LASSO Regression

CMED6040 – Session 8

Tim Tsang (matklab@hku.hk)

School of Public Health
The University of Hong Kong

11 July 2023

Session 8 learning objectives

After this session, students should be able to

- Understand the bias-variance tradeoff in model selection
- Perform model selection using ridge, LASSO and elastic net regressions
- Recognize pros and cons of various model selection approaches

Model selection

Modelling an outcome variable on a set of predictor variables

- Select a subset of the candidate predictor variables
- Objective: simple, interpretable model with accurate predictions
- Goodness of fit
  - How similar the observed and predicted values are
  - Measured by R-squared, AIC, BIC, etc
- Parsimony
  - Occam's razor: for models with similar goodness of fit, the simpler one is preferred
- Data sets
  - Training set for developing the model
  - Testing set for examining the model in another data set

Example – polynomial regression

- Consider a simple sinusoidal model

$$Y = 3\sin(X) + \varepsilon$$

where $\varepsilon \sim Normal(0, 1)$

- Generate training data

```
> set.seed(6789)
> x <- c(0:8) / 8 * 2 * pi
> y <- 3 * sin(x) + rnorm(n=9)
> data <- data.frame(x, y)
```

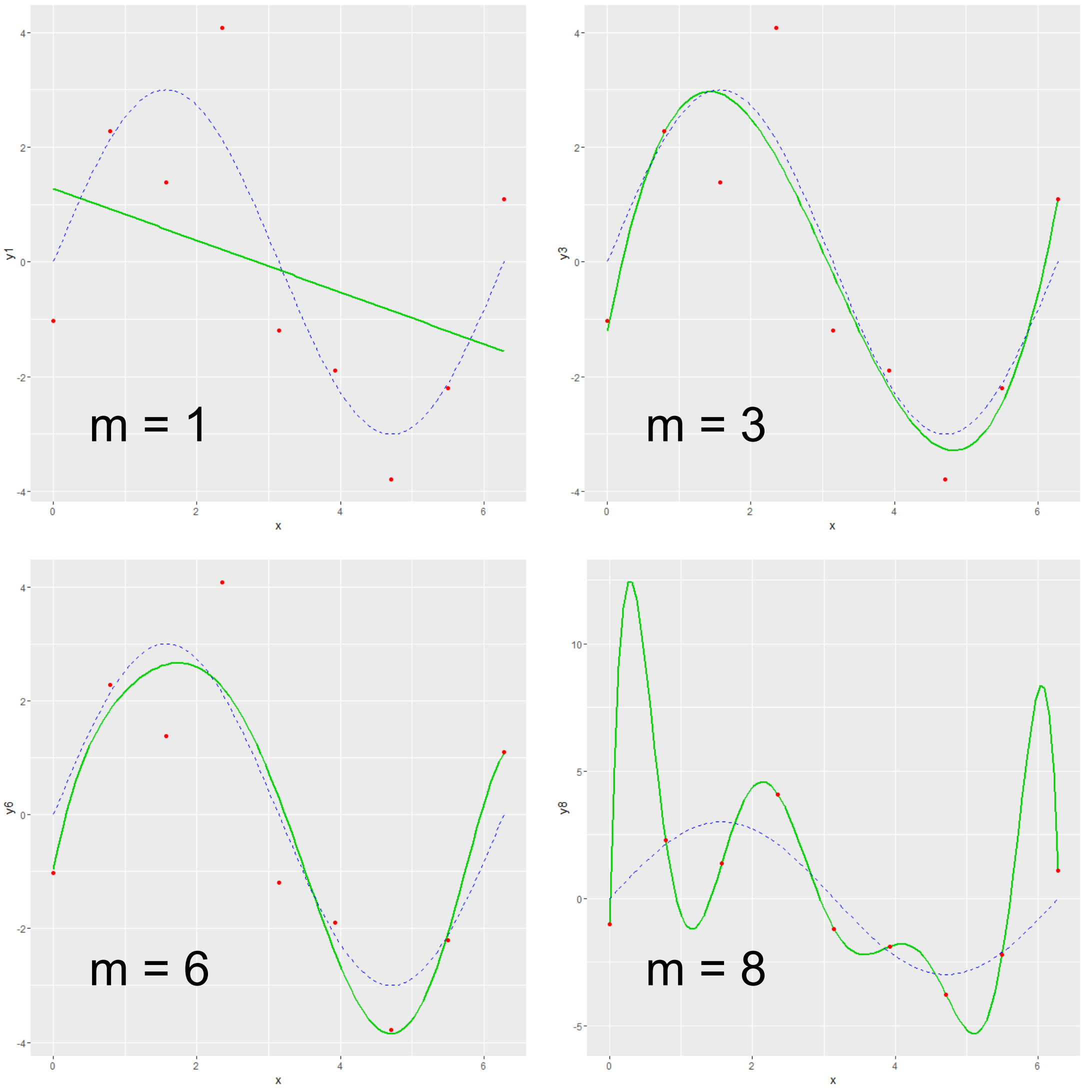
- Fit a polynomial regression model

$$y = \alpha + \sum_{k=1}^{m} \beta_k x^k$$

Example – polynomial regression

Which model is the best?

```
> lm1 <-lm(y ~ x, data = data)
> lm3 <-lm(y ~ x + I(x^2) + I(x^3),
              data = data)
> lm6 <-lm(y ~ x + I(x^2) + I(x^3) +
              I(x^4) + I(x^5) +
              I(x^6), data = data)
> lm8 <-lm(y ~ x + I(x^2) + I(x^3) +
              I(x^4) + I(x^5) +
              I(x^6) + I(x^7) +
              I(x^8), data = data)
```



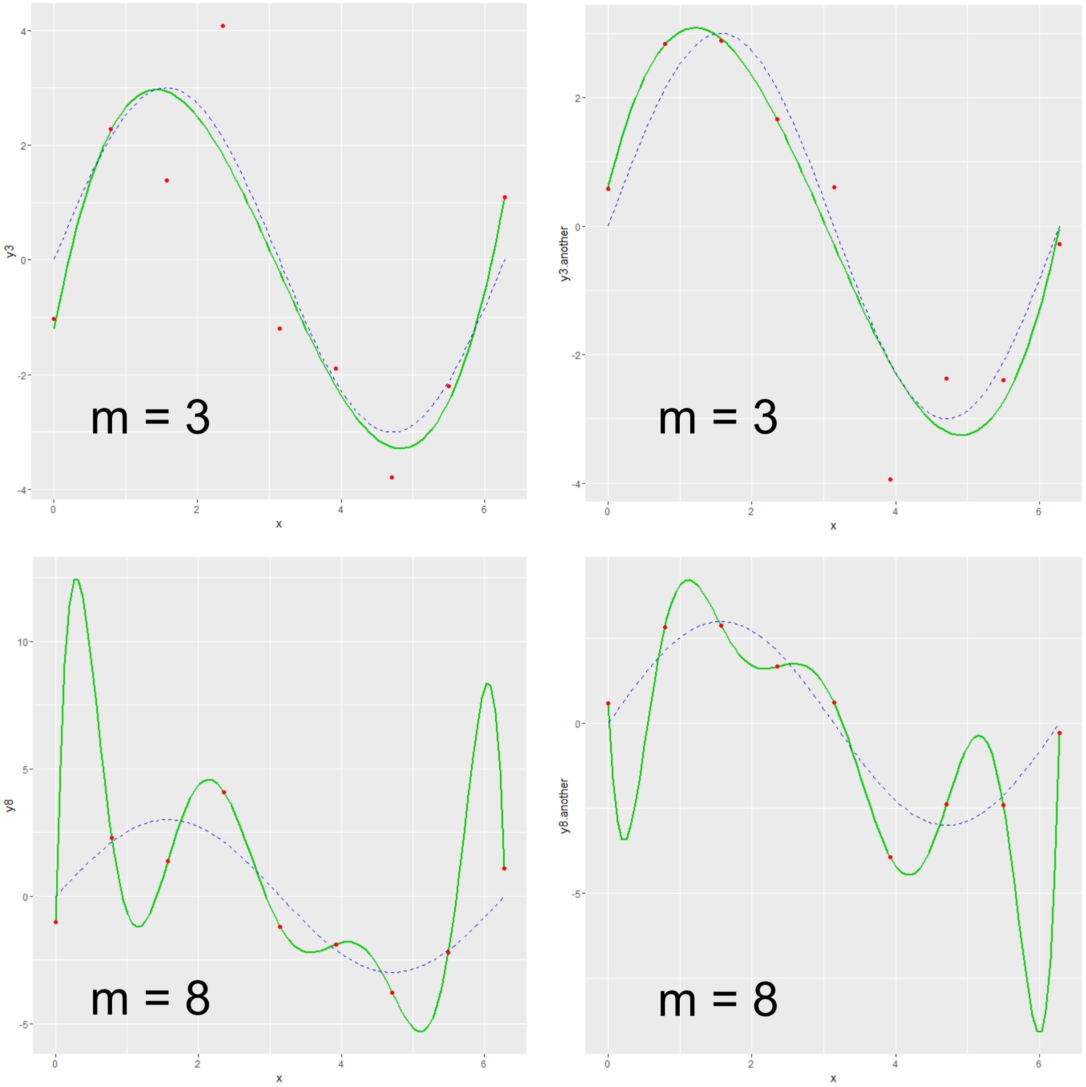
Over-fitting

- The model with m = 8 perfectly predicts the observed values, but…
- There are random fluctuations (or noise, errors) in the data
- Over-fitting causes the model to fit the noise rather than the actual underlying structure
- Too sensitive to the data that the fitted model would be very different if another set of training data is used
- May result in large prediction error if the model is applied to a set of testing data

