Regularization Methods

Business Analytics Practice
Winter Term 2015/16
Stefan Feuerriegel

Today's Lecture

Objectives

1. Avoiding overfitting and improving model interpretability with the help of regularization methods
2. Understanding both ridge regression and the LASSO
3. Applying these methods for variable selection

Outline

Outline

Motivation for Regularization

- Linear models are frequently favorable due to their interpretability and often good predictive performance
- Yet, Ordinary Least Squares (OLS) estimation faces challenges

Challenges

1. Interpretability
   - OLS cannot distinguish variables with little or no influence
   - These variables distract from the relevant regressors
2. Overfitting
   - OLS works well when number of observation n is bigger than the number of predictors p, i. e. $n \gg p$
   - If $n \approx p$, overfitting results into low accuracy on unseen observations
   - If $n < p$, variance of estimates is infinite and OLS fails
   - As a remedy, one can identify only relevant variables by feature selection

Motivation for Regularization

Fitting techniques as alternatives to OLS

- **Subset selection**
  - Pick only a subset of all p variables which is assumed to be relevant
  - Estimate model with least squares using these reduced set of variables
- **Dimension reduction**
  - Project p predictors into a d-dimensional subspace with $d < p$
  - These d features are used to fit a linear model by least squares
- **Shrinkage methods**, also named regularization
  - Fit model with all p variables
  - However, some coefficients are shrunken towards zero
  - Has the effect of reducing variance

Regularization Methods

- Fit linear models with least squares but impose constraints on the coefficients
- Instead, alternative formulations add a penalty in the OLS formula
- Best known are ridge regression and LASSO (least absolute shrinkage operator)
  - Ridge regression can shrink parameters close to zero
  - LASSO models can shrink some parameters exactly to zero
    $\rightarrow$ Performs implicit variable selection

Outline

Ridge Regression

OLS estimation

- Recall the OLS technique to estimate $\boldsymbol{\beta} = [\beta_0, \beta_1, \ldots, \beta_p]^T$
- Minimizes the residual sum of squares (RSS)

$$\boldsymbol{\beta}_{\text{OLS}} = \min_{\boldsymbol{\beta}} RSS = \min_{\boldsymbol{\beta}} \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2$$

Ridge regression

- Imposes a penalty on the size of the coefficients to reduce the variance of the estimates

$$\boldsymbol{\beta}_{\text{ridge}} = \min_{\boldsymbol{\beta}} \underbrace{\sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2}_{\text{RSS}} + \underbrace{\lambda \sum_{j=1}^{p} \beta_j^2}_{\text{shrinkage penalty}}$$

Tuning Parameter

- Tuning parameter $\lambda > 0$ controls the relative impact of the penalty
- Penalty $\lambda \sum_{j=1}^{p} \beta_j^2$ has the effect of shrinking β_j towards zero
- If $\lambda \approx 0$, penalty term has no effect (similar to OLS)
- Choice of λ is critical $\rightarrow$ determined separately via cross validation

Ridge Regression in R

- Predicting salaries of U. S. baseball players based on game statistics
- Loading data `Hitters`

```r
library(ISLR) # Hitters is located inside ISLR
data(Hitters)
Hitters <- na.omit(Hitters) # salary can be missing
```

- Loading package `glmnet` which implements ridge regression

```r
library(glmnet)
```

- Main function `glmnet(x, y, alpha=0)` requires dependent variable `y` and regressors `x`
- Function only processes numerical input, whereas categorical variables needs to be transformed via `model.matrix(...)`

Ridge Regression in R

- Prepare variables

```
set.seed(0)

# drop 1st column with intercept (glmnet has already one)
x <- model.matrix(Salary ~ ., Hitters)[, -1]
y <- Hitters$Salary
train_idx <- sample(nrow(x), size=0.9*nrow(x))

x.train <- x[train_idx, ]
x.test <- x[-train_idx, ]
y.train <- y[train_idx]
y.test <- y[-train_idx]
```

- Call ridge regression and automatically test a sequence of λ

```
lm.ridge <- glmnet(x.train, y.train, alpha=0)
```

