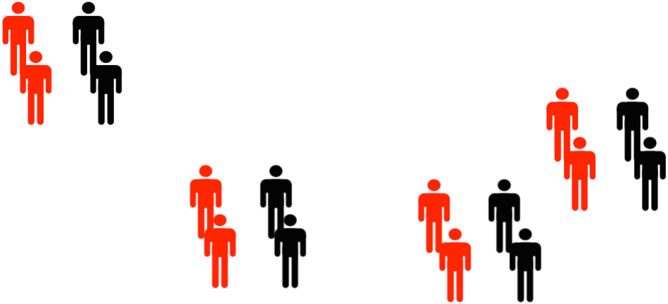
4. Repeat 2-3 for all folds and average all CV errors

$$\text{CV Error} = \frac{1}{k} \sum_{i=1}^{k} \text{CV Error}_i$$

Example: *4-fold CV*

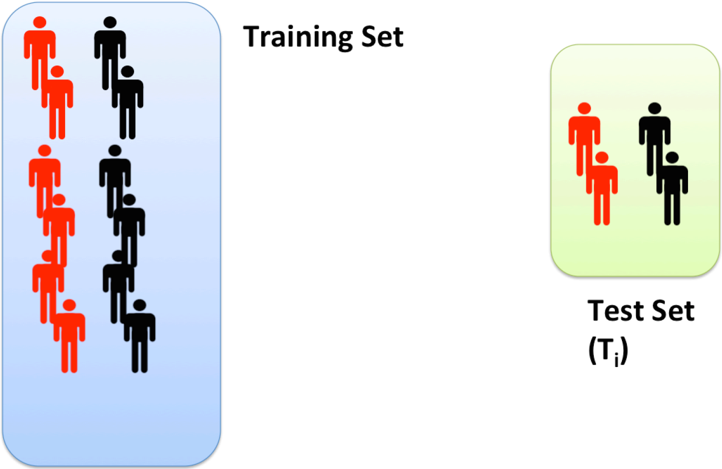


Example: *4-fold CV*



Example: *4-fold CV*

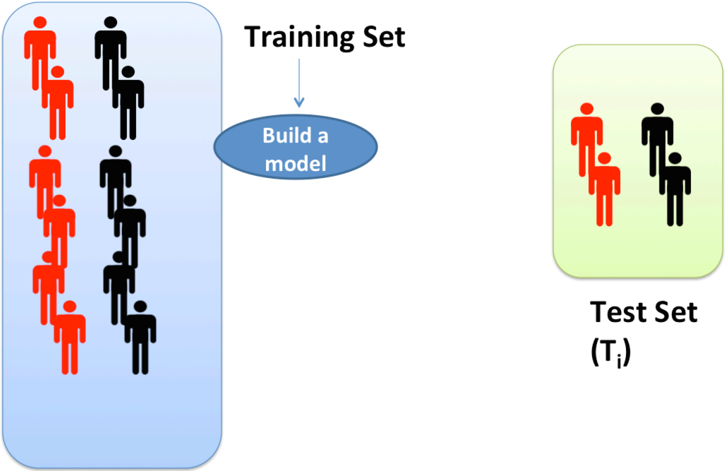
Example: *4-fold CV*



Training Set

Test Set (T_i)

Example: *4-fold CV*

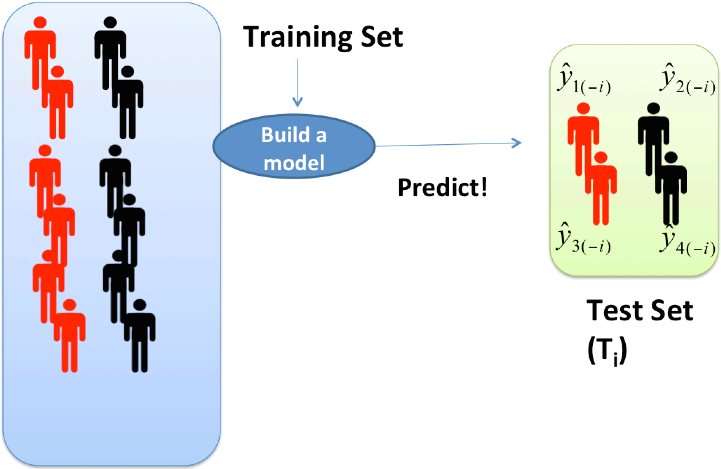


Training Set

Build a model

Test Set (T_i)

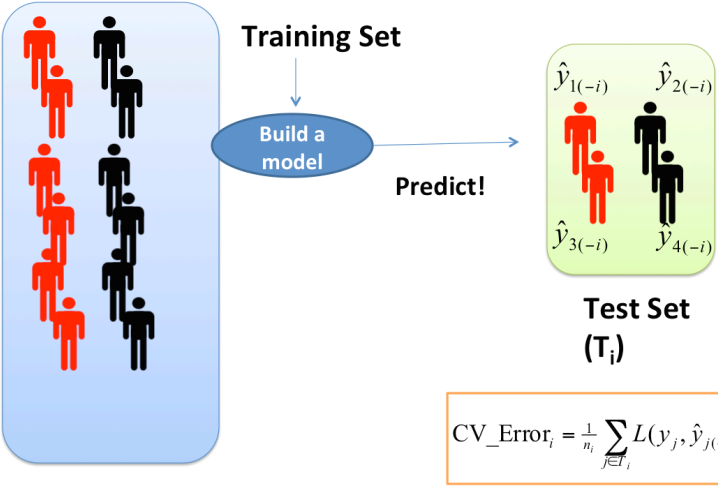
Example: *4-fold CV*



Test Set

(T_i)

Example: *4-fold CV*



Training Set

Build a model

Predict!