Another training data set

- Generate training data

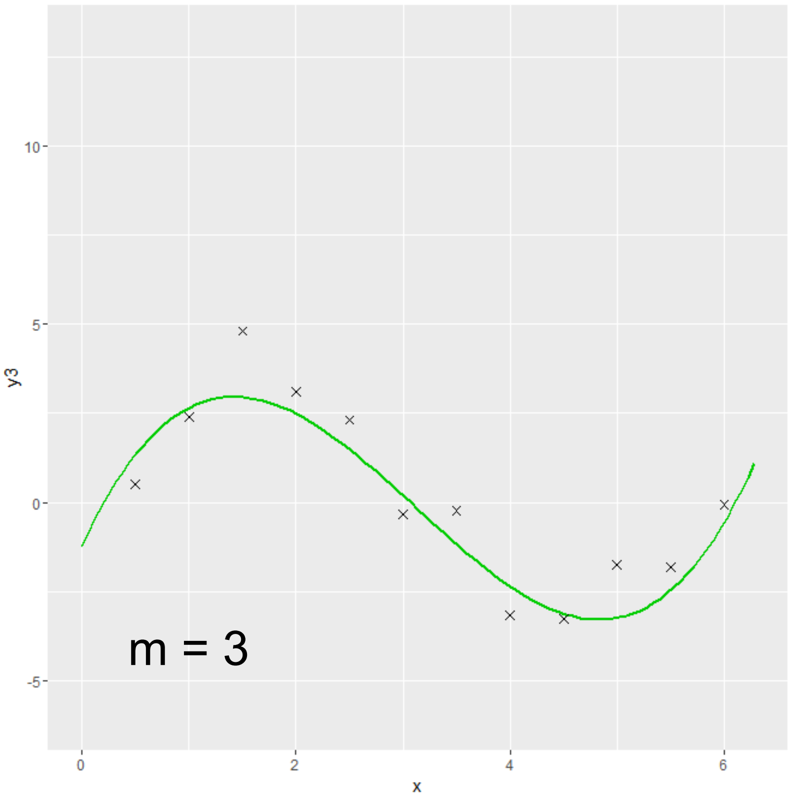
```
> set.seed(12345)
> x <- c(0:8) / 8 * 2 * pi
> y <- 3 * sin(x) + rnorm(n=9)
> another <- data.frame(x, y)
```

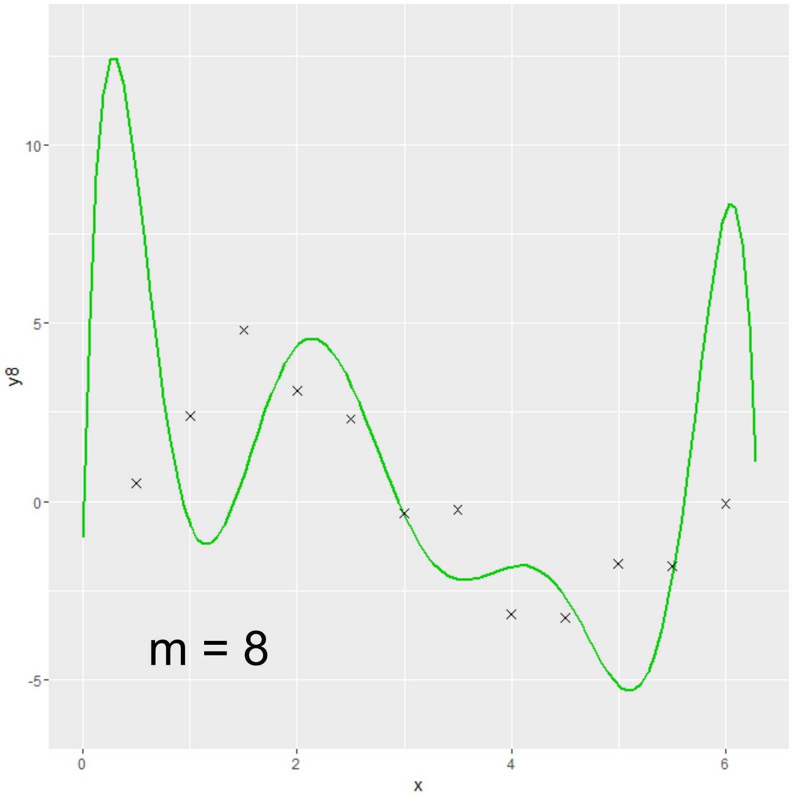


Testing data

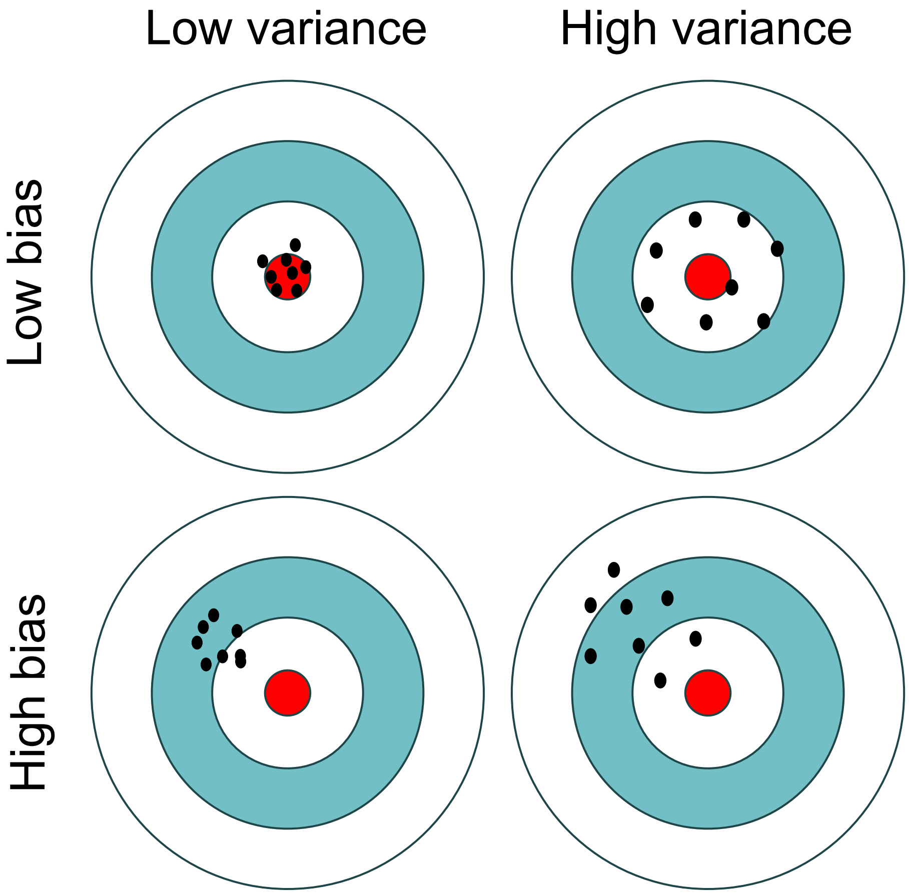
- Generate testing data

```
> x <- c(1:12) / 2
> y <- 3 * sin(x) + rnorm(n=12)
> test <- data.frame(x, y)
```

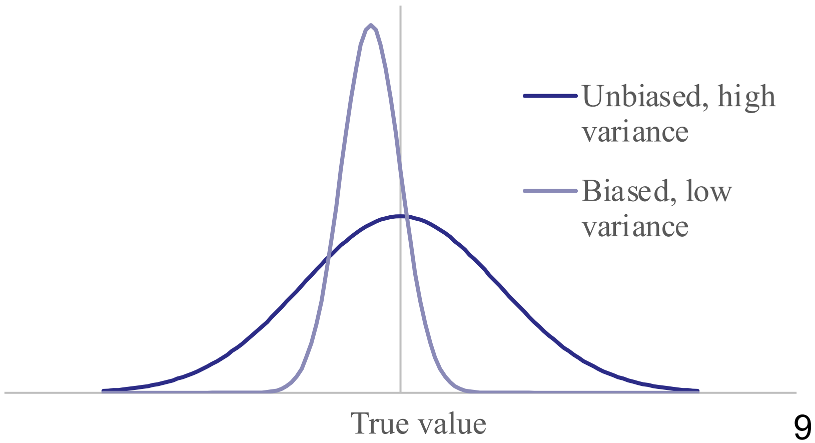




Bias and variance



- Bias - how far the predicted values deviated from the true
- Variance - how scattered the predicted values are
- Ideally low bias and low variance
- But in practical there is a tradeoff between bias and variance



Bias-variance decomposition

- Observed value $Y = f(X) + \varepsilon$, where $E(\varepsilon) = 0, Var(\varepsilon) = \sigma^2$
- Predicted value $\hat{Y} = \hat{f}(X)$ is a random variable
- Bias = $f(X) - E[\hat{f}(X)]$ = Expected difference between true and predicted
- Expectation of squared prediction error $E\left[\left(Y - \hat{Y}\right)^2\right]$

$$= E\left[\left(f(X) + \varepsilon - \hat{f}(X)\right)^2\right]$$

$$= E[\varepsilon^2] + 2E\left[\varepsilon\left(f(X) - \hat{f}(X)\right)\right] + E\left[\left(f(X) - \hat{f}(X)\right)^2\right]$$

(Grey part equals zero: $E\left[\varepsilon\left(f(X) - \hat{f}(X)\right)\right] = E[\varepsilon]E\left[\left(f(X) - \hat{f}(X)\right)\right] = 0$)

$$= E[\varepsilon^2] + E\left[\left(f(X) - E[\hat{f}(X)] + E[\hat{f}(X)] - \hat{f}(X)\right)^2\right]$$ (Grey part equals zero)