Ridge Regression in R

- `coef(...)` retrieves coefficients belonging to each λ

```
dim(coef(lm.ridge))

## [1]  20 100
```

$\rightarrow$ here: 100 models with different λ and each with 20 coefficients

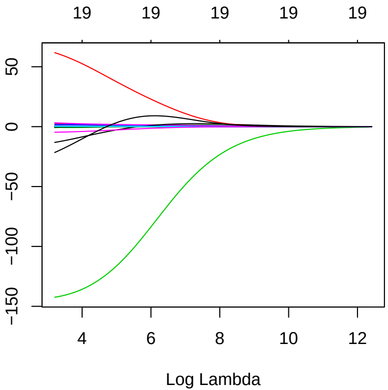
- For example, the 50th model is as follows

```
lm.ridge$lambda[50]        # tested lambda value

## [1] 2581.857

head(coef(lm.ridge)[,50]) # estimated coefficients

##   (Intercept)       AtBat        Hits       HmRun        Runs
## 211.76123020  0.08903326  0.37913073  1.21041548  0.64115228
##          RBI
##   0.59834311
```

Ridge Regression in R

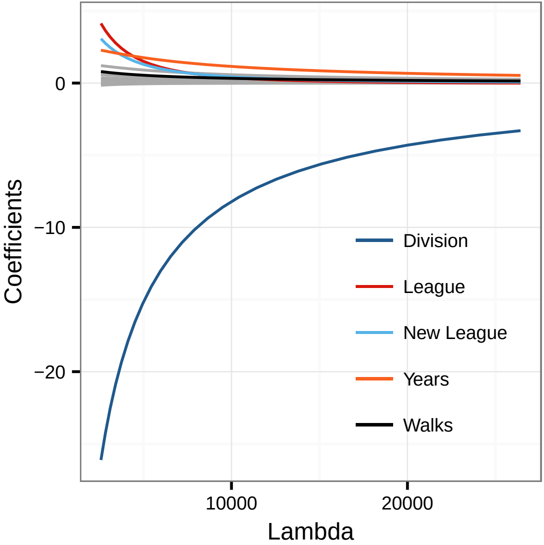
- `plot(model, xvar="lambda")` investigates the influence of λ on the estimated coefficients for all variables

```r
plot(lm.ridge, xvar="lambda")
```



- Bottom axis gives $\ln \lambda$, top the number of non-zero coefficients

Plot: Lambda vs. Coefficients



- Manual effort necessary for pretty format
- As λ increases, coefficients shrink towards zero
- All other variables are shown in gray

Parameter Tuning

- Optimal λ is determined via cross validation by minimizing the mean squared error from a prediction
- Usage is `cv.glmnet(x, y, alpha=0)`

```
cv.ridge <- cv.glmnet(x.train, y.train, alpha=0)
```

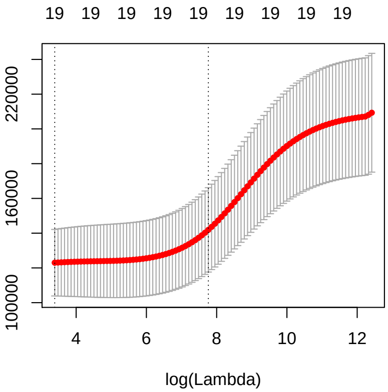
- Optimal λ and corresponding coefficients

```
cv.ridge$lambda.min

## [1] 29.68508

head(coef(cv.ridge, s="lambda.min"))

## 6 x 1 sparse Matrix of class "dgCMatrix"
##                         1
## (Intercept) 109.4192279
## AtBat        -0.6764771
## Hits          2.5974777
## HmRun        -0.7058689
## Runs          1.8565943
## RBI           0.3434801
```

Parameter Tuning

- `plot(cv.model)` compares the means squared error across λ

```
plot(cv.ridge)
```



