QuantLab
주식회사 퀀트랩

Sentiment Analysis

감정사전 & 감정점수 만들기

김형준
Analytic Director / (주) 퀀트랩 / kim@mindscale.kr

퀀트랩 소개

퀀트랩 소개

- 2011년 설립
- 데이터 분석, 직무역량평가, 전문성 개발 전문 컨설팅 기업

members

유재명
서울대학교 산업공학과
서울대학교 인지과학 박사(수료)

황창주
서울대학교 심리학과
서울대학교 심리학 박사(수료)

김형준
서울대학교 인류학과 / 심리학과
서울대학교 인지과학 석사

clients

- LG생활건강
- LG U+
- NC소프트
- SK플래닛
- 교통안전공단
- 삼성전자
- 이지웰페어
- 웅진씽크빅
- 중소기업진흥공단
- 한화
- 현대자동차

4

워크숍 관련 온라인 사이트

http://course.mindscale.kr/course/text-analysis

Facebook 아이디에 연결되어 있습니다 연결 해제

코스

현재 수강 중인 코스입니다.

제목
텍스트에서 여론과 감정을 발견하기 : R을 이용한 텍스트 데이터 분석 (05/30)
텍스트에서 여론과 감정을 발견하기 : R을 이용한 텍스트 데이터 분석
토픽 분석
R을 이용한 웹 크롤링

오늘의 목표

- 감정 사전 만들기
- 감정 점수 만들기

- 상관관계 이해하기
- 회귀분석 이해하기
- 모형평가 이해하기

왜 감정분석을 하는가?

설문지의 단점

1) 조사 비용 발생

2) 미리 정해진 문항만 측정 가능

3) 사회적 바람직성 등 편향 발생

감정분석

텍스트에서 감정 단어를 추출하여 점수화

1) 기계 학습 (Machine Learning)