Test Set (T_i)

$$\text{CV\_Error}_i = \tfrac{1}{n_i} \sum_{j \in T_i} L(y_j, \hat{y}_{j(-i)})$$

Remarks on CV

- There exists a trade-off between bias and variance of the estimated CV errors
  - The smaller the k, the lower the variance but potentially at a cost of a higher bias
  - Run several k-fold CV (e.g., 100 times) and average the CV errors from each run
- The case of n-fold CV is usually called leave-one-out cross validation. There is no randomness but its CV error estimate may be highly variable

Goal: prediction!

of which samples??

How do we define best?

and based on what information??

Performance for continuous response

Examples:

- Squared error loss

$$L(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \|\boldsymbol{y} - \hat{\boldsymbol{y}}\|_2^2 = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

- Absolute error loss

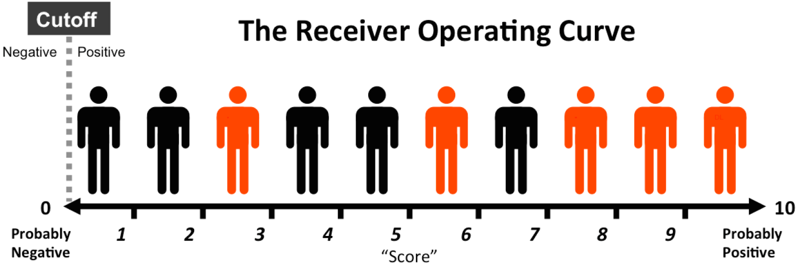
$$L(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \|\boldsymbol{y} - \hat{\boldsymbol{y}}\|_1 = \sum_{i=1}^{n} |y_i - \hat{y}_i|$$

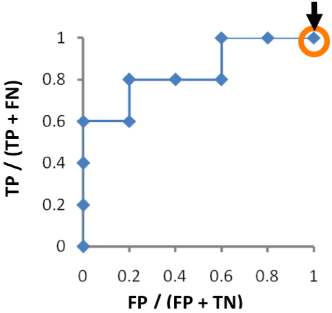
Performance measures for binary classifiers

- True positive (TP)
- True negative (TN)
- False positive (?Type I error?) (FP)
- False negative (?Type II error?) (FN)

- Accuracy (TP + TN)/N (=1-error rate)
- Sensitivity: TP/(TP+FN) (recall)
- Specificity: TN/(TN + FP)

The Receiver Operating Curve





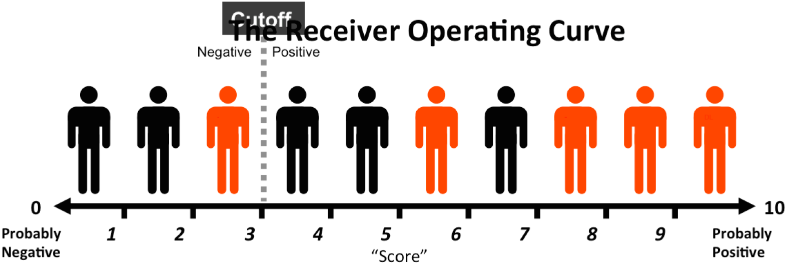
sensitivity

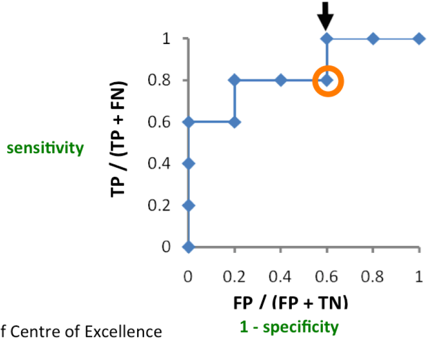
1 - specificity

Cutoff	FP	TP	FN	TN
0	5	5	0	0
1	4	5	0	1
2	3	5	0	2
3	3	4	1	2
4	2	4	1	3
5	1	4	1	4
6	1	3	2	4
7	0	3	2	5
8	0	2	3	5
9	0	1	4	5
10	0	0	5	5