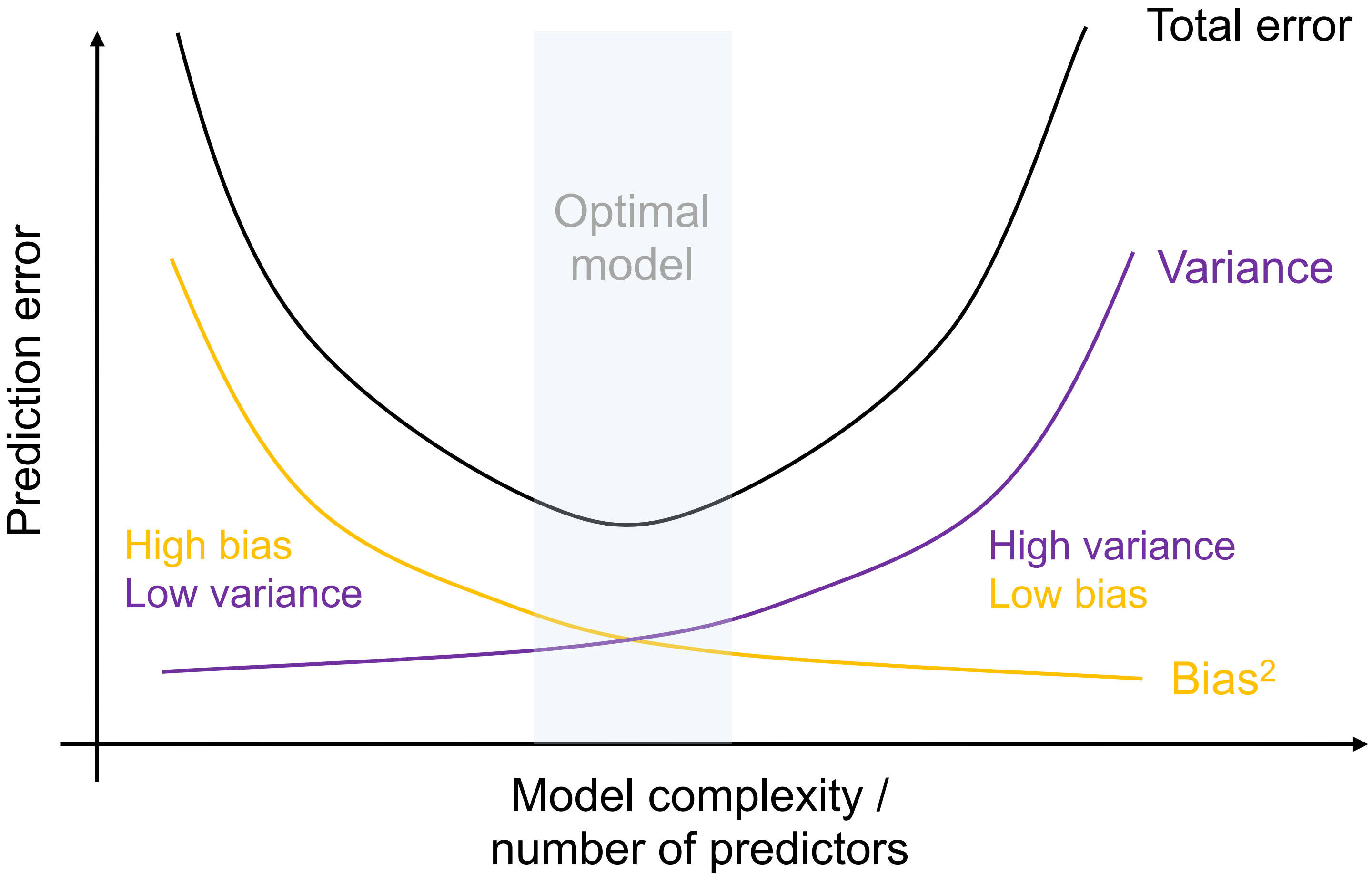
$$= E[\varepsilon^2] + E\left[\left(f(X) - E[\hat{f}(X)]\right)^2\right] + E\left[\left(\hat{f}(X) - E[\hat{f}(X)]\right)^2\right]$$ (Grey part is constant)

$$\mathrm{E}\left[\left(f(X) - E[\hat{f}(X)]\right)\left(E[\hat{f}(X)] - \hat{f}(X)\right)\right] = \left(f(X) - E[\hat{f}(X)]\right)\mathrm{E}\left[\left(E[\hat{f}(X)] - \hat{f}(X)\right)\right] = 0$$

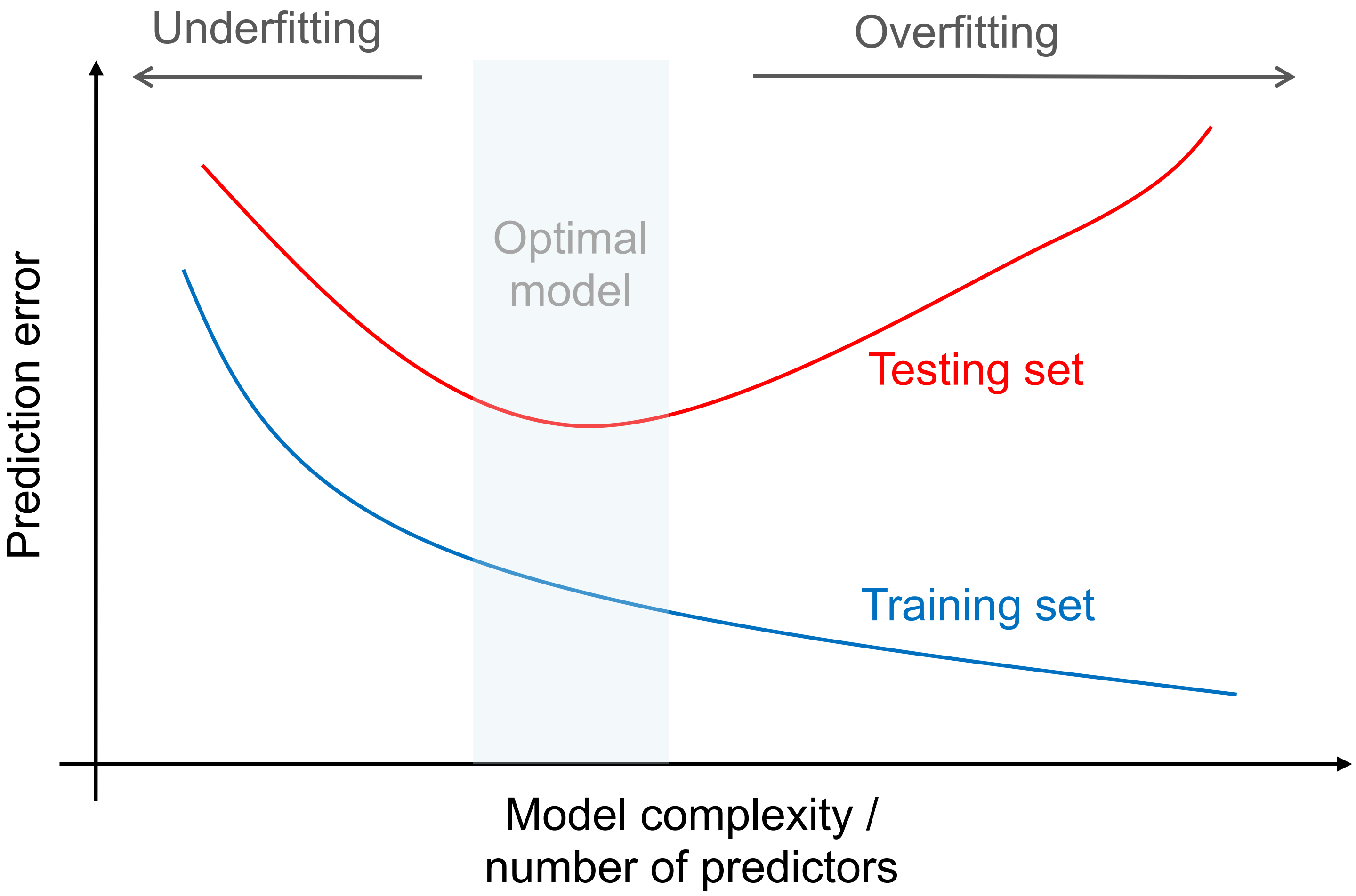
$$= \sigma^2 + \left(f(X) - E[\hat{f}(X)]\right)^2 + Var\left(\hat{f}(X)\right)$$

= Noise^2 + Bias^2 + Variance

Bias-variance tradeoff



Training and testing sets



Example – Prostate cancer

- Source: Stamey, T.A., et al. (1989) Prostate specific antigen in the diagnosis and treatment of adenocarcinoma of the prostate: II. radical prostatectomy treated patients, Journal of Urology 141(5), 1076–1083.