2) 단어 사전 기반

사전 기반 분석

장점

- 사용하기 간편

단점

- 주제에 따라 사전이 달라 짐

- 동음이의어 처리 힘듦 e.g) bank

기계학습 기반 분석

장점

- 높은 정확도

단점

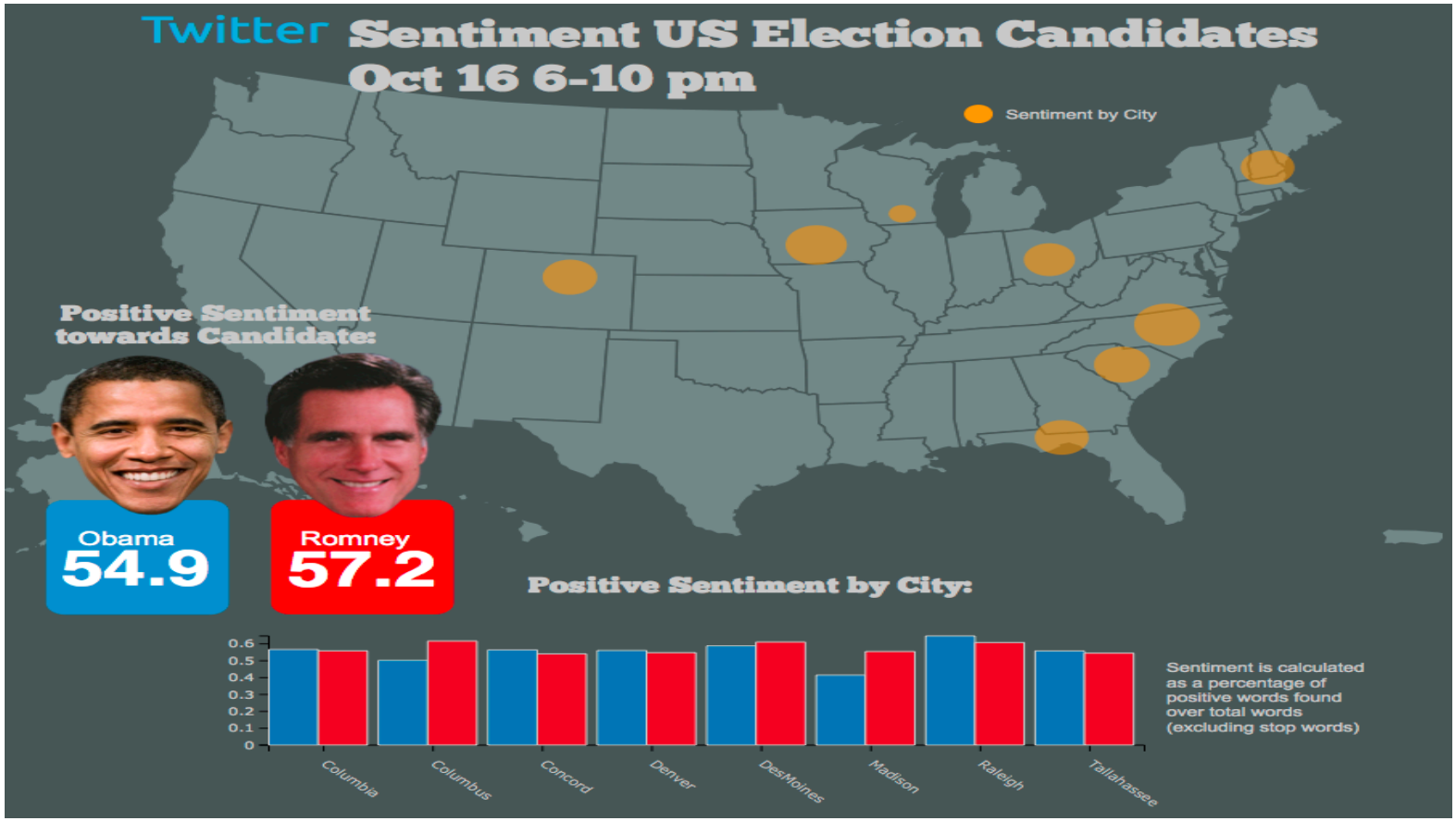
- Over-fitting 해결

- 많은 데이터 필요

예) 나이브 베이즈 / 최대 엔트로피 / 서포트벡터머신 /

랜덤 포레스트 / 토픽 모델

감정 분석 예시


Twitter Sentiment US Election Candidates
Oct 16 6-10 pm
Sentiment by City
Positive Sentiment towards Candidate:
Obama 54.9
Romney 57.2
Positive Sentiment by City:
0.6
0.5
0.4
0.3
0.2
0.1
0
Columbia
Columbus
Concord
Denver
DesMoines
Madison
Raleigh
Tallahassee
Sentiment is calculated as a percentage of positive words found over total words (excluding stop words)

감정 분석 예시

사전 지식

감정분석: 문장에 사용된 단어로 감정을 예측

예: "이 영화는 좀 길지만 재미있고 신난다"

- 길다 -> 부정

- 재미있다 -> 긍정

- 신나다 -> 긍정

예측 분석

예측분석

선형회귀분석

SVM

RandomForest

Deep Learning

회귀분석(선형(직선) 모형)

예시

- 키가 1cm 증가할 때마다 몸무게가 1kg 증가
- 월 소득이 100만원 증가할 때마다 몸무게가 1kg 감소
- 부정단어가 1개 증가할 때 마다 평점 .1점 감점
- 긍정단어가 1개 증가할 때 마다 평점 .1점 증가

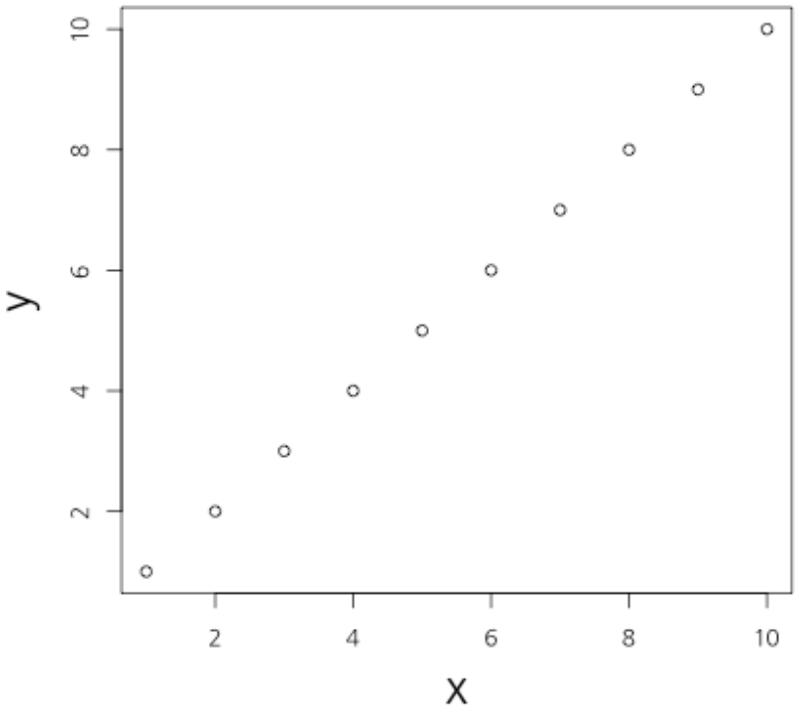
회귀분석의 문제

- 변수가 많아지면 과적합(overfitting)이 발생
- 회귀계수가 극단적으로 커지거나 작아짐
- 예측력이 떨어짐
- 과적합을 막아주는 방법이 필요