- Mean squared error first remains fairly constant and then rises sharply

Ridge Regression in R

- `predict(model, newx=x, s=lambda)` makes predictions for new data `x` and a specific λ

```
pred.ridge <- predict(cv.ridge, newx=x.test, s="lambda.min")
head(cbind(pred.ridge, y.test))
```

```
##                          1   y.test
## -Alan Ashby       390.1766  475.000
## -Andre Dawson    1094.5741  500.000
## -Andre Thornton   798.5886 1100.000
## -Alan Trammell    893.8298  517.143
## -Barry Bonds      518.9105  100.000
## -Bob Dernier      353.4100  708.333
```

- Mean absolute percentage error (MAPE)

```
mean(abs((y.test - pred.ridge)/y.test))
```

```
## [1] 0.6811053
```

Scaling of Estimates

OLS estimation

- Recall: least square estimates are scale equivalent
- Multiplying $\boldsymbol{x}_j$ by $c \Rightarrow$ scaling of β_j by a factor $1/c$

Ridge regression

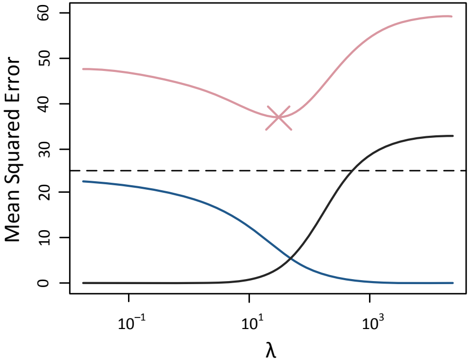
- In contrast, coefficients in ridge regression can change substantially when scaling variable $\boldsymbol{x}_j$ due to penalty term
- Best is to use the following approach
  1. Scale variables via

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

     which divides by the standard deviation of x_j
  2. Estimate the coefficients of ridge regression
- `glmnet` scales accordingly, but returns coefficients on original scale

Bias-Variance Trade-Off

- Ridge regressions benefits from bias-variance trade-off
- As λ increases, the flexibility of ridge regression coefficients decreases
  $\rightarrow$ This Decreases variance **but** increases bias



- Squared bias (in black), variance (blue), and error on test set (red)
- Dashed line is minimum possible mean squared error

Pros and Cons

Advantages

- Ridge regression can reduce the variance (with an increasing bias) $\rightarrow$ works best in situations where the OLS estimates have high variance
- Can improve predictive performance
- Works in situations where $p < n$
- Mathematically simple computations

Disadvantages

- Ridge regression is not able to shrink coefficients to exactly zero
- As a result, it cannot perform variable selection

$\Rightarrow$ Alternative: Least Absolute Shrinkage and Selection Operator (LASSO)

Outline

LASSO

Least Absolute Shrinkage and Selection Operator (LASSO)

- Ridge regression always includes p variables, but LASSO performs variables selection
- LASSO only changes the shrinkage penalty

$$\boldsymbol{\beta}_{\text{LASSO}} = \min_{\boldsymbol{\beta}} \underbrace{\sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2}_{\text{RSS}} + \underbrace{\lambda \sum_{j=1}^{p} |\beta|_j}_{\text{shrinkage penalty}}$$

- Here, the LASSO uses the L_1-norm $\|\boldsymbol{\beta}\|_1 = \sum_j |\beta_j|$
- This penalty allows coefficients to shrink towards exactly zero
- LASSO usually results into sparse models, that are easier to interpret

LASSO in R

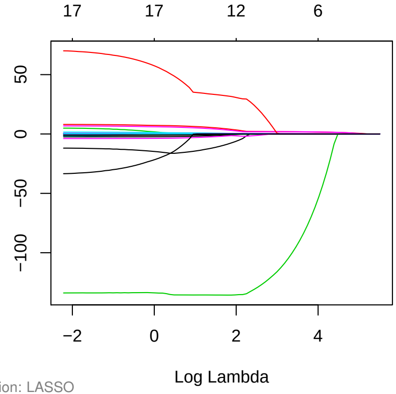
- Implemented in `glmnet(x, y, alpha=1)` as part of the `glmnet` package

```
lm.lasso <- glmnet(x.train, y.train, alpha=1)
```

Note: different value for `alpha`

- `plot(...)` shows how λ changes the estimated coefficients

```
plot(lm.lasso, xvar="lambda")
```