Proof Centre of Excellence

The Receiver Operating Curve

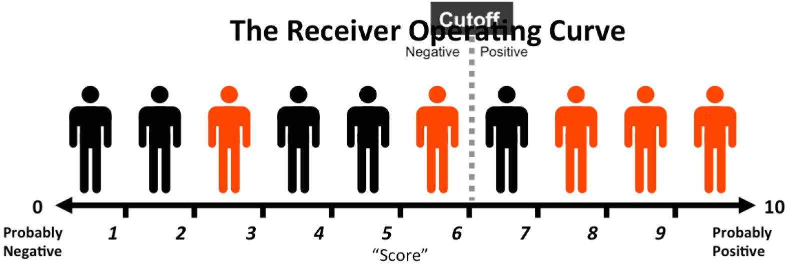


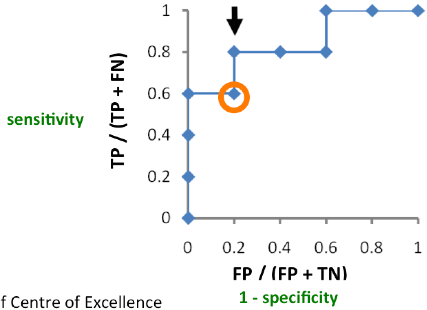


Cutoff	FP	TP	FN	TN
0	5	5	0	0
1	4	5	0	1
2	3	5	0	2
3	3	4	1	2
4	2	4	1	3
5	1	4	1	4
6	1	3	2	4
7	0	3	2	5
8	0	2	3	5
9	0	1	4	5
10	0	0	5	5

Proof Centre of Excellence

The Receiver Operating Curve

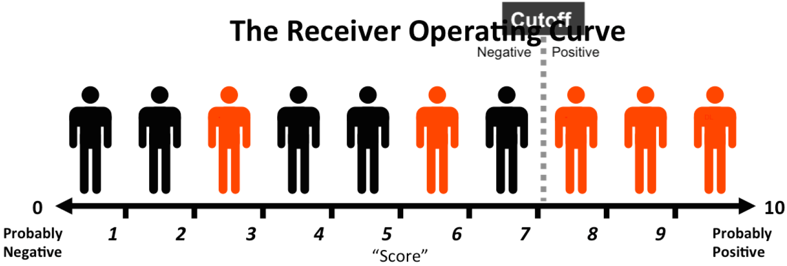




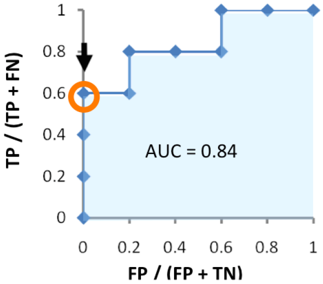
Cutoff	FP	TP	FN	TN
0	5	5	0	0
1	4	5	0	1
2	3	5	0	2
3	3	4	1	2
4	2	4	1	3
5	1	4	1	4
6	1	3	2	4
7	0	3	2	5
8	0	2	3	5
9	0	1	4	5
10	0	0	5	5

Proof Centre of Excellence

The Receiver Operating Curve



sensitivity



1 - specificity

Cutoff	FP	TP	FN	TN
0	5	5	0	0
1	4	5	0	1
2	3	5	0	2
3	3	4	1	2
4	2	4	1	3
5	1	4	1	4
6	1	3	2	4
7	0	3	2	5
8	0	2	3	5
9	0	1	4	5
10	0	0	5	5

Proof Centre of Excellence

Conclusions so far

- **New goal**: predict! (classification or continuous response)
- If a test set is not available, cross validation can be used to validate and/test the prediction model.
- There is a trade-off between bias and variance of the estimated CV errors.
- Different performance measures can be used to evaluate the prediction model.

Goal: prediction!

of which samples??

How do we define best?

and based on what information??

- In -omics studies p $>>$ n (i.e., the number of potential covariates is much larger than the number of samples) e.g., $\sim$54000 genes and $\sim$100 samples
- Among the p covariates available, many may be highly correlated, do we need all cousins in the party? e.g., many genes from a common pathway

- In -omics studies p $>>$ n (i.e., the number of potential covariates is much larger than the number of samples) e.g., $\sim$54000 genes and $\sim$100 samples
- Among the p covariates available, many may be highly correlated, do we need all cousins in the party? e.g., many genes from a common pathway

Which covariates should be included in the model??

- In -omics studies p $>>$ n (i.e., the number of potential covariates is much larger than the number of samples) e.g., $\sim$54000 genes and $\sim$100 samples
- Among the p covariates available, many may be highly correlated, do we need all cousins in the party? e.g., many genes from a common pathway

Which covariates should be included in the model??

Model Selection

Linear Regression Models

Assume a linear model of the form:

$$\mathbf{y} = \boldsymbol{\beta}^T \mathbf{x} + \varepsilon$$

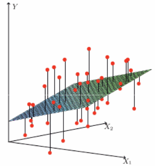


FIGURE 3.1. *Linear least squares fitting with $X \in \mathbb{R}^2$. We seek the linear function of X that minimizes the sum of squared residuals from Y.*

Then, the **Least Squares** (LS) estimators of β are those that minimize the residuals sum-of-squares (RSS) [1]:

$$\hat{\boldsymbol{\beta}} = \min_{\boldsymbol{\beta} \in \mathbb{R}^p} \sum_{i=1}^{n} \left(y_i - \boldsymbol{\beta}^T \mathbf{x}_i \right)^2 = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y}$$

In matrix notation:

$$\text{RSS} = (\mathbf{y} - \boldsymbol{\beta}^T \mathbf{X})^T (\mathbf{y} - \boldsymbol{\beta}^T \mathbf{X}) = (\boldsymbol{\beta} - \hat{\boldsymbol{\beta}})^T \mathbf{X}^T \mathbf{X} (\boldsymbol{\beta} - \hat{\boldsymbol{\beta}})$$