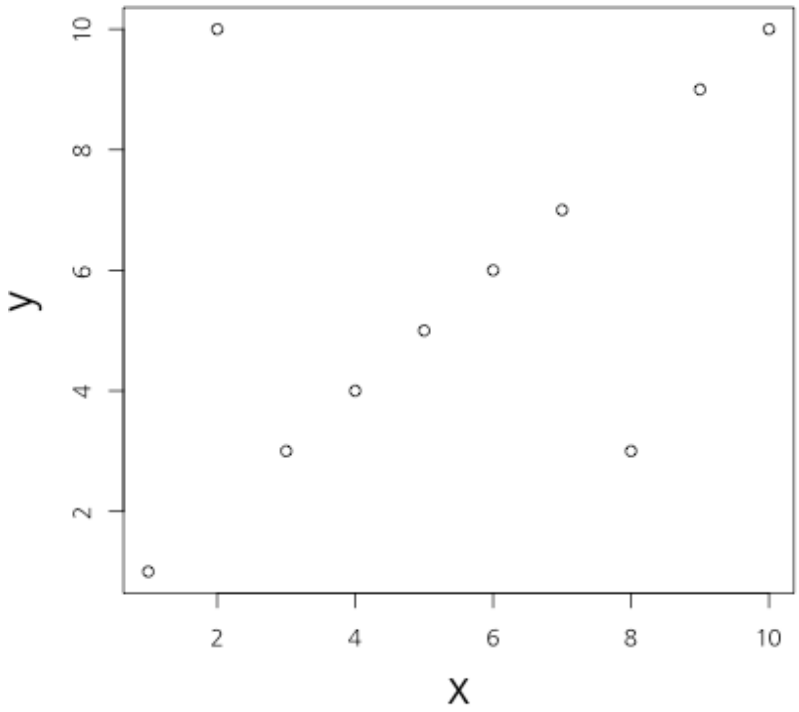
과적합을 막는 법

- 라쏘(lasso): 작은 회귀계수를 0으로 만듦
- 릿지(ridge): 전반적으로 회귀계수를 줄여줌
- 엘라스틱넷(elastic net): 라쏘 + 릿지
- 감정분석에서 라쏘를 쓰면 감정 단어만 추출됨

상관관계




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




```
## [1] 0.4885042
```