Parameter Tuning

- `cv.glmnet(x, y, alpha=1)` determines optimal λ via cross validation by minimizing the mean squared error from a prediction

```
set.seed(0)
cv.lasso <- cv.glmnet(x.train, y.train, alpha=1)
```

- Optimal λ and corresponding coefficients ("." are removed variables)

```
cv.lasso$lambda.min

## [1] 2.143503

head(coef(cv.lasso, s="lambda.min"))

## 6 x 1 sparse Matrix of class "dgCMatrix"
##                       1
## (Intercept) 189.7212235
## AtBat        -1.9921887
## Hits          6.6124279
## HmRun         0.6674432
## Runs          .
## RBI           .
```

Parameter Tuning

- Total variables

```r
nrow(coef(cv.lasso))
```

```
## [1] 20
```

- Omitted variables

```r
dimnames(coef(cv.lasso, s="lambda.min"))[[1]][which(
  coef(cv.lasso, s="lambda.min") == 0)]
```

```
## [1] "Runs"    "RBI"     "CAtBat" "CHits"
```

- Included variables

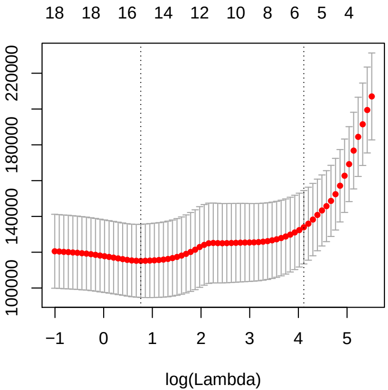
```r
dimnames(coef(cv.lasso, s="lambda.min"))[[1]][which(
  coef(cv.lasso, s="lambda.min") != 0)]
```

```
##  [1] "(Intercept)" "AtBat"       "Hits"        "HmRun"
##  [6] "Years"       "CHmRun"      "CRuns"       "CRBI"
## [11] "LeagueN"     "DivisionW"   "PutOuts"     "Assists"
## [16] "NewLeagueN"
```

Parameter Tuning

- `plot(cv.model)` compares the means squared error across λ

```
plot(cv.lasso)
```



- Mean squared error first remains fairly constant and then rises sharply
- Top axis denotes the number of included model variables

LASSO in R

- `predict(model, newx=x, s=lambda)` makes predictions for new data `x` and a specific λ

```r
pred.lasso <- predict(cv.lasso, newx=x.test, s="lambda.min")
```

- Mean absolute percentage error (MAPE) of LASSO

```r
mean(abs((y.test - pred.lasso)/y.test))

## [1] 0.6328225
```

- For comparison, error of ridge regression

```
## [1] 0.6811053
```

Outline

Problem Formulation

Both ridge regression and LASSO can be rewritten as

$$\boldsymbol{\beta}_{\text{ridge}} = \min_{\boldsymbol{\beta}} \underbrace{\sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2}_{\text{RSS}} \qquad \text{s.t.} \qquad \sum_{j=1}^{p} \beta_j^2 \leq \theta$$

$$\boldsymbol{\beta}_{\text{LASSO}} = \min_{\boldsymbol{\beta}} \underbrace{\sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2}_{\text{RSS}} \qquad \text{s.t.} \qquad \sum_{j=1}^{p} |\beta_j| \leq \theta$$

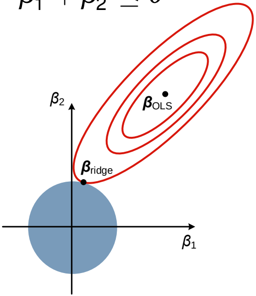
Outlook: both ridge regression and LASSO have Bayesian formulations

Variable Selection with LASSO

Comparison of previous constraints

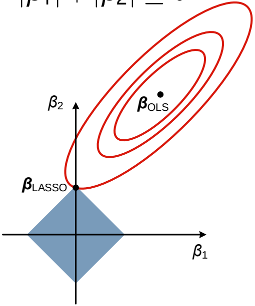
Ridge regression

$$\beta_1^2 + \beta_2^2 \leq \theta$$



LASSO

$$|\beta_1| + |\beta_2| \leq \theta$$



- Objective function *RSS* as contours in red
- Constraints (blue) in 2 dimensions
- Intersection occurs at $\beta_1 = 0$ for LASSO

