Ridge Regression, LASSO and Elastic Net

A talk given at NYC open data meetup, find more at www.nycopendata.com

Yunting Sun
Google Inc

Overview

- Linear Regression
- Ordinary Least Square
- Ridge Regression
- LASSO
- Elastic Net
- Examples
- Exercises

Note: make sure that you have installed elasticnet package

```
library(MASS)
library(elasticnet)
```

Linear Regression

n observations, each has one response variable and p predictors

$$Y = (y_1, \cdots, y_n)^T, \qquad n \times 1$$

$$X = (X_1, \cdots, X_p), \qquad n \times p$$

- We want to find a linear combination β of predictors $x = (x_1, \cdots, x_p)$ to
  - describe the actual relationship between y and $x_1, \cdots, x_p$
  - use $\hat{y} = x^T\beta$ to predict y
- Examples
  - find relationship between pressure and water boiling point
  - use GDP to predict interest rate (the accuracy of the prediction is important but the actual relationship may not matter)

Quality of an estimator $\hat{\beta}$

Suppose β_0 is the true value and $y = x^T\beta_0 + \sigma\epsilon, \epsilon \sim \mathcal{N}(0,1)$

- Prediction error at x_0, the difference between the actual response and the model prediction

$$\text{EPE}(x_0) = E[(y - x_0^T\hat{\beta})^2 | x = x_0]$$

$$\text{EPE}(x_0) = \sigma^2 + E(x_0^T\beta_0 - x_0^T\hat{\beta})^2$$

$$\text{EPE}(x_0) = \sigma^2 + [\text{Bias}^2(x_0^T\hat{\beta}) + \text{Var}(x_0^T\hat{\beta})]$$

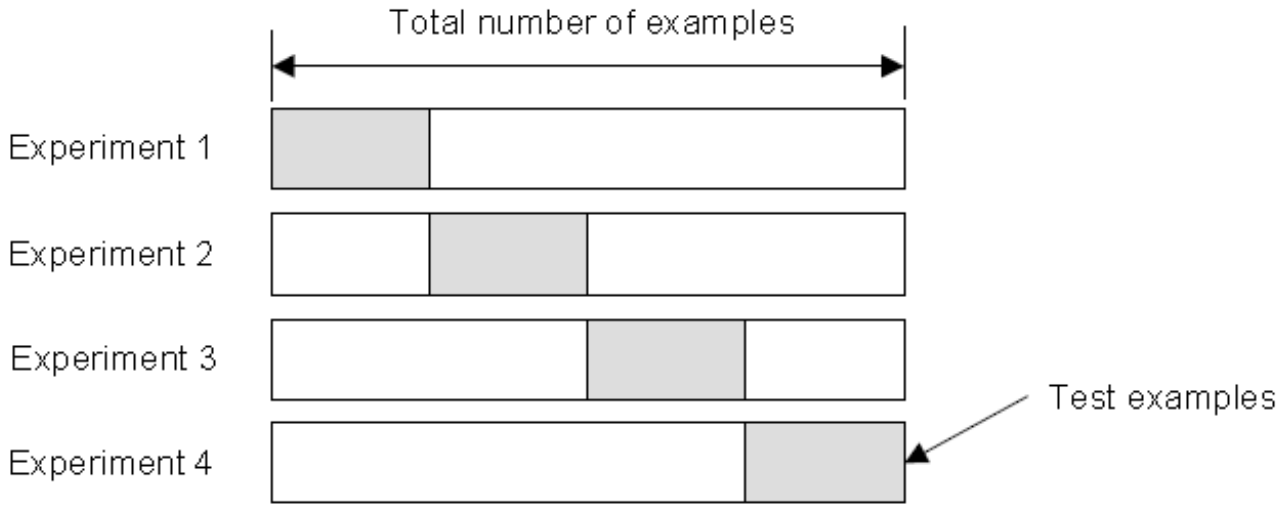
  Where $\text{Bias}(x_0^T\hat{\beta}) = E(x_0^T\beta_0 - x_0^T\hat{\beta})$.

- The second and third terms make up the mean squared error of $x_0^T\hat{\beta}$ in estimating $x_0^T\beta_0$.
- How to estimate prediction error?