상관관계

x가 증가(혹은 감소)할때 y가 선형적으로 증가(혹은 감소)하는 정도

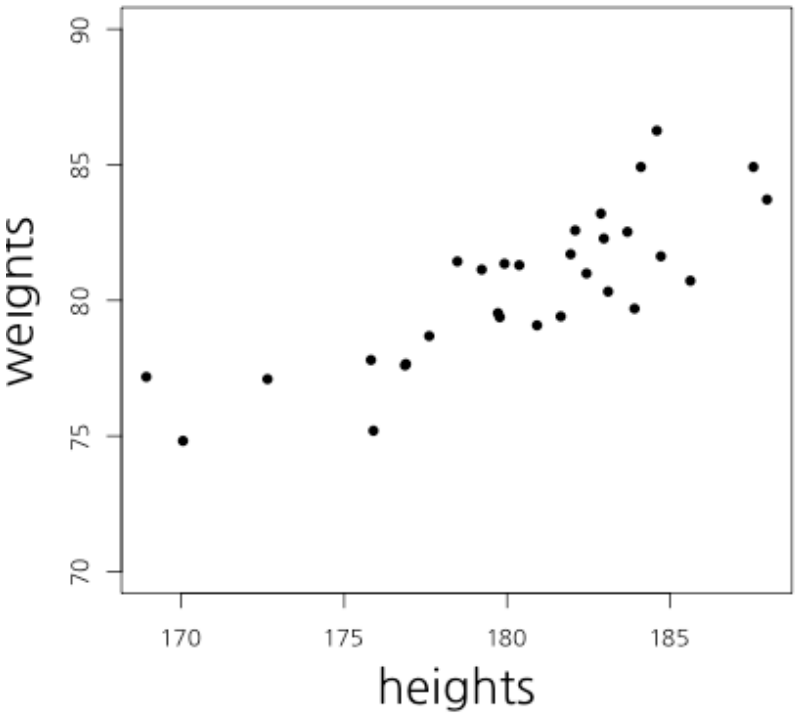
scale

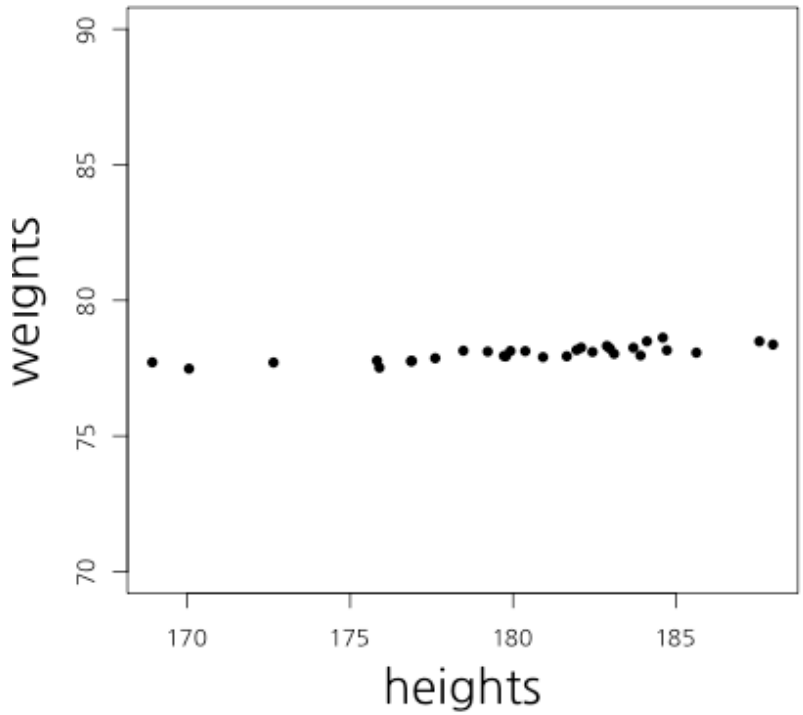
키가 만약 cm라면, 키가 1cm 증가하면 몸무게는 1kg증가

키가 만약 mm라면, 키가 1mm 증가하면 몸무게는 0.1kg 증가

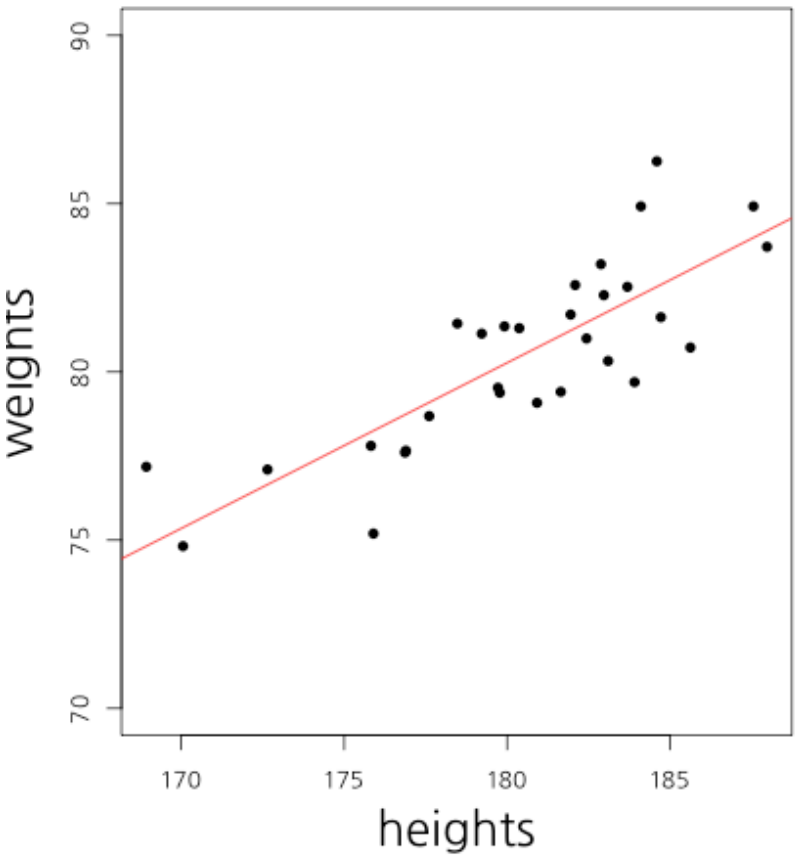
-> 표준화해야 한다

둘 중 무엇이 상관이 더 클까요?





상관관계 및 회귀분석

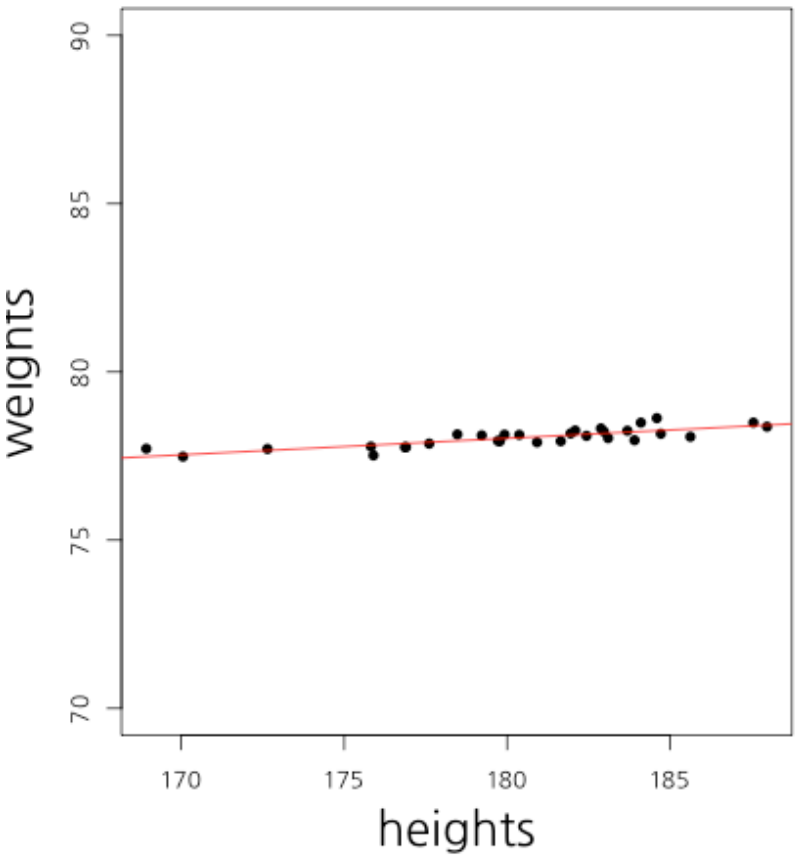


	ESTIMATE	STD. ERROR	T VALUE	PR(>|T|)
(Intercept)	-8.29	11.74	-0.71	0.49
heights	0.49	0.07	7.56	0.00

```
cor(weights, heights)
```

```
## [1] 0.8194181
```