Case Study

Example

- Comparison to OLS estimator

```r
lm.ols <- lm(y.train ~ x.train)

# Workaround as predict.lm only accepts a data.frame
pred.ols <-  predict(lm.ols, data.frame(x.train=I(x.test)))

mean(abs((y.test - pred.ols)/y.test))

## [1] 0.6352089
```

- Comparison

OLS	Ridge regression	LASSO
0.64	0.68	0.63

- Here: LASSO can outperform OLS with fewer predictors

Elastic Net

- Elastic net generalizes the ideas of both LASSO and ridge regression
- Combination of both penalties

$$\boldsymbol{\beta}_{\text{ElasticNet}} = \min_{\boldsymbol{\beta}} RSS + \lambda \left[\underbrace{(1-\alpha) \|\boldsymbol{\beta}\|_2^2 / 2}_{L_2\text{-penalty}} + \underbrace{\alpha \|\boldsymbol{\beta}\|_1}_{L_1\text{-penalty}} \right]$$

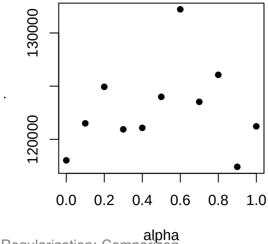
- L_1-penalty helps generating a sparse model
- L_2-part overcomes a strict selection
- Parameter α controls numerical stability
- $\alpha = 0.5$ tends to handle correlated variables as groups

Elastic Net in R

Example

- Test the elastic net with a sequence of values for α
- Report in-sample mean squared error

```
set.seed(0)
alpha <- seq(from=0, to=1, by=0.1)
en <- lapply(alpha, function(a)
        cv.glmnet(x.train, y.train, alpha=a))
en.mse <- unlist(lapply(en, function(i)
            i$cvm[which(i$lambda==i$lambda.min)]))
plot(alpha, en.mse, ylab="Mean Squared Error", pch=16)
```

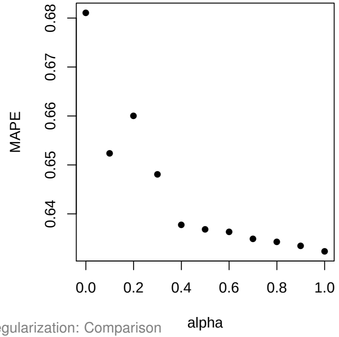


Elastic Net in R

Example (continued)

- Report out-of-sample mean absolute prediction error

```
en.mape <- unlist(lapply(en, function(i) {
             pred <- predict(i, newx=x.test,
                                   s="lambda.min")
             mean(abs((y.test - pred)/y.test))
           }))
plot(alpha, en.mape, ylab="MAPE", pch=16)
```



Outline

Summary

Regularization methods

- Regularization methods bring advantages beyond OLS
- Cross validation chooses tuning parameter λ
- LASSO performs variable selection
- Neither ridge regression nor LASSO dominates one another
- Cross validation finds the best approach for a given dataset

Outlook

- In practice, λ is scaled by rule of thumb to get better results
- Research has lately developed several variants and improvements
- Spike-and-slab regression can be a viable alternative for inferences

Further Readings

Package glmnet

- glmnet tutorial: `http://web.stanford.edu/~hastie/glmnet/glmnet_alpha.html`
- glmnet webinar: `http://web.stanford.edu/~hastie/TALKS/glmnet_webinar.pdf`
  $\rightarrow$ see Hastie's website for data and scripts

Background on methods

- Talk on elastic net: `http://web.stanford.edu/~hastie/TALKS/enet_talk.pdf`
- Section 6.2 in the book "An Introduction to Statistical Learning"

Applications

- Especially healthcare analytics, but also sports

  $\rightarrow$ e. g. Groll, Schauberger & Tutz (2015): Prediction of major international soccer tournaments based on team-specific regularized Poisson regression: An application to the FIFA World Cup 2014. In: Journal of Quantitiative Analysis in Sports, 10:2, pp. 97–115.