[1] Note that the intercept can be capture in $\boldsymbol{\beta}$ adding a column of 1s to $\mathbf{x}$.Image: The Elements of Statistical Learning, Hastie, Tibshirani, Friedman

Pros and Cons of LS

- $\hat{\beta}$ has the smallest variance among all linear unbiased estimates.
- However, if we sacrifice bias, there may be other estimators with lower variance
- Lower variance may result in better prediction accuracy (remember that prediction is our main goal!!)
- The model may be difficult to interpret when there are *many* covariates (e.g., 1000 genes ?!)
- When covariates are highly correlated, the LS estimators can become very unstable

Pros and Cons of LS

- $\hat{\beta}$ has the smallest variance among all linear unbiased estimates.
- However, if we sacrifice bias, there may be other estimators with lower variance
- Lower variance may result in better prediction accuracy (remember that prediction is our main goal!!)
- The model may be difficult to interpret when there are *many* covariates (e.g., 1000 genes ?!)
- When covariates are highly correlated, the LS estimators can become very unstable

Model Selection

Feature selection

- The "filtering" approach
  - Filter features *without* looking at the labels/response: e.g., only keep genes with STD/mean above a threshold
  - Filter features *considering* the labels/response: e.g., only keep genes significantly associated with response
    CAREFUL! This is part of building a model/classifier
- The "wrapper" approach
  - Identifying features that lead to good performance
    CAREFUL! This is part of building a model/classifier
- The "regularized" approach
  - Modify the objective function to identify a reduced feature set intrinsically

Model / feature selection - Regularized methods

They select coefficients in a more continuous way by adding a bound (penalty) to their size (belong to the family of *regularized* methods).

Examples:

- Ridge regression (Hoerl, A.E. and Kennard, R., *Technometrics*, 1970).
- Lasso regression (Tibshirani, *JRSS*, 1996)
- Elastics Net (Zou and Hastie, *JRSS* 2005)

Ridge regression

The ridge coefficients minimize a *penalized* RSS, i.e.,

$$(\hat{\alpha}, \hat{\boldsymbol{\beta}})_R \;=\; \min_{\alpha, \boldsymbol{\beta} \in \mathbb{R}^p} \sum_{i=1}^{n} \left(y_i - \alpha - \boldsymbol{\beta}^T \mathbf{x}_i \right)^2$$

subject to

$$\sum_{j=1}^{p} \boldsymbol{\beta}_j^2 \;\leq\; C \;\; \text{for some } C > 0.$$

Given this penalty on the parameters, the estimated coefficient will not adjust to different scales in the covariates.

- Thus, we need to standardize the covariates before using regularized methods.
- It can be shown that if we center the predictors, then $\hat{\alpha}_R = \bar{y}$, and $\hat{\boldsymbol{\beta}}_R$ can be estimated separately.

Ridge regression (cont.)

The ridge (penalized) residual sum-of-squares is given by

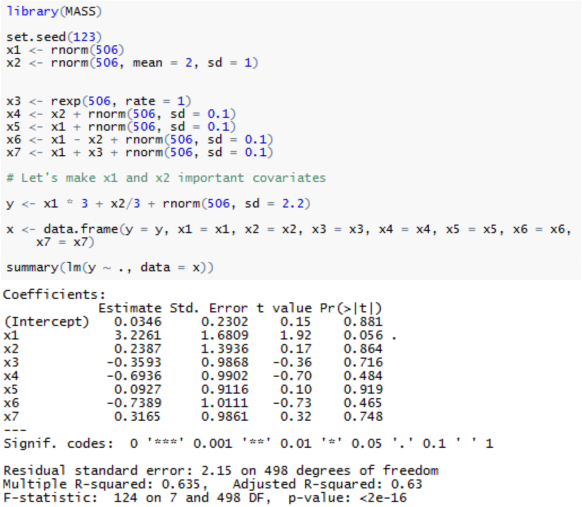
$$\sum_{i=1}^{n} \left(y_i - \alpha - \boldsymbol{\beta}^T \mathbf{x}_i\right)^2 + \lambda \sum_{j=1}^{p} \boldsymbol{\beta}_j^2$$

If $\mathbf{x}_i$ are centered, the Ridge estimator has a closed-form:

$$\hat{\boldsymbol{\beta}}_R = \left(\mathbf{X}^T \mathbf{X} + \lambda \mathbf{I}\right)^{-1} \mathbf{X}^T \mathbf{y}$$

- There is a solution for each λ (path of solutions).
- λ controls the size of the coefficients (i.e., amount of regularization). If $\lambda = 0$, $\hat{\boldsymbol{\beta}}_R = \hat{\boldsymbol{\beta}}_{LS}$. The $\hat{\boldsymbol{\beta}}_R$ shrink as λ increases and tend to 0 as λ goes to infinity.
- If variables in $\mathbf{X}$ are highly correlated, $\mathbf{X}^T \mathbf{X}$ is close to being singular (i.e., LS are very unstable). This problem is solved by ridge regression. In fact, this was its original motivation.
- It can also be thought as a way of reducing the variance of $\hat{\boldsymbol{\beta}}_R$