상관관계 및 회귀분석

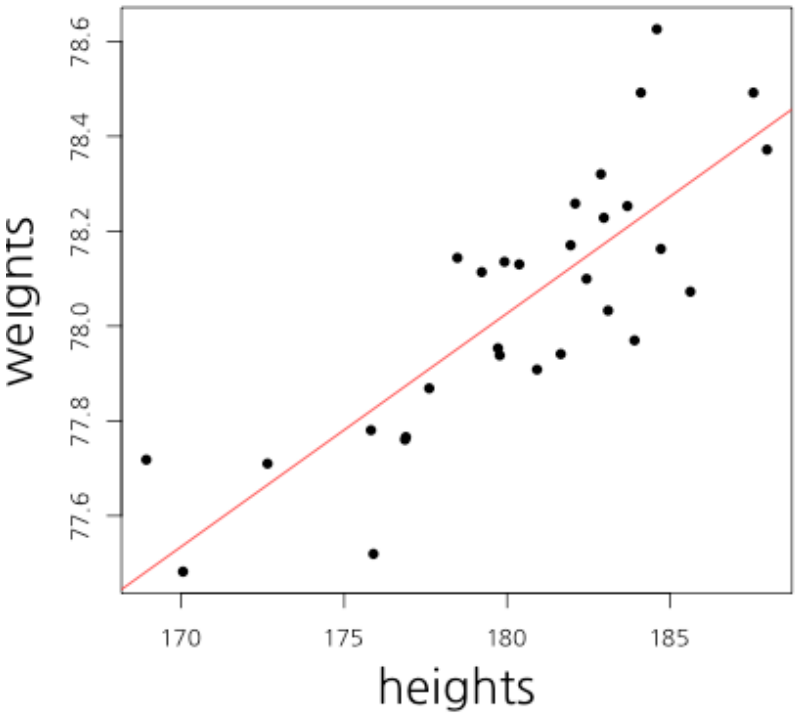


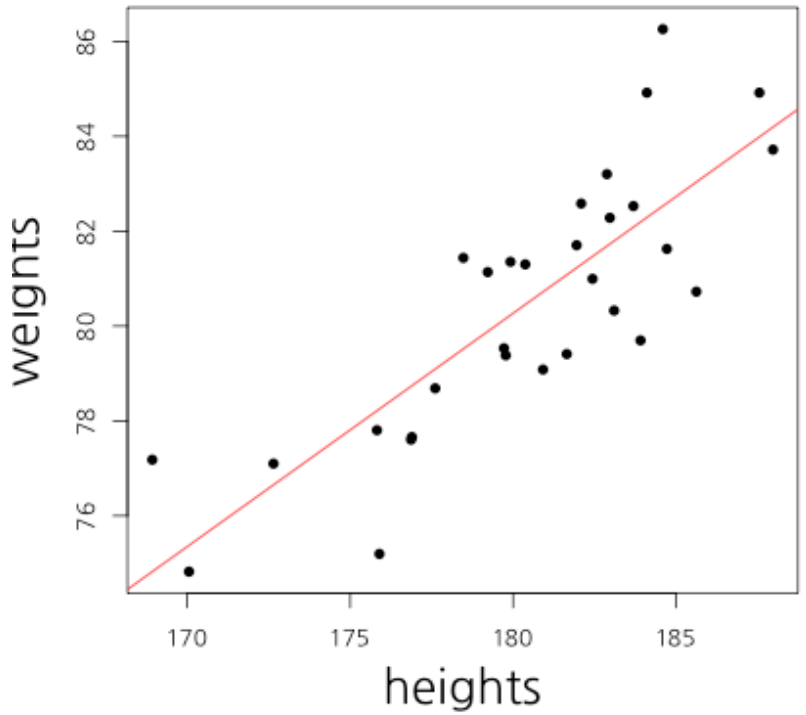
	ESTIMATE	STD. ERROR	T VALUE	PR(>|T|)
(Intercept)	69.17	1.17	58.93	0.00
heights	0.05	0.01	7.56	0.00

```
cor(weights, heights)
```