K-fold Cross Validation

- Split dataset into K groups
  - leave one group out as test set
  - use the rest K-1 groups as training set to train the model
  - estimate prediction error of the model from the test set

K-fold Cross Validation



Let E_i be the prediction errors for the ith test group, the average prediction error is

$$E = \frac{1}{K}\sum_{i=1}^{K} E_i$$

Quality of an estimator $\hat{\beta}$

- Mean squared error of the estimator

$$\mathrm{MSE}(\hat{\beta}) = E[(\hat{\beta} - \beta_0)^2]$$

$$\mathrm{MSE}(\hat{\beta}) = \mathrm{Bias}^2(\hat{\beta}) + \mathrm{Var}(\hat{\beta})$$

- A biased estimator may achieve smaller MSE than an unbiased estimator
- useful when our goal is to understand the relationship instead of prediction

Least Squares Estimator (LSE)

$$Y_{n\times 1} = X_{n\times p}\beta_{p\times 1} + \sigma\epsilon_{n\times 1}$$

$$y_i = \sum_{j=1}^{p} x_{ij}\beta_j + \sigma\epsilon_i, i = 1, \cdots, n$$

$$\epsilon_i \sim^{\text{i.i.d}} \mathcal{N}(0, 1)$$

Minimize Residual Sum of Square (RSS)

$$\hat{\beta} = \arg\min_{\beta}(Y - X\beta)^T(Y - X\beta) = (X^TX)^{-1}X^TY$$

The solution is uniquely well defined when $n > p$ and X^TX inversible

Pros

$$\mathrm{E}(\hat{\beta}) = \beta \quad \text{unbiased}$$

- LSE has the minimum MSE among unbiased linear estimator though a biased estimator may have smaller MSE than LSE
- explicit form
- computation $O(np^2)$
- confidence interval, significance of coefficient

Cons

$$\mathrm{Var}(\hat{\beta}) = (X^T X)^{-1}\sigma^2$$

- Multicollinearity leads to high variance of estimator
  - exact or approximate linear relationship among predictors
  - $(X^T X)^{-1}$ tends to have large entries
- Requires n > p, i.e., number of observations larger than the number of predictors

$$E_{x_0}\mathrm{EPE}(x_0) =\sim \sigma^2(p/n) + \sigma^2 \qquad \text{estimated prediction error}$$

- Prediction error increases linearly as a function of p
- Hard to interpret when the number of predictors is large, need a smaller subset that exhibit the strongest effects

Example: Leukemia classification

- Leukemia Data, Golub et al. Science 1999
- There are 38 training samples and 34 test samples with total $p = 7129$ genes (p >> n)
- X_{ij} is the gene expression value for sample i and gene j
- Sample i either has tumor type AML or ALL
- We want to select genes relevant to tumor type
  - eliminate the trivial genes
  - grouped selection as many genes are highly correlated
- LSE does not work here!

Solution: regularization

- instead of minimizing RSS,

$$\text{minimize} \quad (\text{RSS} + \lambda \times \text{penalty on the parameters})$$

- Trade bias for smaller variance, biased estimator when $\lambda ! = 0$
- Continuous variable selection (unlike AIC, BIC, subset selection)
- λ can be chosen by cross validation

Ridge Regression

$$\hat{\beta}^{\text{ridge}} = \arg\min_{\beta} \{ \|Y - X\beta\|_2^2 + \lambda \|\beta\|_2^2 \}$$

$$\hat{\beta}^{\text{ridge}} = (X^T X + \lambda I)^{-1} X^T Y$$

Pros:

- p >> n
- Multicollinearity
- biased but smaller variance and smaller MSE (Mean Squared Error)
- explicit solution

Cons:

- shrink coefficients to zero but can not produce a parsimonious model

Grouped Selection

- if two predictors are highly correlated among themselves, the estimated coefficients will be similar for them.
- if some variables are exactly identical, they will have same coefficients

Ridge is good for grouped selection but not good for eliminating trivial genes

Example: Ridge Regression (Collinearity)

- multicollinearity $x_3 = x_1 + x_2$
- show that ridge regression beats OLS in the multilinearity case

```
library(MASS)
n = 500
z = rnorm(n, 0, 1)
y = z + 0.2 * rnorm(n, 0, 1)
x1 = z + rnorm(n, 0, 1)
x2 = z + rnorm(n, 0, 1)
x3 = x1 + x2
d = data.frame(y = y, x1 = x1, x2 = x2, x3 = x3)
```

OLS

```
# OLS fail to calculate coefficient for x3
ols.model = lm(y ~ . - 1, d)
coef(ols.model)
```

```
##     x1     x2     x3
## 0.3053 0.3187     NA
```