Example of Ridge Regression

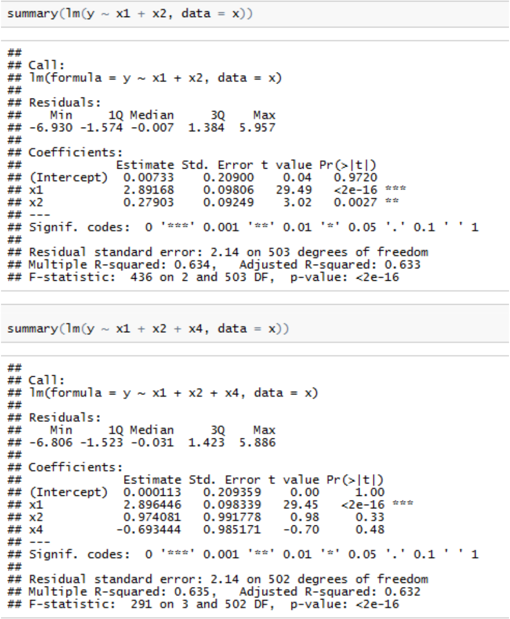


```
library(MASS)

set.seed(123)
x1 <- rnorm(506)
x2 <- rnorm(506, mean = 2, sd = 1)


x3 <- rexp(506, rate = 1)
x4 <- x2 + rnorm(506, sd = 0.1)
x5 <- x1 + rnorm(506, sd = 0.1)
x6 <- x1 - x2 + rnorm(506, sd = 0.1)
x7 <- x1 + x3 + rnorm(506, sd = 0.1)

# Let's make x1 and x2 important covariates

y <- x1 * 3 + x2/3 + rnorm(506, sd = 2.2)

x <- data.frame(y = y, x1 = x1, x2 = x2, x3 = x3, x4 = x4, x5 = x5, x6 = x6,
    x7 = x7)

summary(lm(y ~ ., data = x))
```

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   0.0346     0.2302    0.15    0.881
x1            3.2261     1.6809    1.92    0.056 .
x2            0.2387     1.3936    0.17    0.864
x3           -0.3593     0.9868   -0.36    0.716
x4           -0.6936     0.9902   -0.70    0.484
x5            0.0927     0.9116    0.10    0.919
x6           -0.7389     1.0111   -0.73    0.465
x7            0.3165     0.9861    0.32    0.748
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.15 on 498 degrees of freedom
Multiple R-squared: 0.635,    Adjusted R-squared: 0.63
F-statistic:  124 on 7 and 498 DF,  p-value: <2e-16
```

Nothing is significant!

)



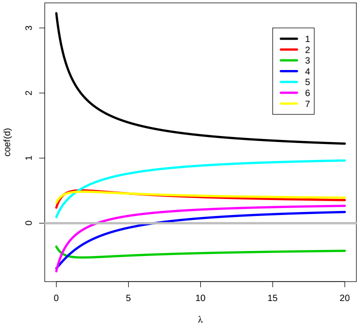
```r
summary(lm(y ~ x1 + x2, data = x))
```

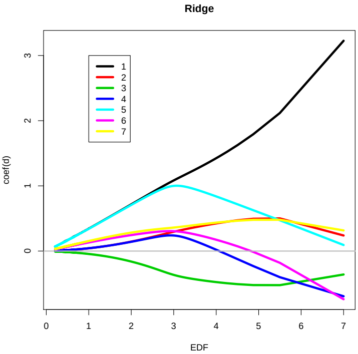
```
##
## Call:
## lm(formula = y ~ x1 + x2, data = x)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -6.930 -1.574 -0.007  1.384  5.957
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.00733    0.20900    0.04   0.9720
## x1           2.89168    0.09806   29.49   <2e-16 ***
## x2           0.27903    0.09249    3.02   0.0027 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.14 on 503 degrees of freedom
## Multiple R-squared: 0.634,   Adjusted R-squared: 0.633
## F-statistic:  436 on 2 and 503 DF,  p-value: <2e-16
```

```r
summary(lm(y ~ x1 + x2 + x4, data = x))
```

```
##
## Call:
## lm(formula = y ~ x1 + x2 + x4, data = x)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -6.806 -1.523 -0.031  1.423  5.886
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.000113   0.209359    0.00     1.00
## x1           2.896446   0.098339   29.45   <2e-16 ***
## x2           0.974081   0.991778    0.98     0.33
## x4          -0.693444   0.985171   -0.70     0.48
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.14 on 502 degrees of freedom
## Multiple R-squared: 0.635,   Adjusted R-squared: 0.632
## F-statistic:  291 on 3 and 502 DF,  p-value: <2e-16
```

Ridge Coefficients



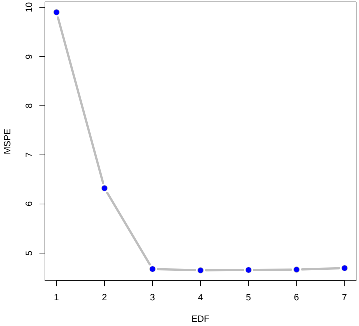


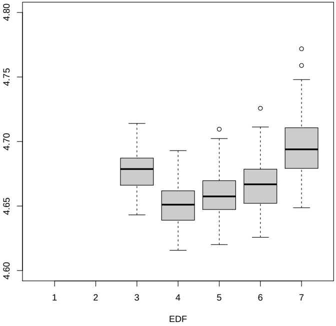
- The effective degrees of freedom EDF$(\lambda) = \sum_{j=1}^{p} \frac{d_j^2}{d_j^2+\lambda}$, where d_j are the single values of $\mathbf{X}$.
- What is a good amount of shrinkage? (i.e., choosing λ)
- We might want to look at their predictive power!

Choosing λ

A standard practice is to use cross-validation to select a good λ.

I will use 100 10-fold CV estimates of the mean squared prediction error (MSPE).





Choosing λ (cont.)