Variable	Description	Data type	Level
lcavol	log(cancer volume)	Continuous	
lweight	log(prostate weight)	Continuous	
age	age	Continuous	
lbph	log(benign prostatic hyperplasia amount)	Continuous	
svi	seminal vesicle invasion	Categorical	1: Yes, 0: No
lcp	log(capsular penetration)	Continuous	
gleason	Gleason score	Continuous	
pgg45	percentage Gleason scores 4 or 5	Continuous	
lpsa	log(prostate specific antigen) (Outcome)	Continuous	

Data sets

- Read data and assign factors

```
Prostate <- read.csv("prostate.csv")
```

- Divide into training and testing sets

```
set.seed(23456)

TestingIndex <- sample(1:nrow(Prostate), 20)

Training <- Prostate[-TestingIndex, ]

Testing <- Prostate[TestingIndex, ]
```

Automated selection procedure

- Backward selection

```
Full <- lm(lpsa ~ lcavol + lweight + age + lbph + svi + lcp + gleason + pgg45,
             data = Training)
Backward1 <- step(Full, direction = "backward")

lpsa ~ lcavol + lweight + age + lbph + svi + lcp + pgg45

          Df Sum of Sq    RSS     AIC
<none>                 38.794 -42.787
- lbph     1    2.0087 40.803 -40.900
- lweight  1    2.9786 41.773 -39.091
- svi      1    5.8688 44.663 -33.939
- lcavol   1   27.6768 66.471  -3.322
```

Automated selection procedure

- Alternatives

A period "." means all other variables in the data set, a minus "–" excludes a variable

```
Alt <- lm(lpsa ~ . - age - svi, data = Training)
```

Add F-test or Chi-squared test in the output

```
Backward2 <- step(Full, direction = "backward", test = "F")
Backward3 <- step(Full, direction = "backward", test = "Chisq")
```

- Forward and stepwise selection

Forward selection

```
Null <- lm(lpsa ~ 1, data = Training)
Forward <- step(Null, scope = list(lower = Null, upper = Full),
                  direction = "forward")
```

Stepwise selection

```
Stepwise1 <- step(Full, direction = "both")
Stepwise2 <- step(Null, scope = list(lower = Null, upper = Full),
                    direction = "both")
```

Ordinary least squares

- At each step of the above selection procedures, the coefficients were found by ordinary least squares (OLS), i.e. minimizing the sum of squared residuals:

$$\sum_{i=1}^{N}\left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j\right)^2$$

- Unbiased
- Alternative: Penalized regression
  - Bias-variance trade-off: introduce bias to reduce variance

Penalized regression

- Minimize the sum of squared residuals

$$\sum_{i=1}^{N}\left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j\right)^2$$

subject to a constraint

Ridge regression: $\sum_{j=1}^{p} \beta_j^2 \leq t_1$

LASSO regression: $\sum_{j=1}^{p} |\beta_j| \leq t_2$

where t_1 *and* $t_2 > 0$ are penalty parameters

- No longer unbiased, but reduces the variance

Ridge regression

- Minimize the sum of squared residuals subject to

$$\sum_{j=1}^{p} \beta_j^2 \leq t$$

  where *t* > 0 is a penalty parameter

- This is equivalent to find the set of coefficients β_j that minimizes the objective function with various non-negative values of λ:

$$\sum\nolimits_{i=1}^{N} \left(y_i - \beta_0 - \sum\nolimits_{j=1}^{p} x_{ij}\beta_j \right)^2 + \lambda \sum_{j=1}^{p} \beta_j^2$$

- As λ → 0, ridge coefficients → OLS coefficients
- As λ → ∞, ridge coefficients → 0

Ridge and LASSO regression in R

- Package: glmnet
- glmnet(x, y, alpha, lambda, family…)
  - x: matrix of predictor variables
  - y: response variable
  - alpha: 0 = Ridge regression, 1 = LASSO regression
  - lambda: user-supplied λ values
  - family: Response type, e.g. “gaussian”, “binomial”, etc

Ridge regression

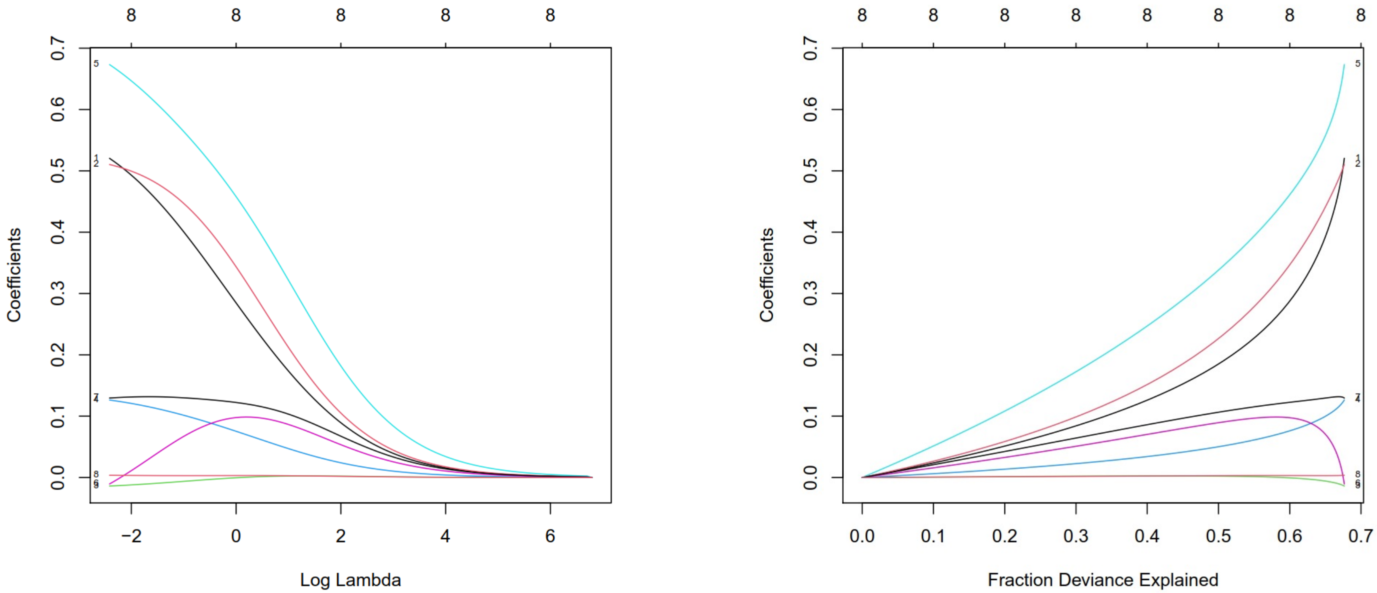
```
library(glmnet)
x <- model.matrix(lpsa ~ . - 1, data = Training)
y <- Training$lpsa
Ridge1 <- glmnet(x, y, alpha = 0, lambda = c(0.5, 1))
Ridge1

Call:  glmnet(x = x, y = y, alpha = 0, lambda = c(0.5, 1))

     Df   %Dev Lambda
[1,]  8  59.80    1.0
[2,]  8  64.06    0.5

coef(Ridge1)

                       s0           s1
(Intercept) -0.1042577829 -0.309343110
lcavol       0.2846293551  0.366144109
lweight      0.3435344292  0.420581866
age         -0.0004531201 -0.004340865
lbph         0.0753331555  0.094017529
svi          0.4579589510  0.534373698
lcp          0.0977048229  0.080735763
gleason      0.1222559227  0.128152107
pgg45        0.0031661125  0.003072614
```

Ridge regression

- Not specifying λ

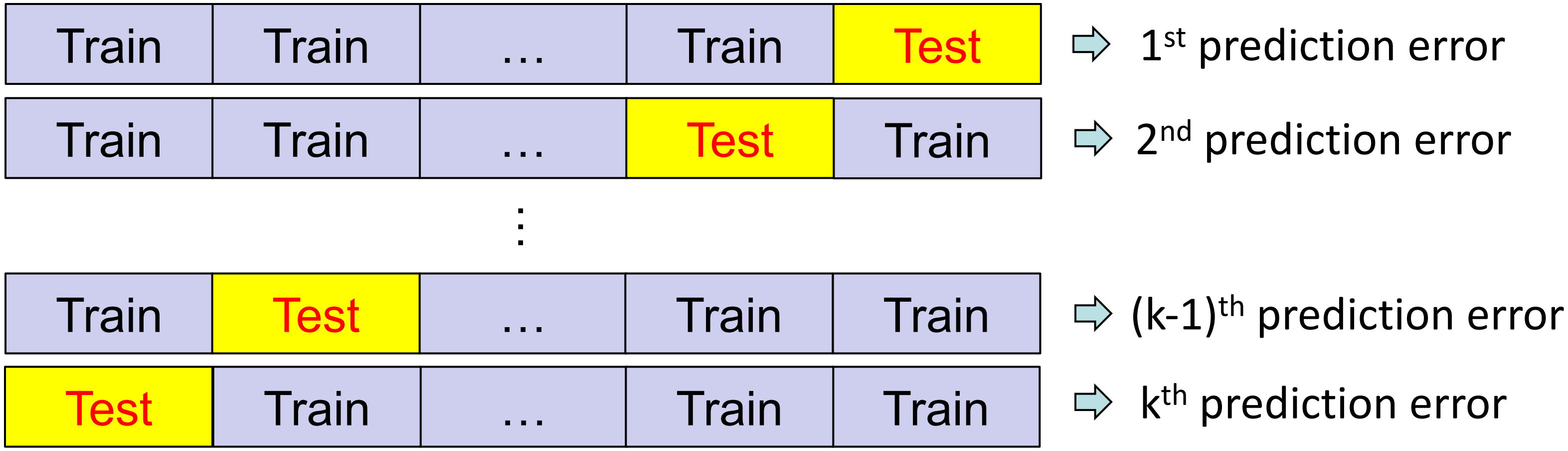
```
Ridge2 <- glmnet(x, y, alpha = 0)
plot(Ridge2, xvar="lambda", label=TRUE)
plot(Ridge2, xvar="dev", label=TRUE)
```

- Magnitude of coefficients
  - Decrease with λ, approach to (but not become) 0 due to the penalty $\lambda \sum_{i=1}^{m} \beta_i^2$
  - Overall, increase with the proportion of deviance explained



k-fold Cross-validation

- Randomly divide the data set into k (usually 10) equal-sized subsets
  - Use (k – 1) subsets for estimation of model (training)
  - The remaining subset for computation of prediction error (testing)



Train	Train	…	Train	Test	⇨	1st prediction error
Train	Train	…	Test	Train	⇨	2nd prediction error
		⋮				
Train	Test	…	Train	Train	⇨	(k-1)th prediction error
Test	Train	…	Train	Train	⇨	kth prediction error

  - Calculate the mean and standard error of mean squared error (MSE)
- Leave-one-out: k = sample size