Ridge Regression

```
# choose tuning parameter
ridge.model = lm.ridge(y ~ . - 1, d, lambda = seq(0, 10, 0.1))
lambda.opt = ridge.model$lambda[which.min(ridge.model$GCV)]
# ridge regression (shrink coefficients)
coef(lm.ridge(y ~ . - 1, d, lambda = lambda.opt))
```

```
##     x1     x2     x3
## 0.1771 0.1902 0.1258
```

Approximately multicollinear

- show that ridge regreesion correct coefficient signs and reduce mean squared error

```
x3 = x1 + x2 + 0.05 * rnorm(n, 0, 1)
d = data.frame(y = y, x1 = x1, x2 = x2, x3 = x3)
d.train = d[1:400, ]
d.test = d[401:500, ]
```

OLS

```
ols.train = lm(y ~ . - 1, d.train)
coef(ols.train)
```

```
##      x1      x2      x3
## -0.3764 -0.3522  0.6839
```

```
# prediction errors
sum((d.test$y - predict(ols.train, newdata = d.test))^2)
```

```
## [1] 37.53
```

Ridge Regression

```
# choose tuning parameter for ridge regression
ridge.train = lm.ridge(y ~ . - 1, d.train, lambda = seq(0, 10, 0.1))
lambda.opt = ridge.train$lambda[which.min(ridge.train$GCV)]
ridge.model = lm.ridge(y ~ . - 1, d.train, lambda = lambda.opt)
coef(ridge.model)  # correct signs
```

```
##     x1     x2     x3
## 0.1713 0.1936 0.1340
```

```
coefs = coef(ridge.model)
sum((d.test$y - as.matrix(d.test[, -1]) %*% matrix(coefs, 3, 1))^2)
```

```
## [1] 36.87
```

LASSO

$$\hat{\beta}^{\text{lasso}} = \arg\min_{\beta} \{ \|Y - X\beta\|_2^2 + \lambda \|\beta\|_1 \}$$

Or equivalently

$$\min_{\beta} \|Y - X\beta\|_2^2 \quad \text{s.t.} \quad \|\beta\|_1 = \sum_{j=1}^{p} |\beta_j| \leq t$$

Pros

- allow p >> n
- enforce sparcity in parameters
- quadratic programming problem, lars solution requires $O(np^2)$
- λ goes to 0, t goes to ∞, OLS solution
- λ goes to ∞, t goes to 0, $\hat{\beta} = 0$

Cons

- if a group of predictors are highly correlated among themselves, LASSO tends to pick only one of them and shrink the other to zero
- can not do grouped selection, tend to select one variable

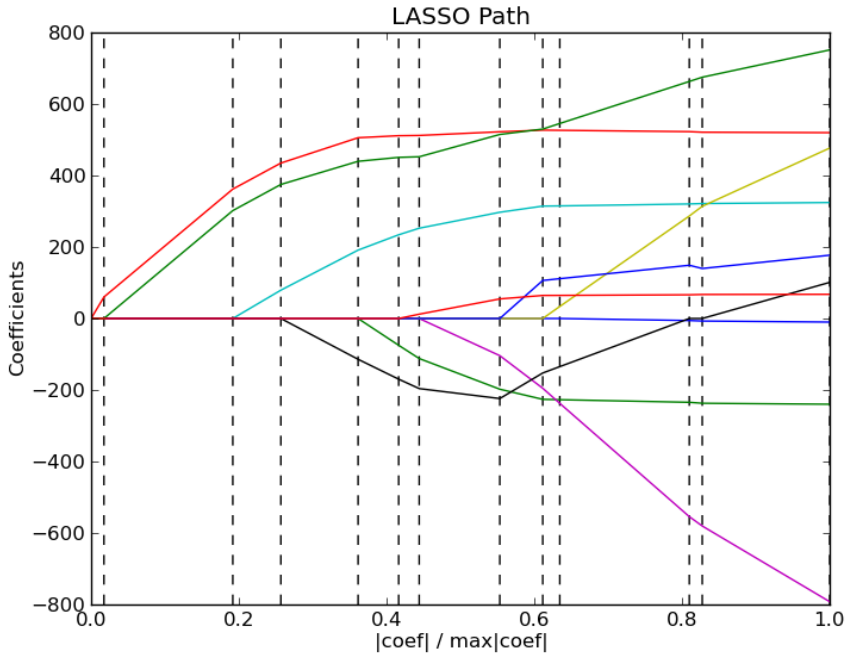
LASSO is good for eliminating trivial genes but not good for grouped selection