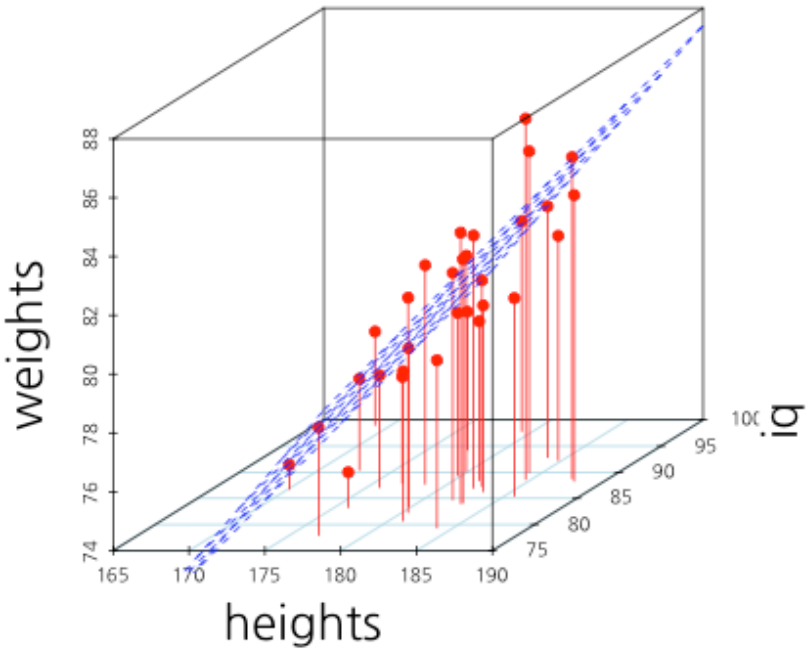
```
## [1] 0.8194181
```

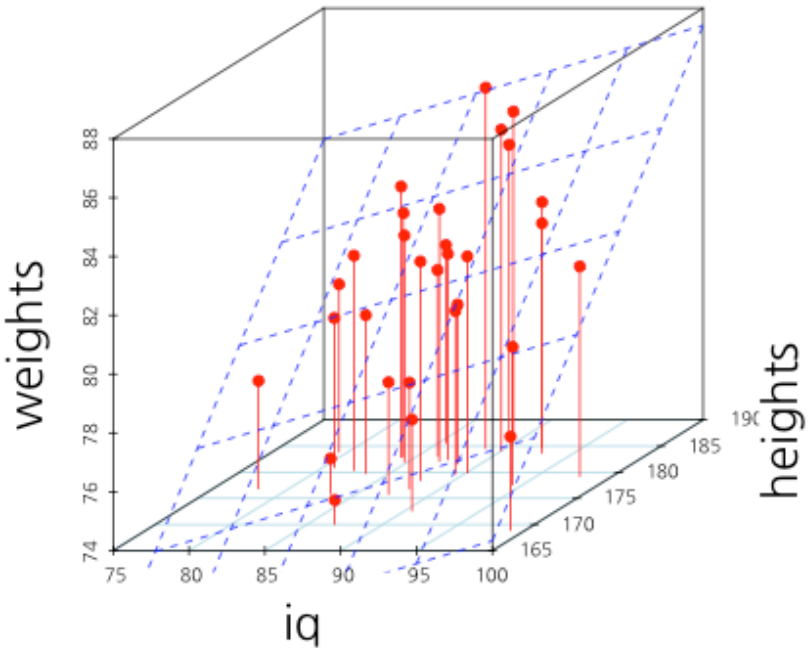
상관관계 및 회귀분석


weights
77.6
77.8
78.0
78.2
78.4
78.6
170
175
180
185
heights


weights
76
78
80
82
84
86
170
175
180
185
heights

X가 2개라면?






```
## [1] 0.8194181
```

```
## [1] 0.09818667
```

다중회귀분석

	ESTIMATE	STD. ERROR	T VALUE	PR(>|T|)
(Intercept)	-27.49	12.81	-2.15	0.04
iq	0.15	0.06	2.68	0.01
heights	0.52	0.06	8.72	0.00

예측력

MSE(Mean of Square Error)

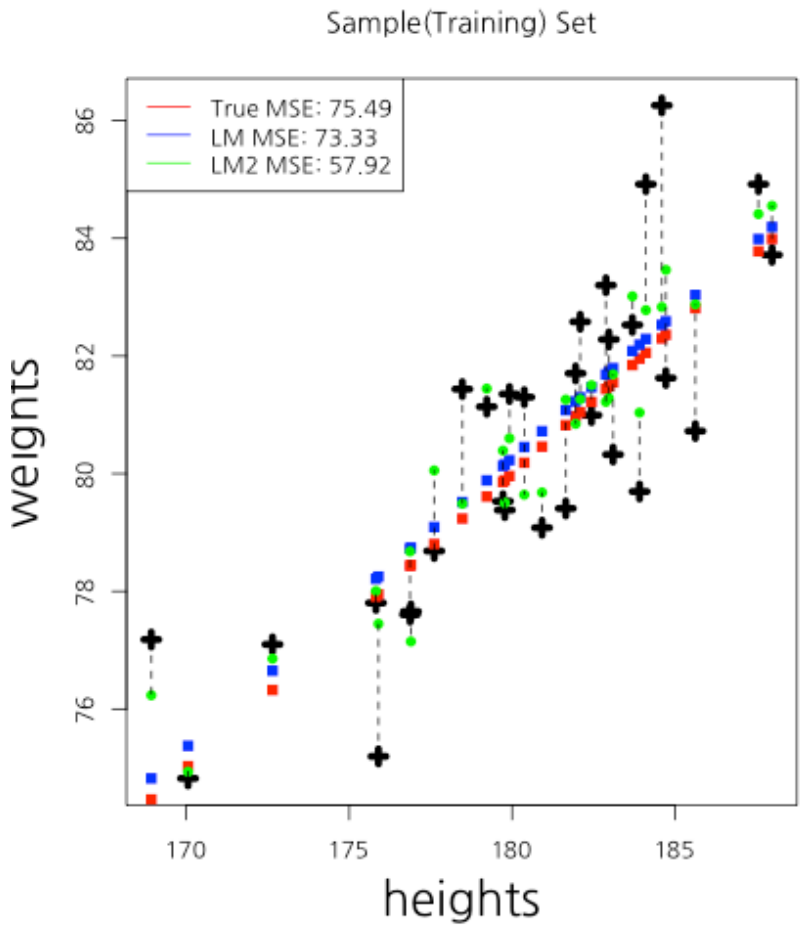
$$MSE = \sum_{i=1}^{n} (Y_i - \hat{Y}_i)^2$$