Ridge regression

- Cross-validation to determine λ

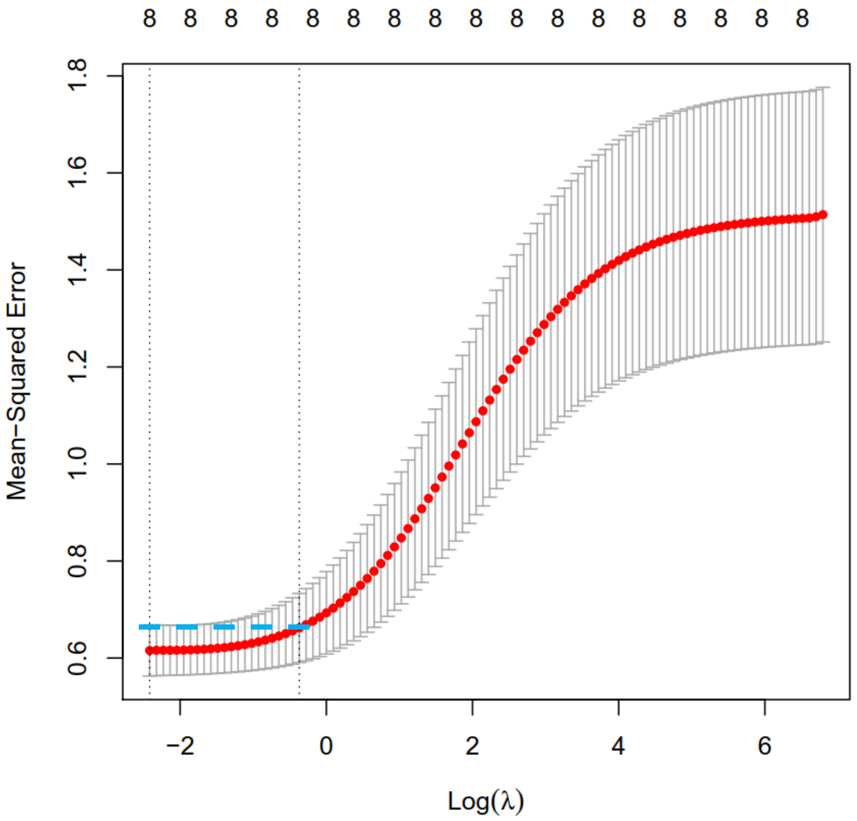
```
set.seed(56789)
Ridge2.cv <- cv.glmnet(x, y, alpha = 0)
plot(Ridge2.cv)
```

- Mean MSE (red dot) ± SE (vertical bar)
- Left vertical line: minimum MSE

```
min(Ridge2.cv$cvm)
[1] 0.615571
log(Ridge2.cv$lambda.min)
[1] -2.416623
```

- Right vertical line: maximum value of λ within 1 SE of the minimum (1-SE rule)

```
log(Ridge2.cv$lambda.1se)
[1] -0.3698812
```



Ridge regression

- Coefficients by 1-SE rule

```
coef(Ridge2, s = Ridge2.cv$lambda.1se)

                      1
(Intercept) -0.234804860
lcavol       0.328430340
lweight      0.387430006
age         -0.002396460
lbph         0.085566483
svi          0.500485900
lcp          0.091304642
gleason      0.125769781
pgg45        0.003117404
```

Predicted values on testing set

```
x.test <- model.matrix(lpsa ~ . - 1, data = Testing)
Ridge2.Pred <- predict(Ridge2, newx = x.test, s = Ridge2.cv$lambda.1se)
```

LASSO regression

- Ridge regression
  - Coefficients shrink towards but never reach 0
  - All predictor variables retain, cannot produce a parsimonious model
- LASSO = Least Absolute Shrinkage and Selection Operator
- Minimize the sum of squared residuals subject to

$$\sum_{j=1}^{p} |\beta_j| \leq t$$

where $t > 0$ is a penalty parameter

- Equivalent to the objective function:

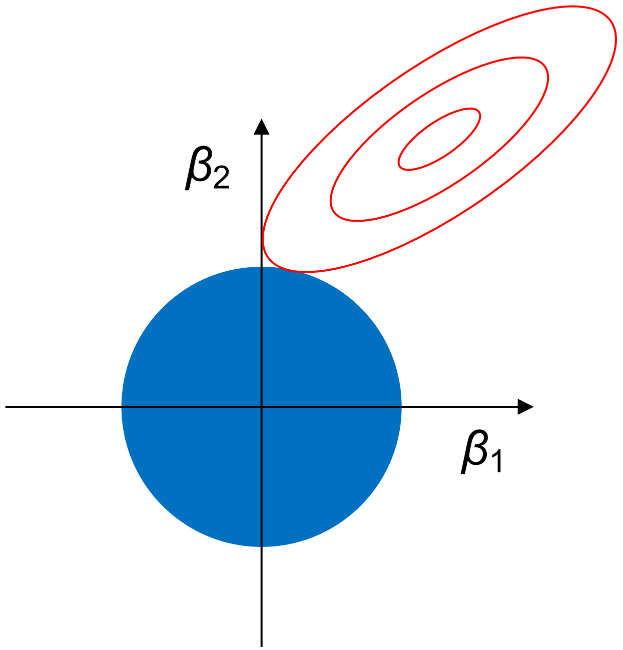
$$\sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 + \lambda \sum_{j=1}^{p} |\beta_j|$$

LASSO regression

- Same package glmnet as ridge regression, but alpha = 1

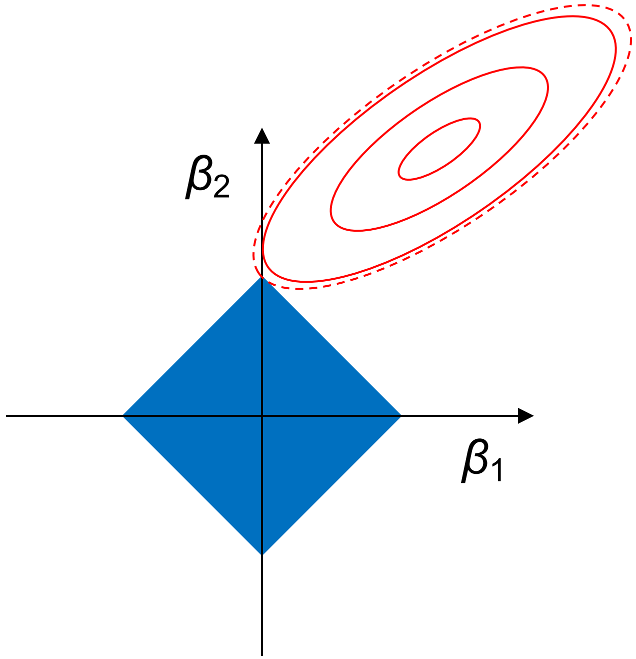
```
LASSO1 <- glmnet(x, y, alpha = 1, lambda = c(0.25, 0.5, 0.75, 1))
LASSO1

     Df    %Dev Lambda
[1,]  0    0.00   1.00
[2,]  1   15.72   0.75
[3,]  1   36.75   0.50
[4,]  3   55.34   0.25

coef(LASSO1)

                  s0        s1        s2        s3
(Intercept) 2.473523 2.3139032 2.0333247 1.1850329
lcavol      .        0.1177542 0.3247419 0.4661090
lweight     .        .         .         0.1598011
age         .        .         .         .
lbph        .        .         .         .
svi         .        .         .         0.3314309
lcp         .        .         .         .
gleason     .        .         .         .
pgg45       .        .         .         .
```

Geometric interpretation



Ridge
$\beta_1^2 + \beta_2^2 \leq t$

both β_1 and β_2
are non-zero



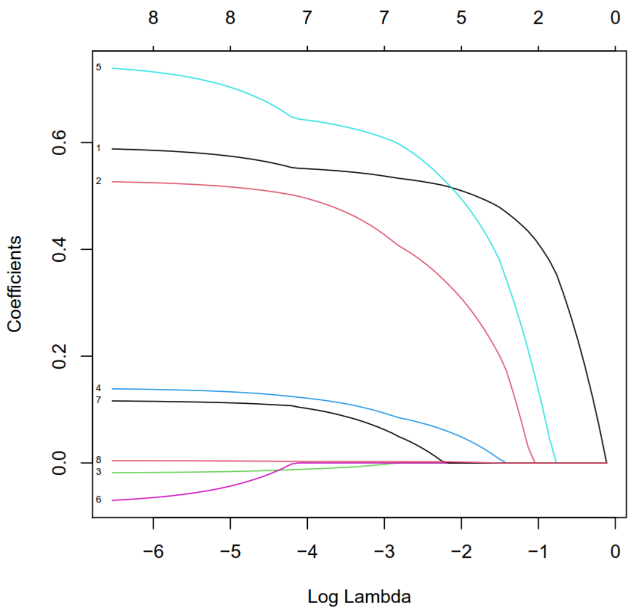
LASSO
$|\beta_1| + |\beta_2| \leq t$

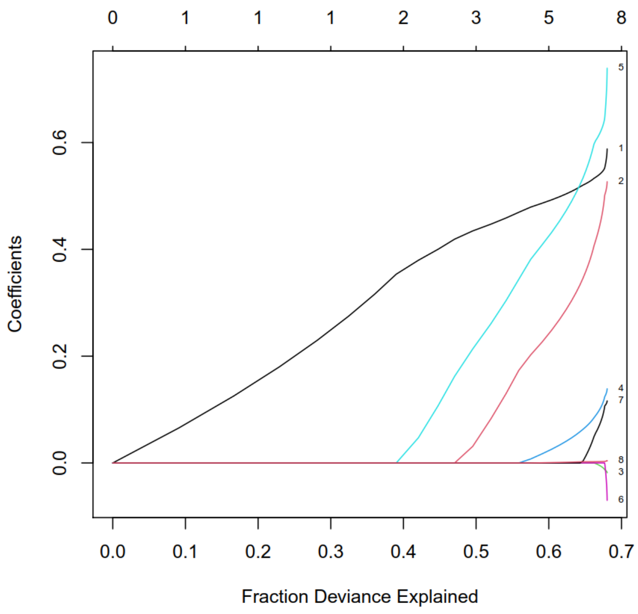
β_1 or β_2 could be
zero

LASSO regression

```
LASSO2 <- glmnet(x, y, alpha = 1)
plot(LASSO2, xvar="lambda", label=TRUE)
plot(LASSO2, xvar="dev", label=TRUE)
```