LARS algorithm of Efron et al (2004)

- stepwise variable selection (Least angle regression and shrinkage)
- less greedy version of traditional forward selection methods
- solve the entire lasso solution path efficiently
- same order of computational efforts as a single OLS fit $O(np^2)$

LARS Path

$$\min_{\beta} \|Y - X\beta\|_2^2 \quad \text{s.t.} \quad \|\beta\|_1 \leq s\|\hat{\beta}^{\text{OLS}}\|_1, s \in [0, 1]$$



parsimonious model

```
library(MASS)
n = 20
# beta is sparse
beta = matrix(c(3, 1.5, 0, 0, 2, 0, 0, 0), 8, 1)
p = length(beta)
rho = 0.3
corr = matrix(0, p, p)
for (i in seq(p)) {
    for (j in seq(p)) {
        corr[i, j] = rho^abs(i - j)
    }
}
X = mvrnorm(n, mu = rep(0, p), Sigma = corr)
y = X %*% beta + 3 * rnorm(n, 0, 1)
d = as.data.frame(cbind(y, X))
colnames(d) = c("y", paste0("x", seq(p)))
```

OLS

```
n.sim = 100
mse = rep(0, n.sim)
for (i in seq(n.sim)) {
    X = mvrnorm(n, mu = rep(0, p), Sigma = corr)
    y = X %*% beta + 3 * rnorm(n, 0, 1)
    d = as.data.frame(cbind(y, X))
    colnames(d) = c("y", paste0("x", seq(p)))
    # fit OLS without intercept
    ols.model = lm(y ~ . - 1, d)
    mse[i] = sum((coef(ols.model) - beta)^2)
}
median(mse)
```

```
## [1] 6.32
```

Ridge Regression

```
n.sim = 100
mse = rep(0, n.sim)
for (i in seq(n.sim)) {
    X = mvrnorm(n, mu = rep(0, p), Sigma = corr)
    y = X %*% beta + 3 * rnorm(n, 0, 1)
    d = as.data.frame(cbind(y, X))
    colnames(d) = c("y", paste0("x", seq(p)))
    ridge.cv = lm.ridge(y ~ . - 1, d, lambda = seq(0, 10, 0.1))
    lambda.opt = ridge.cv$lambda[which.min(ridge.cv$GCV)]
    # fit ridge regression without intercept
    ridge.model = lm.ridge(y ~ . - 1, d, lambda = lambda.opt)
    mse[i] = sum((coef(ridge.model) - beta)^2)
}
median(mse)
```

```
## [1] 4.074
```

LASSO

```
library(elasticnet)
n.sim = 100
mse = rep(0, n.sim)
for (i in seq(n.sim)) {
    X = mvrnorm(n, mu = rep(0, p), Sigma = corr)
    y = X %*% beta + 3 * rnorm(n, 0, 1)
    obj.cv = cv.enet(X, y, lambda = 0, s = seq(0.1, 1, length = 100), plot.it = FALSE,
        mode = "fraction", trace = FALSE, max.steps = 80)
    s.opt = obj.cv$s[which.min(obj.cv$cv)]
    lasso.model = enet(X, y, lambda = 0, intercept = FALSE)
    coefs = predict(lasso.model, s = s.opt, type = "coefficients", mode = "fraction")
    mse[i] = sum((coefs$coefficients - beta)^2)
}
median(mse)
```

```
## [1] 3.393
```

Elastic Net

$$\hat{\beta}^{\text{enet}} = \arg\min_{\beta} \{(Y - X\beta)^T (Y - X\beta) + \lambda_1 \|\beta\|_1 + \lambda_2 \|\beta\|_2^2\}$$

Pros

- enforce sparsity
- no limitation on the number of selected variable
- encourage grouping effect in the presence of highly correlated predictors

Cons

- naive elastic net suffers from double shrinkage

Correction

$$\hat{\beta}_{enet} = (1 + \lambda_2)\hat{\beta}$$

LASSO vs Elastic Net

Construct a data set with grouped effects to show that Elastic Net outperform LASSO in grouped selection

- response y
- 6 predictors fall into two group, x_1, x_2, x_3 as dominant factors, x_4, x_5, x_6 as minor factors we would like to shrink to zero