```
        lambda    x1    x2     x3     x4    x5     x6    x7
    1 2809.521 0.344 0.043 -0.045  0.043 0.342  0.132 0.154
    2  662.711 0.716 0.145 -0.161  0.142 0.707  0.245 0.283
    3   66.671 1.084 0.296 -0.366  0.239 1.000  0.305 0.360
->  4    8.001 1.406 0.419 -0.472  0.034 0.850  0.183 0.432
    5    2.667 1.792 0.495 -0.522 -0.230 0.627 -0.012 0.479
    6    1.333 2.118 0.501 -0.523 -0.401 0.472 -0.180 0.480
    7    0.000 3.226 0.239 -0.359 -0.694 0.093 -0.739 0.317
```

LASSO: least absolute shrinkage and selection operator

The lasso coefficients minimize a different *penalized* RSS:

$$(\hat{\alpha}, \hat{\boldsymbol{\beta}})_L = \min_{\alpha, \boldsymbol{\beta} \in \mathbb{R}^p} \sum_{i=1}^{n} \left(y_i - \alpha - \boldsymbol{\beta}^T \mathbf{x}_i \right)^2$$

subject to

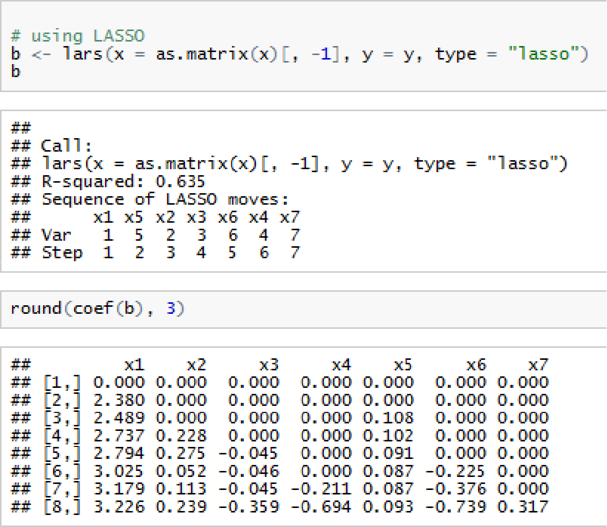
$$\sum_{j=1}^{p} |\boldsymbol{\beta}_j| \leq C \text{ for some } C > 0.$$

or equivalently:

$$(\hat{\alpha}, \hat{\boldsymbol{\beta}})_L = \min_{\alpha, \boldsymbol{\beta} \in \mathbb{R}^p} \left\{ \sum_{i=1}^{n} \left(y_i - \alpha - \boldsymbol{\beta}^T \mathbf{x}_i \right)^2 + \lambda \sum_{j=1}^{p} |\boldsymbol{\beta}_j| \right\}$$

- Again, we have a tuning parameter λ that controls the amount of regularization.

LASSO: Example



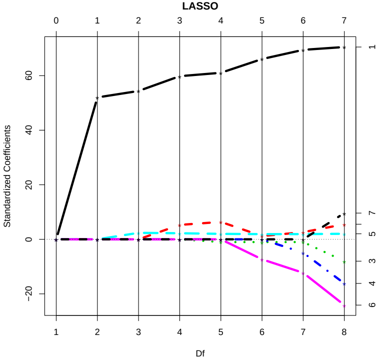
```
# using LASSO
b <- lars(x = as.matrix(x)[, -1], y = y, type = "lasso")
b
```

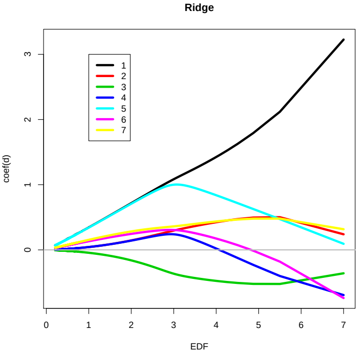
```
##
## Call:
## lars(x = as.matrix(x)[, -1], y = y, type = "lasso")
## R-squared: 0.635
## Sequence of LASSO moves:
##      x1 x5 x2 x3 x6 x4 x7
## Var   1  5  2  3  6  4  7
## Step  1  2  3  4  5  6  7
```