- Magnitude of coefficients
  - Decrease with λ, same as ridge regression
  - Become 0 one by one (x-axis on top); all become 0 as λ is large (λ = 1 in this example)
  - Overall, increase with the proportion of deviance explained (only intercept when λ = 1)





LASSO regression

- Cross-validation to determine λ

```
set.seed(54321)
LASSO2.cv <- cv.glmnet(x, y, alpha = 1)
plot(LASSO2.cv)
```

- Minimum MSE

```
min(LASSO2.cv$cvm)
[1] 0.6172947
log(LASSO2.cv$lambda.min)
[1] -3.370219
```

- 1-SE rule

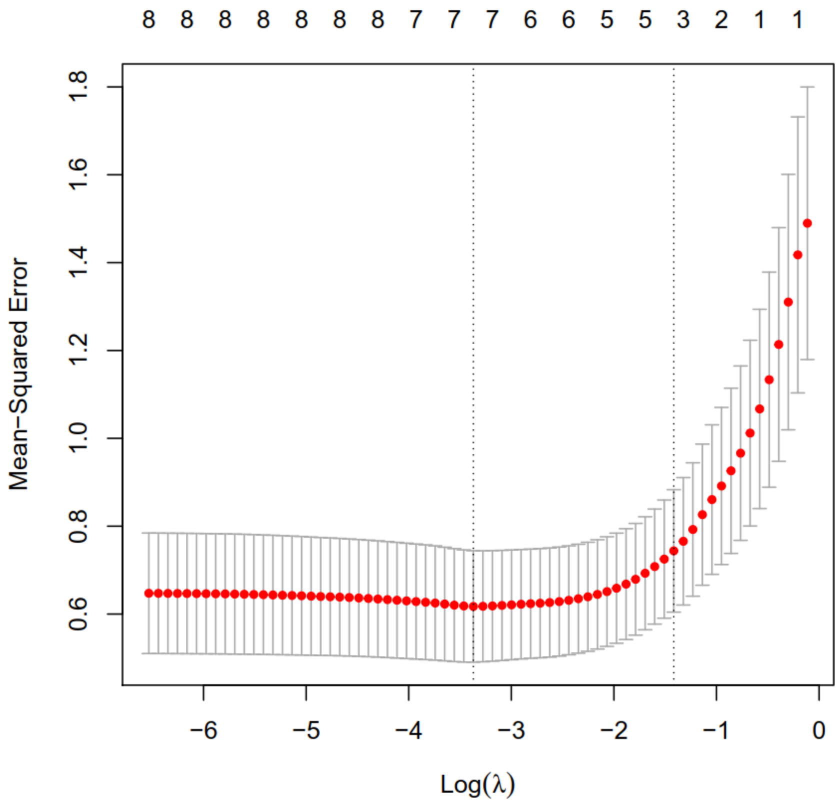
```
log(LASSO2.cv$lambda.1se)
[1] -1.416511
```

- How many non-zero coefficients?

```
coef(LASSO2, s = LASSO2.cv$lambda.1se)
```

Export the predicted values for further analysis

```
LASSO2.Pred <- predict(LASSO2, newx = x.test,
                      s = LASSO2.cv$lambda.1se)
```



Multicollinearity

- Grouped selection
  - A group of (usually correlated) predictor variables included in the model
- Create a new predictor variable which is the sum of two existing ones

```
Prostate.col <- Prostate

Prostate.col$new1 <- Prostate.col$lcavol + Prostate.col$lweight
```

- OLS

```
lm(formula = lpsa ~ lcavol + lweight + new1, data = Prostate.col)
Coefficients:
(Intercept)       lcavol      lweight          new1
    -0.3026       0.6775       0.5109            NA
```

- Ridge

```
 (Intercept) 0.2958473
lcavol       0.2510017
lweight      0.2103176
new1         0.2149762
```

Multicollinearity

- Create another new variable which is nearly the sum of two existing ones

```
set.seed(234567)
Prostate.col$new2 <- Prostate.col$new1 + rnorm(nrow(Prostate.col), 0, 0.1)
```

- OLS

```
lm(formula = lpsa ~ lcavol + lweight + new2, data = Prostate.col)
Coefficients:
(Intercept)       lcavol      lweight         new2
    -0.2495      -0.5894      -0.7710       1.2719
```

Ridge

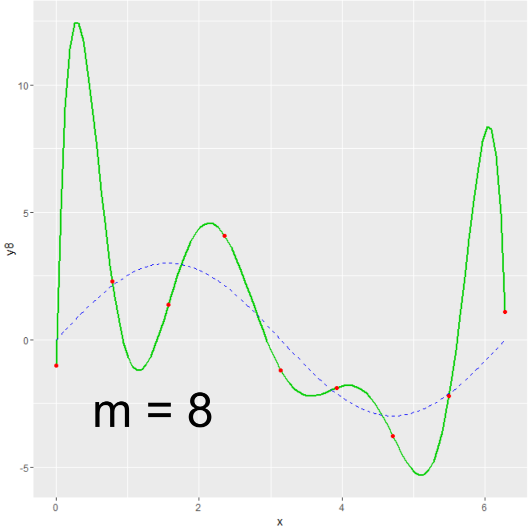
```
(Intercept) 0.2957072
lcavol      0.2495335
lweight     0.2072931
new2        0.2183987
```

OLS is sensitive to multicollinearity

- Ridge regression is more robust

Number of predictors

- Recall the polynomial regression example
  - Sample size = 9
  - Perfectly fit when m = 8 (i.e. 9 parameters estimated)
  - What happen if m ≥ 9?



- Prostate example

```
set.seed(21098)
SmallIndex <- sample(1:nrow(Prostate), 6)
Small <- Prostate[SmallIndex, ]
```

- OLS

```
lm(formula = lpsa ~ lcavol + lweight + age + lbph + svi + lcp +
                   gleason + pgg45, data = Small)


Coefficients:
(Intercept)       lcavol      lweight          age         lbph          svi
   -5.67764      0.69457      0.70011      0.07270     -0.03293           NA
        lcp      gleason        pgg45
    0.15074           NA           NA
```

Number of predictors

- Ridge and LASSO

```
x.Small <- model.matrix(lpsa ~ lcavol + lweight + age + lbph + svi + lcp +
                          gleason + pgg45 - 1, data = Small)
y.Small <- Small$lpsa
Small.Ridge <- glmnet(x.Small, y.Small, alpha = 0, lambda = c(0.1, 0.01))
coef(Small.Ridge)
Small.LASSO <- glmnet(x.Small, y.Small, alpha = 1, lambda = c(0.1, 0.01))
coef(Small.LASSO)
```

Ridge

```
                      s0          s1
(Intercept) -3.983189980 -3.34701545
lcavol       0.634838300  0.71370283
lweight      0.377882039  0.39886193
age          0.052767193  0.04535218
lbph         0.043633242  0.03597348
svi          .            .
lcp          0.337259634  0.42433255
gleason      0.156275254  0.12605415
pgg45        0.005244347  0.00601659
```

LASSO

```
                     s0            s1
(Intercept) -4.61897188 -5.5170679541
lcavol       0.62848146  0.6821712429
lweight      0.45130177  0.5797324637
age          0.06858169  0.0746291622
lbph         .           .
svi          .           .
lcp          .           0.0129012020
gleason      .           .
pgg45        .           0.0005385419
```

Comparison

	OLS	Ridge	LASSO
Bias	Unbiased	Biased	Biased
Variance	Larger variance	Smaller variance	Smaller variance
Parsimony	Number of predictors cannot be reduced	Coefficients shrink but never reach 0, same number of predictors	Some coefficients shrink to 0, fewer predictors
Multicollinearity	Sensitive to strongly correlated predictors	Robust and good for grouped selection	Some coefficients shrink to 0, good for excluding correlated predictors
N < # of parameters	Not estimable	Estimable	Estimable

Elastic net

- Objective functions:
  - Ridge minimizes $\sum_{i=1}^{N}\left(y_i - \beta_0 - \boldsymbol{x}_i^T\boldsymbol{\beta}\right)^2 + \lambda\sum_{j=1}^{p}\beta_j^2$
  - LASSO minimizes $\sum_{i=1}^{N}\left(y_i - \beta_0 - \boldsymbol{x}_i^T\boldsymbol{\beta}\right)^2 + \lambda\sum_{j=1}^{p}\left|\beta_j\right|$
- Averaging the two penalty functions, weighted by a parameter α:
  - $\lambda\left[(1-\alpha)\sum_{j=1}^{p}\beta_j^2 + \alpha\sum_{j=1}^{p}\left|\beta_j\right|\right]$
  - When α = 0 → Ridge
  - When α = 1 → LASSO
  - When 0 < α < 1 → Elastic net
- Recall the R codes for glmnet(x, y, alpha…)
  - alpha: 0 = Ridge regression, 1 = LASSO regression, 0 < α < 1 = Elastic net

Elastic net

- Prostate example

```
x <- model.matrix(lpsa ~ . - 1, data = Training)
y <- Training$lpsa
x.test <- model.matrix(lpsa ~ . - 1, data = Testing)
```