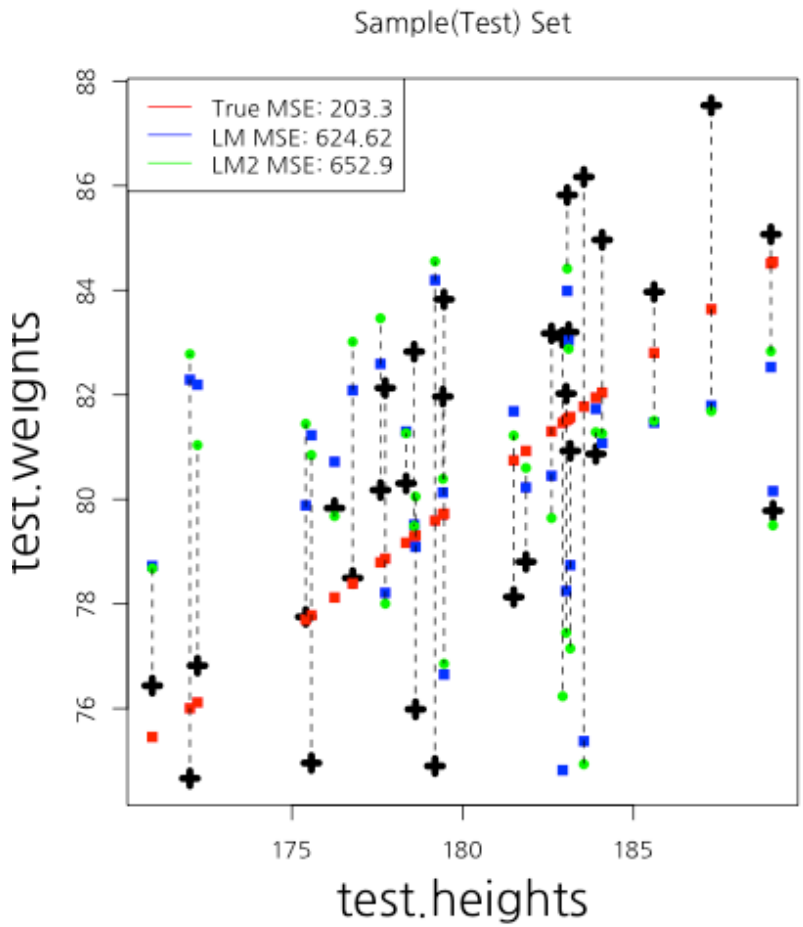
정확도(Accracy)

		실제 라벨	
		긍정 문서	부정 문서
모형이 예측한 라벨	긍정 문서	True Positive	False Positive
	부정 문서	False Negatives	True Negatives

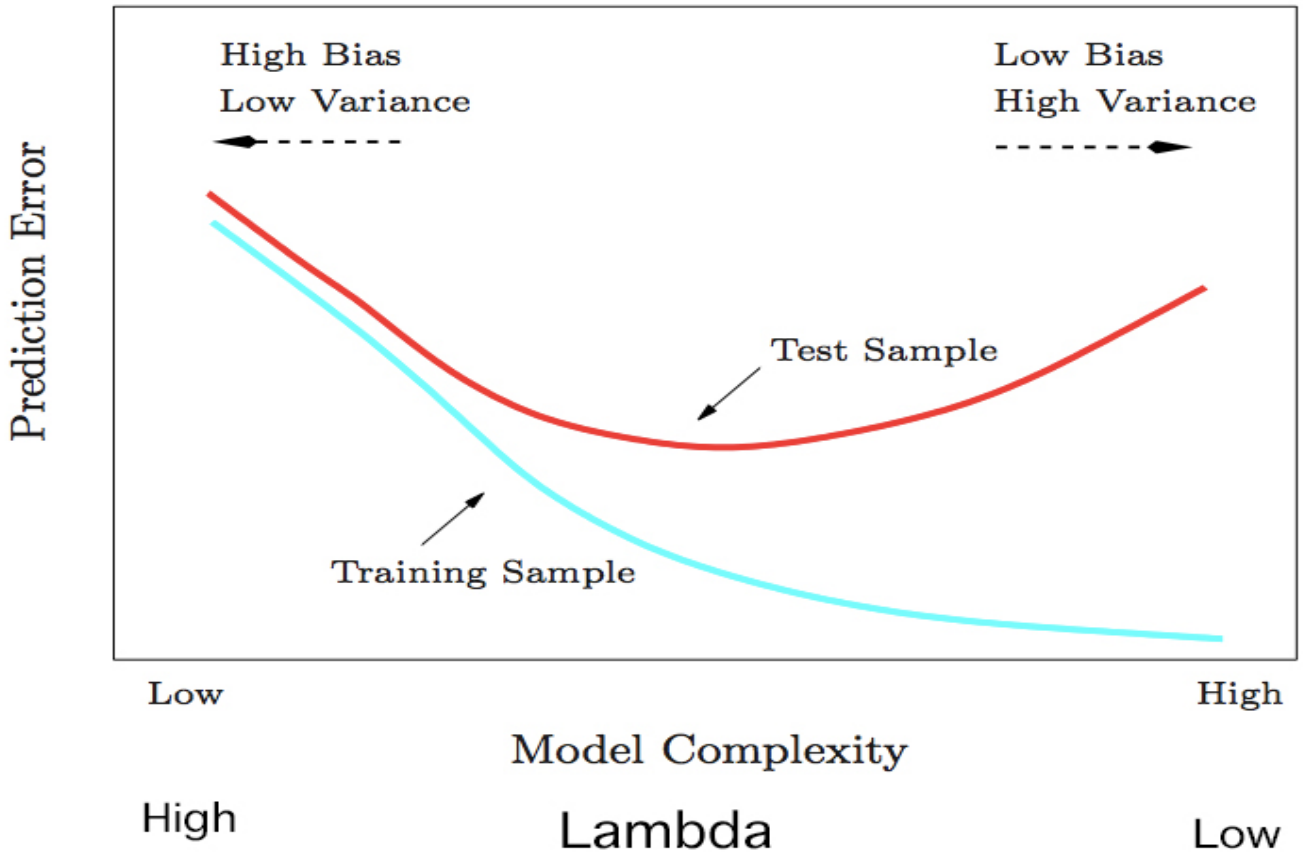
정확도 = (TP + TN) / (TP + FP + TN + FN)