```
round(coef(b), 3)
```

```
##         x1    x2     x3     x4    x5     x6    x7
## [1,] 0.000 0.000  0.000  0.000 0.000  0.000 0.000
## [2,] 2.380 0.000  0.000  0.000 0.000  0.000 0.000
## [3,] 2.489 0.000  0.000  0.000 0.108  0.000 0.000
## [4,] 2.737 0.228  0.000  0.000 0.102  0.000 0.000
## [5,] 2.794 0.275 -0.045  0.000 0.091  0.000 0.000
## [6,] 3.025 0.052 -0.046  0.000 0.087 -0.225 0.000
## [7,] 3.179 0.113 -0.045 -0.211 0.087 -0.376 0.000
## [8,] 3.226 0.239 -0.359 -0.694 0.093 -0.739 0.317
```

LASSO: Example





- The y-axis of the LASSO plot shows each coefficient scaled by the size of the corresponding covariate ("Standardized Coefficients").
- Large enough λ (or small enough C) will set some of the LASSO coefficients exactly equal to 0. (i.e., model selection)!

Ridge *vs.* LASSO

Elements of Statistical Learning (2nd Ed.) ©Hastie, Tibshirani & Friedman 2009 Chap 3

$$\text{RSS} = (\mathbf{y} - \boldsymbol{\beta}^T\mathbf{X})^T(\mathbf{y} - \boldsymbol{\beta}^T\mathbf{X}) = (\boldsymbol{\beta} - \hat{\boldsymbol{\beta}})^T\mathbf{X}^T\mathbf{X}(\boldsymbol{\beta} - \hat{\boldsymbol{\beta}})$$

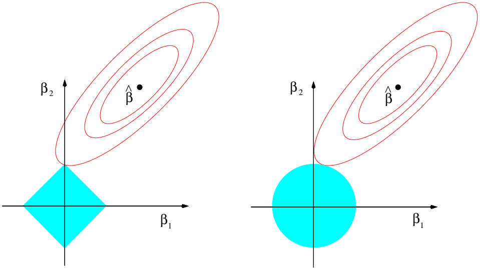


FIGURE 3.11. *Estimation picture for the lasso (left) and ridge regression (right). Shown are contours of the error and constraint functions. The solid blue areas are the constraint regions $|\beta_1| + |\beta_2| \leq t$ and $\beta_1^2 + \beta_2^2 \leq t^2$, respectively, while the red ellipses are the contours of the least squares error function.*

Limitations of LASSO

- If $p > n$, LASSO can select at most n variables out of p candidates (Efron et al., 2004)
- If there is a group of highly correlated variables, LASSO tends to select only one covariate from the group, and in general its prediction performance is dominated by ridge regression.
- As these situations are common in -omics studies, LASSO does not seem to be the most convenient method.

Elastic Net

The Elastic Net coefficients minimize a different *penalized* RSS:

$$(\hat{\alpha}, \hat{\boldsymbol{\beta}})_{EN} = \min_{\alpha, \boldsymbol{\beta} \in \mathbb{R}^p} \sum_{i=1}^{n} \left(y_i - \alpha - \boldsymbol{\beta}^T \mathbf{x}_i \right)^2$$

subject to

$$\sum_{j=1}^{p} (1-\alpha)|\boldsymbol{\beta}_j| + \alpha(\boldsymbol{\beta}_j)^2 \leq C \text{ for some } C > 0.$$

or equivalently:

$$(\hat{\alpha}, \hat{\boldsymbol{\beta}})_{EN} = \min_{\alpha, \boldsymbol{\beta} \in \mathbb{R}^p} \left\{ \sum_{i=1}^{n} \left(y_i - \alpha - \boldsymbol{\beta}^T \mathbf{x}_i \right)^2 + \lambda_1 \sum_{j=1}^{p} |\boldsymbol{\beta}_j| + \lambda_2 \sum_{j=1}^{p} (\boldsymbol{\beta}_j)^2 \right\}$$

A convex combination of the lasso and ridge penalties!
EN has the ”grouping effect” property (i.e., absolute values of coefficients of highly correlated variables tend to be equal).

Elastic Net Penalty

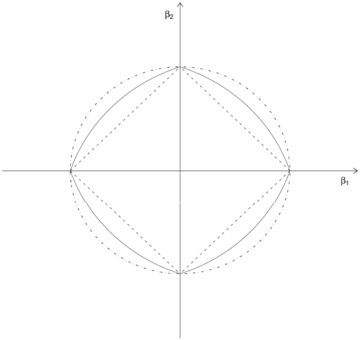


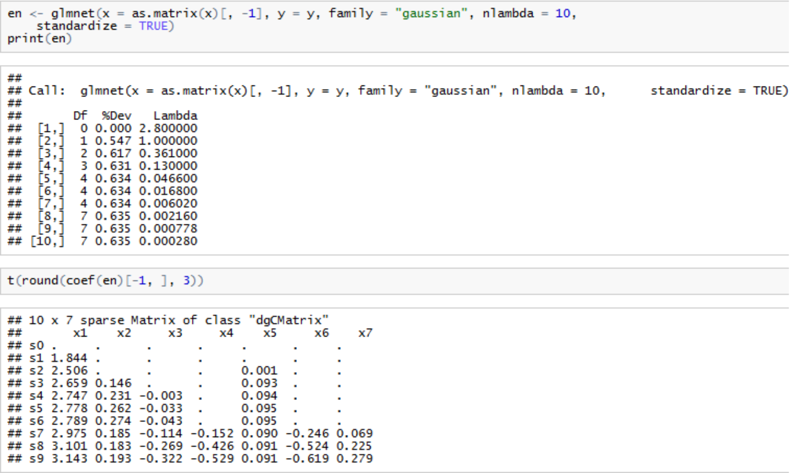
Fig. 1. Two-dimensional contour plots (level 1) (· - · - · -, shape of the ridge penalty; -------, contour of the lasso penalty; ———, contour of the elastic net penalty with $\alpha = 0.5$): we see that singularities at the vertices and the edges are strictly convex; the strength of convexity varies with α

Image: The Elements of Statistical Learning, Hastie, Tibshirani, Friedman

Elastic Net Algorithms

- Two famous algorithms can be used to compute EN coefficient with respective packages in R:
  - lars-en proposed in Zou and Hastie (2005)
  - glmnet in Friedman et al., (2010)
- In general, the latter is computationally more efficient and demonstrated better prediction performance (Friedman et al., 2010).
- glmnet can also be used to obtain LASSO estimates, and for classification (family=”binomial”)!.

Elastic Net: Example



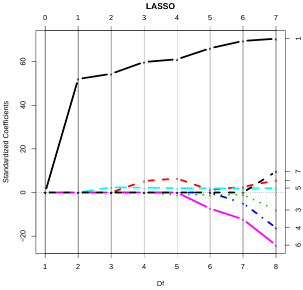
```
en <- glmnet(x = as.matrix(x)[, -1], y = y, family = "gaussian", nlambda = 10,
    standardize = TRUE)
print(en)
```

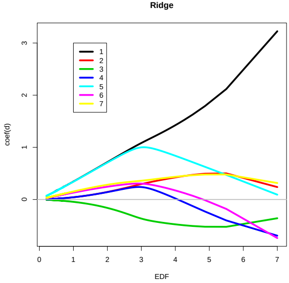
```
##
## Call:  glmnet(x = as.matrix(x)[, -1], y = y, family = "gaussian", nlambda = 10,      standardize = TRUE)
##
##       Df  %Dev   Lambda
##  [1,]  0 0.000 2.800000
##  [2,]  1 0.547 1.000000
##  [3,]  2 0.617 0.361000
##  [4,]  3 0.631 0.130000
##  [5,]  4 0.634 0.046600
##  [6,]  4 0.634 0.016800
##  [7,]  4 0.634 0.006020
##  [8,]  7 0.635 0.002160
##  [9,]  7 0.635 0.000778
## [10,]  7 0.635 0.000280
```

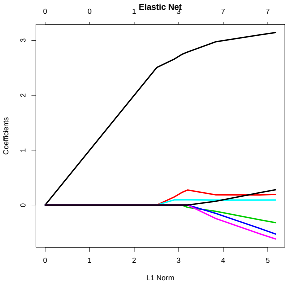
```
t(round(coef(en)[-1, ], 3))
```