Two independent "hidden" factors z_1 and z_2

$$y = z_1 + 0.1 * z_2 + N(0, 1)$$

Correlated grouped covariates

$$x_1 = z_1 + \epsilon_1, x_2 = -z_1 + \epsilon_2, x_3 = z_1 + \epsilon_3$$

$$x_4 = z_2 + \epsilon_4, x_5 = -z_2 + \epsilon_5, x_6 = z_2 + \epsilon_6$$

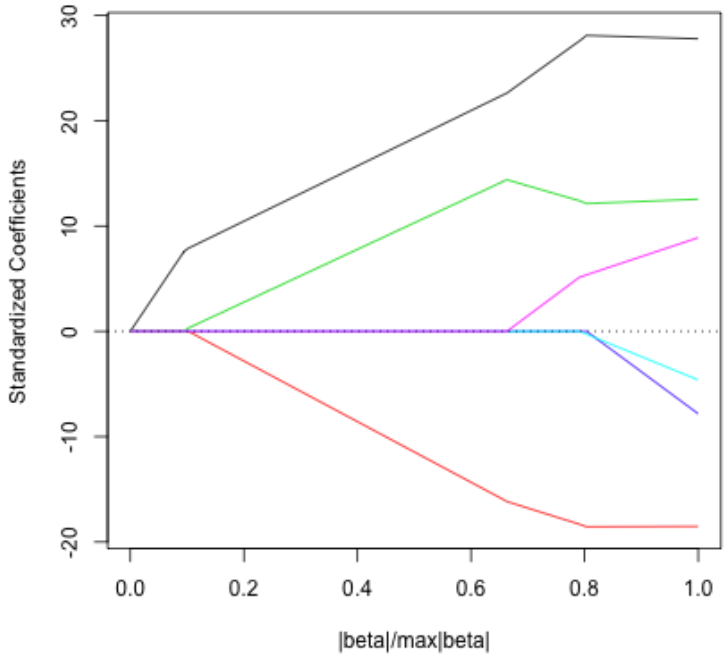
$$X = (x_1, x_2, \cdots, x_6)$$

Simulated data

```
N = 100
z1 = runif(N, min = 0, max = 20)
z2 = runif(N, min = 0, max = 20)
y = z1 + 0.1 * z2 + rnorm(N)
X = cbind(z1 %*% matrix(c(1, -1, 1), 1, 3), z2 %*% matrix(c(1, -1, 1), 1, 3))
X = X + matrix(rnorm(N * 6), N, 6)
```

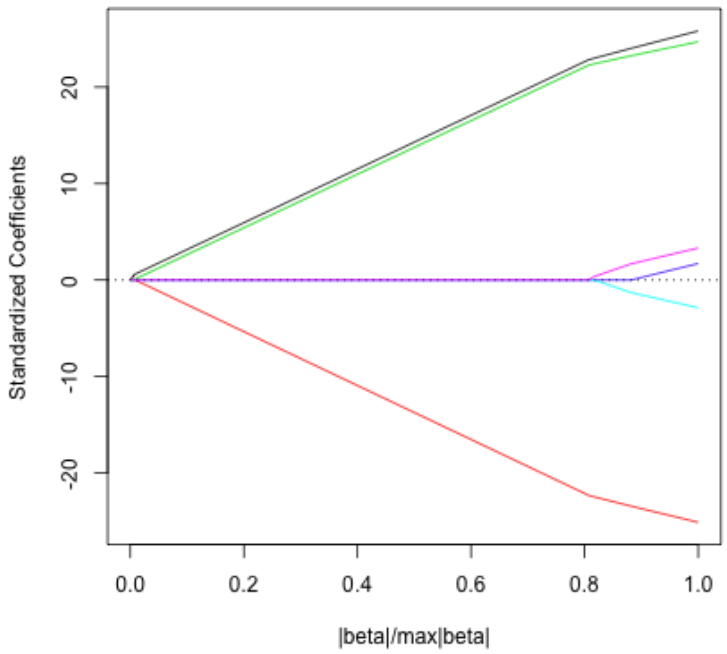
LASSO path

```
library(elasticnet)
obj.lasso = enet(X, y, lambda = 0)
plot(obj.lasso, use.color = TRUE)
```