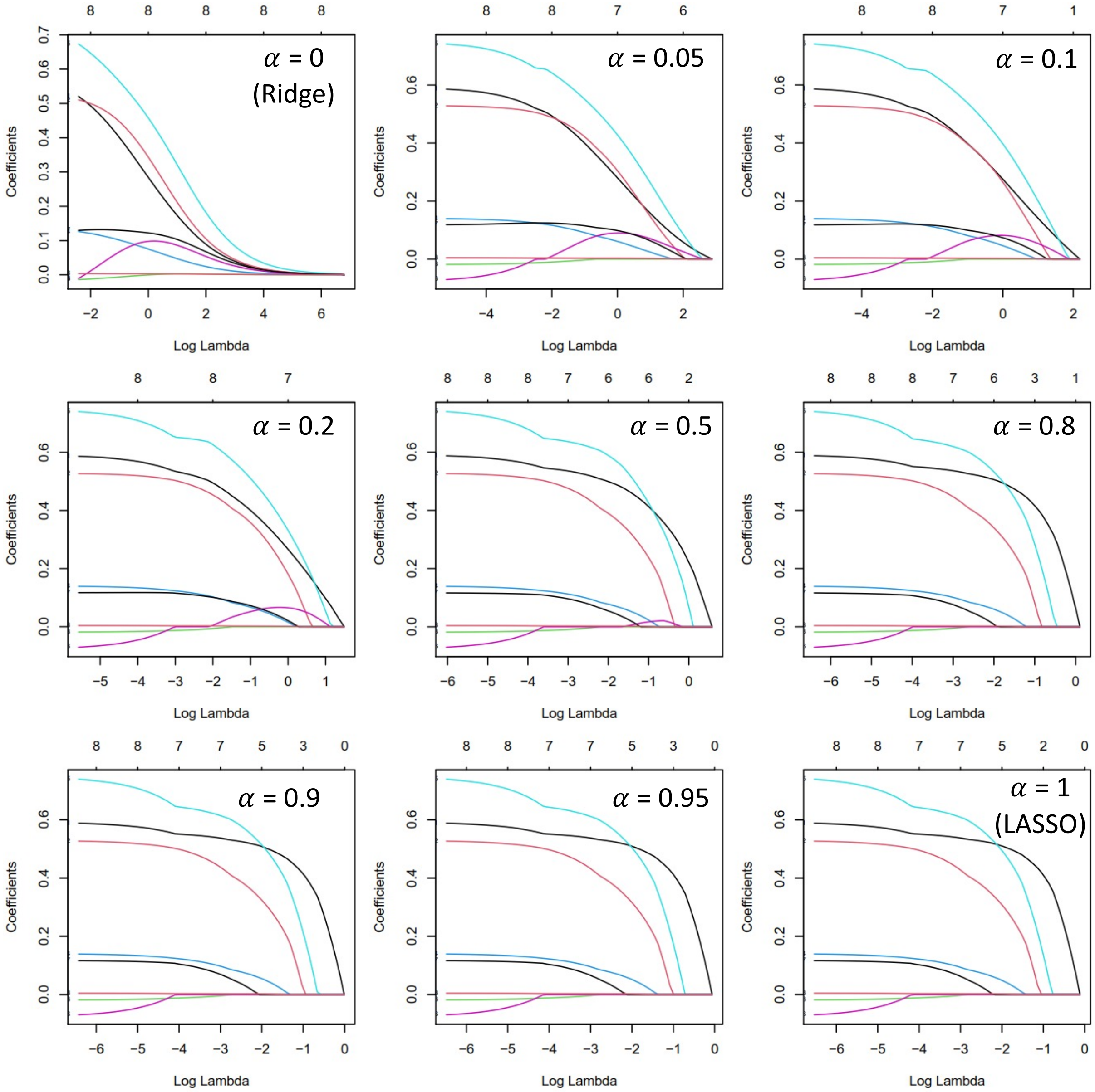
- Using glmnet with various alpha values, without specifying lambda

```
library(glmnet)
Elastic000 <- glmnet(x, y, alpha = 0)
Elastic005 <- glmnet(x, y, alpha = 0.05)
Elastic010 <- glmnet(x, y, alpha = 0.1)
Elastic020 <- glmnet(x, y, alpha = 0.2)
Elastic050 <- glmnet(x, y, alpha = 0.5)
Elastic080 <- glmnet(x, y, alpha = 0.8)
Elastic090 <- glmnet(x, y, alpha = 0.9)
Elastic095 <- glmnet(x, y, alpha = 0.95)
Elastic100 <- glmnet(x, y, alpha = 1)
```

- Plot the graphs of Coefficients again Log Lambda

```
plot(Elastic000, xvar="lambda", label=TRUE)
...
plot(Elastic100, xvar="lambda", label=TRUE)
```

Elastic net


α = 0
(Ridge)
8 8 8 8 8
0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7
−2 0 2 4 6
α = 0.05
8 8 7 6
−4 −2 0 2
α = 0.1
8 8 7 1
−4 −2 0 2
α = 0.2
8 8 7
−5 −4 −3 −2 −1 0 1
α = 0.5
8 8 8 7 6 6 2
−6 −5 −4 −3 −2 −1 0
α = 0.8
8 8 8 7 6 3 1
−6 −5 −4 −3 −2 −1 0
α = 0.9
8 8 7 7 5 3 0
−6 −5 −4 −3 −2 −1 0
α = 0.95
8 8 7 7 5 3 0
−6 −5 −4 −3 −2 −1 0
α = 1
(LASSO)
8 8 7 7 5 2 0
−6 −5 −4 −3 −2 −1 0
0.0 0.2 0.4 0.6
Coefficients
Log Lambda

Elastic net

- Cross-validation to determine λ

```
set.seed(76543)
Elastic.cv <- cv.glmnet(x, y, alpha = 0.5)
plot(Elastic.cv)
```

- Minimum MSE

```
min(Elastic.cv$cvm)
[1] 0.6108528
log(Elastic.cv$lambda.min)
[1] -2.956173
```

- 1-SE rule

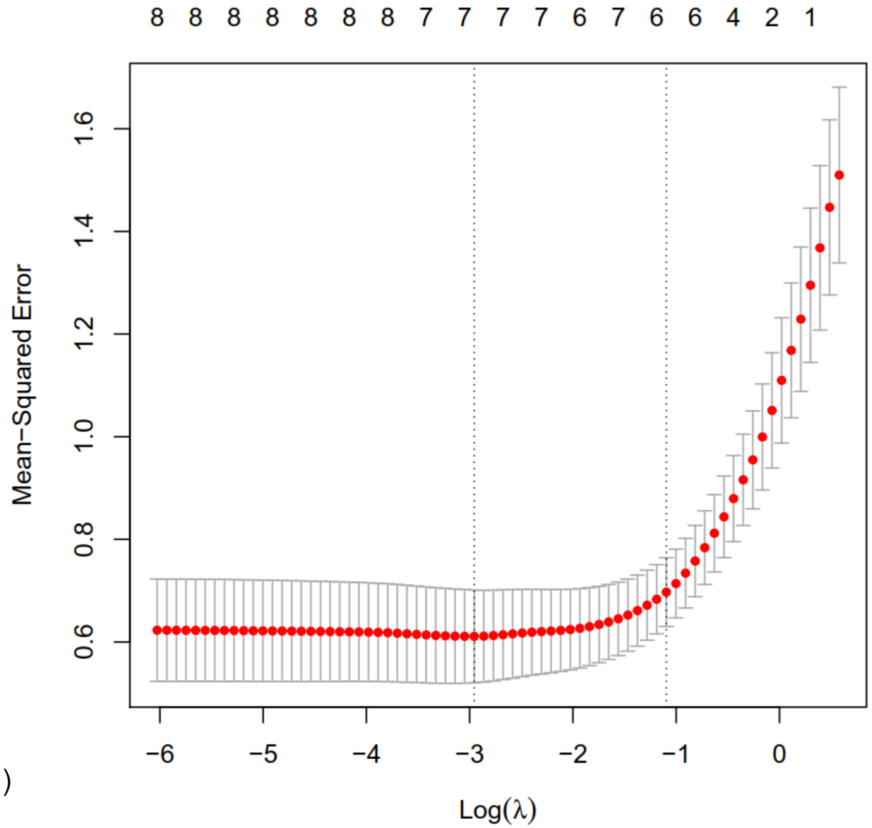
```
log(Elastic.cv$lambda.1se)
[1] -1.095499
```

How many non-zero coefficients?

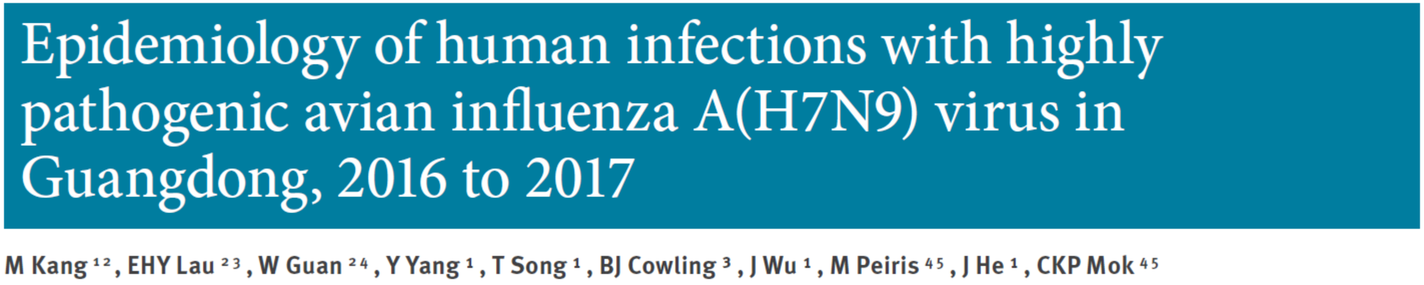
```
coef(Elastic050, s = Elastic.cv$lambda.1se)
```

Export the predicted values for further analysis

```
Elastic.Pred <- predict(Elastic050, newx = x.test,
                  s = Elastic.cv$lambda.1se)
```



Application



Epidemiology of human infections with highly pathogenic avian influenza A(H7N9) virus in Guangdong, 2016 to 2017

M Kang[1,2], EHY Lau[2,3], W Guan[2,4], Y Yang[1], T Song[1], BJ Cowling[3], J Wu[1], M Peiris[4,5], J He[1], CKP Mok[4,5]