```
## 10 x 7 sparse Matrix of class "dgCMatrix"
##       x1    x2     x3     x4    x5     x6    x7
## s0 .     .      .      .      .     .      .
## s1 1.844 .      .      .      .     .      .
## s2 2.506 .      .      .      0.001 .      .
## s3 2.659 0.146  .      .      0.093 .      .
## s4 2.747 0.231 -0.003  .      0.094 .      .
## s5 2.778 0.262 -0.033  .      0.095 .      .
## s6 2.789 0.274 -0.043  .      0.095 .      .
## s7 2.975 0.185 -0.114 -0.152 0.090 -0.246 0.069
## s8 3.101 0.183 -0.269 -0.426 0.091 -0.524 0.225
## s9 3.143 0.193 -0.322 -0.529 0.091 -0.619 0.279
```

Elastic Net: Example (cont.)







Choosing λ

- The function `cv.glmnet` runs glmnet to get the λ values for which an additional coefficient is added to the model.
- Given this sequence, cv.glmnet does an n-fold cross-validation (default n = 10) and estimates the prediction error.
- The average error and standard deviation over the folds can be plotted to choose the optimal λ. Note that cv.glmnet does NOT search for values for alpha.
- lambda.min gives the λ that minimizes the error. lambda.1se is the largest value of lambda with an error within 1 standard error from the minimum (i.e., a less complex model at a low cost).

Conclusions

- There are many methods and algorithms to perform classification and regression (more than those covered here).
- Which one is better? We can evaluate different options using cross-validation.
- There is a trade-off between bias and variance when we increase the number of folds in a CV.
- There is a trade-off between bias and variance when we fit more complex models.
- Even good CV performance does not mean that you will get good performance in the test set.
- Some of the methods we covered are very sensitive to unbalance classes (e.g., SVM).

References

- Hoerl, A.E. and Kennard, R. (1970). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, **12**, 55-67.

- Tibshirani, R. (1996) Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B*, 267-288.

- Hastie, T., Tibshirani, R., and Friedman, J. (2001). The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer Series in Statistics.

- Efron, B., Hastie, T., Johnstone, I., and Tibshirani, R. (2004). Least angle regression. *Annals of Statistics*, **32**, 406-499.

- Zou, H. and Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society, Series B*, **67**, 301-320.

- Friedman, J.; Hastie, T., and Tibshirani, R. (2010) Regularization paths for generalized linear models via coordinate descent. *Journal of statistical software*, **33**,1.