Elastic Net

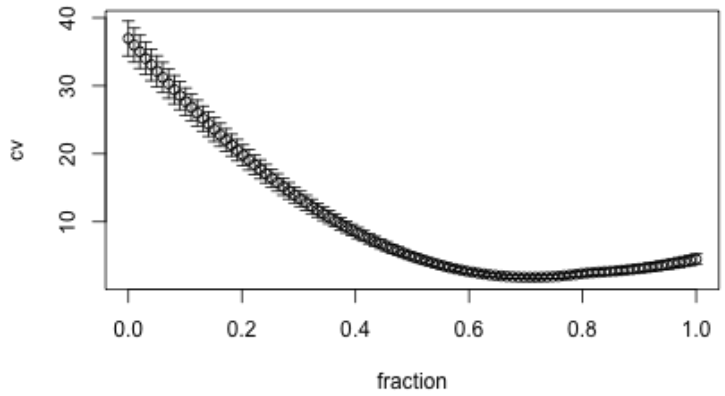
```
library(elasticnet)
obj.enet = enet(X, y, lambda = 0.5)
plot(obj.enet, use.color = TRUE)
```



How to choose tuning parameter

For a sequence of λ, find the s that minimizer of the CV prediction error and then find the λ which minimize the CV prediction error

```
library(elasticnet)
obj.cv = cv.enet(X, y, lambda = 0.5, s = seq(0, 1, length = 100), mode = "fraction",
    trace = FALSE, max.steps = 80)
```



Prostate Cancer Example

- Predictors are eight clinical measures
- Training set with 67 observations
- Test set with 30 observations
- Modeling fitting and turning parameter selection by tenfold CV on training set
- Compare model performance by prediction mean-squared error on the test data

Compare models

Method	*Parameter(s)*	*Test mean-squared error*	*Variables selected*
OLS		0.586 (0.184)	All
Ridge regression	$\lambda = 1$	0.566 (0.188)	All
Lasso	$s = 0.39$	0.499 (0.161)	(1,2,4,5,8)
Naïve elastic net	$\lambda = 1, s = 1$	0.566 (0.188)	All
Elastic net	$\lambda = 1000, s = 0.26$	0.381 (0.105)	(1,2,5,6,8)

- medium correlation among predictors and the highest correlation is 0.76
- elastic net beat LASSO and ridge regression beat OLS

Summary

- Ridge Regression:
  - good for multicollinearity, grouped selection
  - not good for variable selection
- LASSO
  - good for variable selection
  - not good for grouped selection for strongly correlated predictors
- Elastic Net
  - combine strength between Ridge Regression and LASSO
- Regularization
  - trade bias for variance reduction
  - better prediction accuracy

Reference

Most of the materials covered in this slides are adapted from

- Paper: Regularization and variable selection via the elastic net
- Slide: http://www.stanford.edu/~hastie/TALKS/enet_talk.pdf
- The Elements of Statistical Learning

Exercise 1: simulated data

```
beta = matrix(c(rep(3, 15), rep(0, 25)), 40, 1)
sigma = 15
n = 500
z1 = matrix(rnorm(n, 0, 1), n, 1)
z2 = matrix(rnorm(n, 0, 1), n, 1)
z3 = matrix(rnorm(n, 0, 1), n, 1)
X1 = z1 %*% matrix(rep(1, 5), 1, 5) + 0.01 * matrix(rnorm(n * 5), n, 5)
X2 = z2 %*% matrix(rep(1, 5), 1, 5) + 0.01 * matrix(rnorm(n * 5), n, 5)
X3 = z3 %*% matrix(rep(1, 5), 1, 5) + 0.01 * matrix(rnorm(n * 5), n, 5)
X4 = matrix(rnorm(n * 25, 0, 1), n, 25)
X = cbind(X1, X2, X3, X4)
Y = X %*% beta + sigma * rnorm(n, 0, 1)
Y.train = Y[1:400]
X.train = X[1:400, ]
Y.test = Y[400:500]
X.test = X[400:500, ]
```

Questions:

- Fit OLS, LASSO, Ridge regression and elastic net to the training data and calculate the prediction error from the test data
- Simulate the data set for 100 times and compare the median mean-squared errors for those models

Exercise 2: Diabetes

- x a matrix with 10 columns
- y a numeric vector (442 rows)
- x2 a matrix with 64 columns

```
library(elasticnet)
data(diabetes)
colnames(diabetes)
```

```
## [1] "x"  "y"  "x2"
```

Questions

- Fit LASSO and Elastic Net to the data with optimal tuning parameter chosen by cross validation.
- Compare solution paths for the two methods