Euro Surveill. 2017;22(27):pii=30568

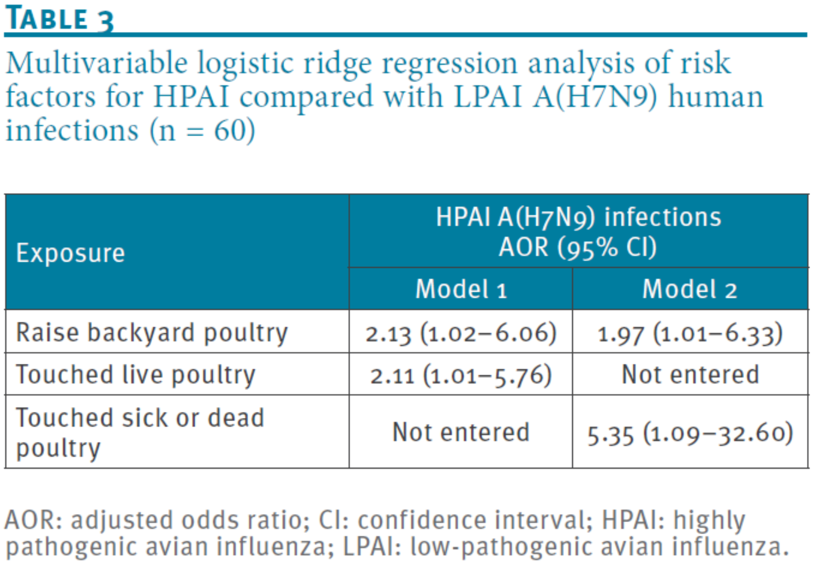


TABLE 3

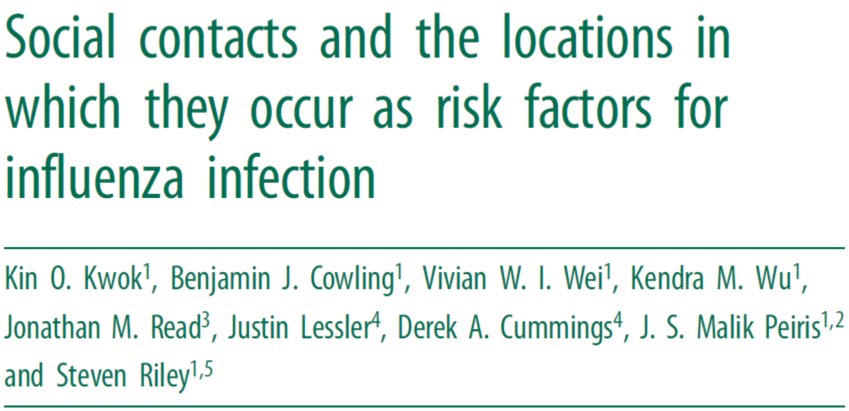
Multivariable logistic ridge regression analysis of risk factors for HPAI compared with LPAI A(H7N9) human infections (n = 60)

Exposure	HPAI A(H7N9) infections AOR (95% CI)	
	Model 1	Model 2
Raise backyard poultry	2.13 (1.02–6.06)	1.97 (1.01–6.33)
Touched live poultry	2.11 (1.01–5.76)	Not entered
Touched sick or dead poultry	Not entered	5.35 (1.09–32.60)

AOR: adjusted odds ratio; CI: confidence interval; HPAI: highly pathogenic avian influenza; LPAI: low-pathogenic avian influenza.

- Sample size = 60 (9 HPAI vs 51 LPAI)
- Low power for OLS
- Not estimable if more than 9 predictors
- Ridge regression was used
- Another R package “ridge”

Application



Social contacts and the locations in which they occur as risk factors for influenza infection

Kin O. Kwok[1], Benjamin J. Cowling[1], Vivian W. I. Wei[1], Kendra M. Wu[1], Jonathan M. Read[3], Justin Lessler[4], Derek A. Cummings[4], J. S. Malik Peiris[1,2] and Steven Riley[1,5]

Proc. R. Soc. B 2014; 281: 20140709.

- Used LASSO as sensitivity analysis
- To confirm the results from OLS where predictors were correlated
- Fixed λ = 4
- 95% CI constructed by bootstrap

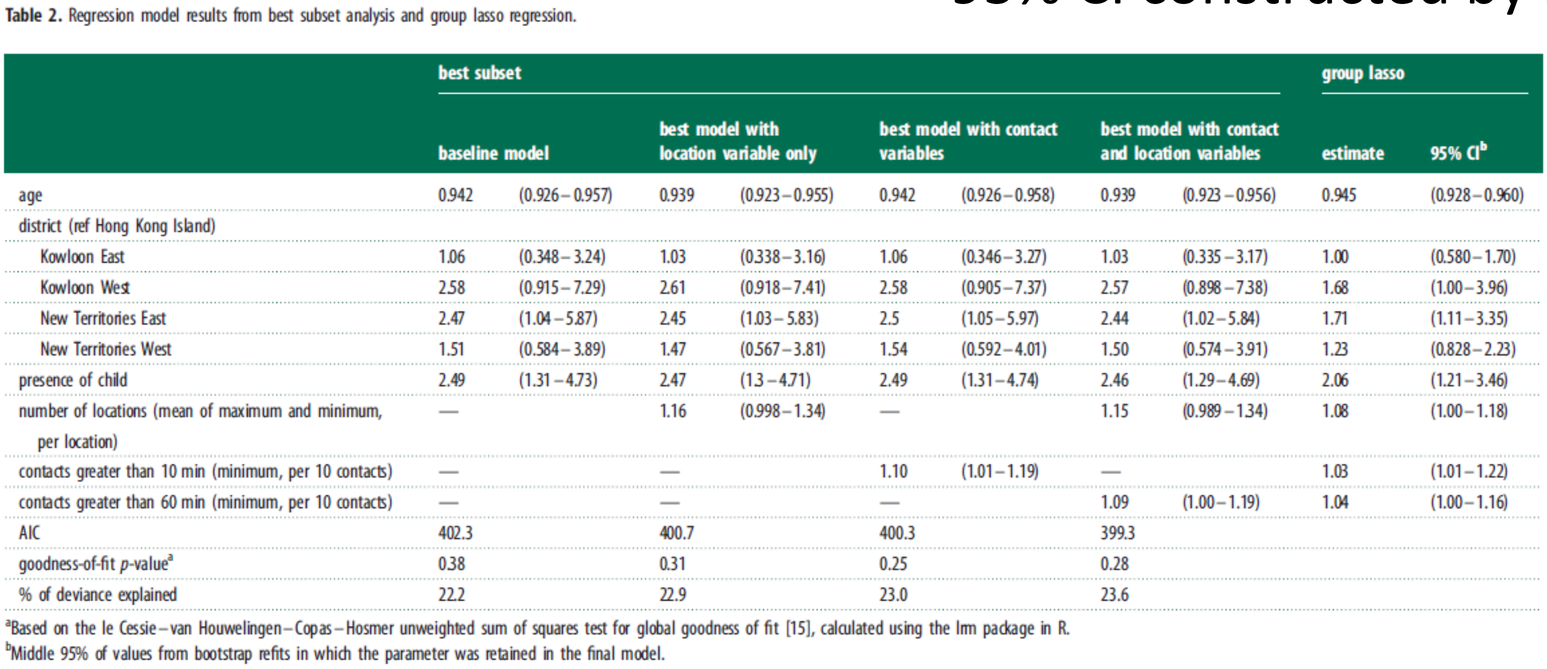


Table 2. Regression model results from best subset analysis and group lasso regression.

	best subset								group lasso	
	baseline model		best model with location variable only		best model with contact variables		best model with contact and location variables		estimate	95% CI[b]
age	0.942	(0.926 – 0.957)	0.939	(0.923 – 0.955)	0.942	(0.926 – 0.958)	0.939	(0.923 – 0.956)	0.945	(0.928 – 0.960)
district (ref Hong Kong Island)										
Kowloon East	1.06	(0.348 – 3.24)	1.03	(0.338 – 3.16)	1.06	(0.346 – 3.27)	1.03	(0.335 – 3.17)	1.00	(0.580 – 1.70)
Kowloon West	2.58	(0.915 – 7.29)	2.61	(0.918 – 7.41)	2.58	(0.905 – 7.37)	2.57	(0.898 – 7.38)	1.68	(1.00 – 3.96)
New Territories East	2.47	(1.04 – 5.87)	2.45	(1.03 – 5.83)	2.5	(1.05 – 5.97)	2.44	(1.02 – 5.84)	1.71	(1.11 – 3.35)
New Territories West	1.51	(0.584 – 3.89)	1.47	(0.567 – 3.81)	1.54	(0.592 – 4.01)	1.50	(0.574 – 3.91)	1.23	(0.828 – 2.23)
presence of child	2.49	(1.31 – 4.73)	2.47	(1.3 – 4.71)	2.49	(1.31 – 4.74)	2.46	(1.29 – 4.69)	2.06	(1.21 – 3.46)
number of locations (mean of maximum and minimum, per location)	—		1.16	(0.998 – 1.34)	—		1.15	(0.989 – 1.34)	1.08	(1.00 – 1.18)
contacts greater than 10 min (minimum, per 10 contacts)	—		—		1.10	(1.01 – 1.19)	—		1.03	(1.01 – 1.22)
contacts greater than 60 min (minimum, per 10 contacts)	—		—		—		1.09	(1.00 – 1.19)	1.04	(1.00 – 1.16)
AIC	402.3		400.7		400.3		399.3			
goodness-of-fit *p*-value[a]	0.38		0.31		0.25		0.28			
% of deviance explained	22.2		22.9		23.0		23.6			

[a] Based on the le Cessie – van Houwelingen – Copas – Hosmer unweighted sum of squares test for global goodness of fit [15], calculated using the lrm package in R.

[b] Middle 95% of values from bootstrap refits in which the parameter was retained in the final model.