Traninig Vs Test





Over-fitting



Over-fitting(과적합)

How to avoid Over-fitting

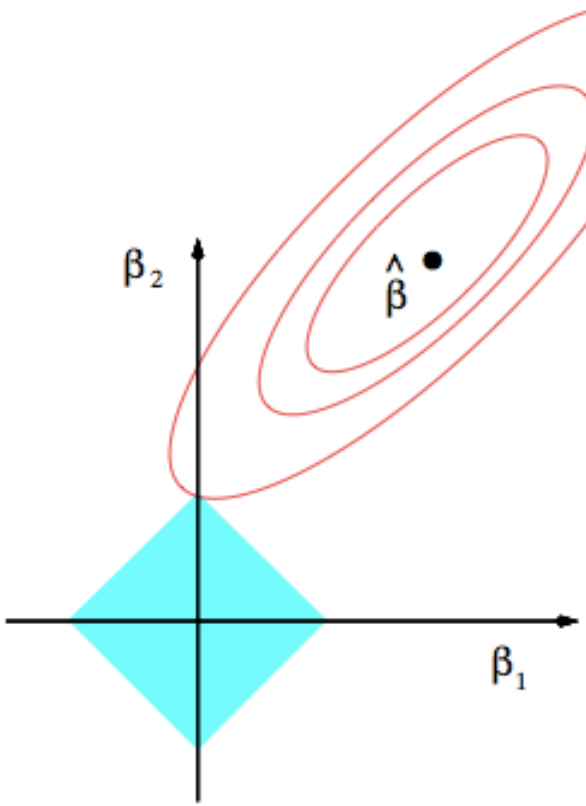
- Penality of Model Complexity (MSE 보정)

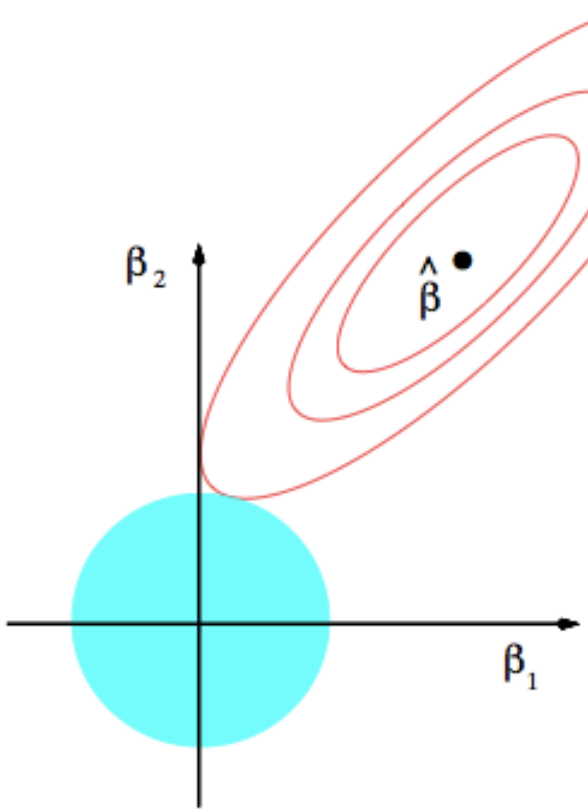
- Regulization (Lasso, Ridge, Elastic Net)

- Bayesian

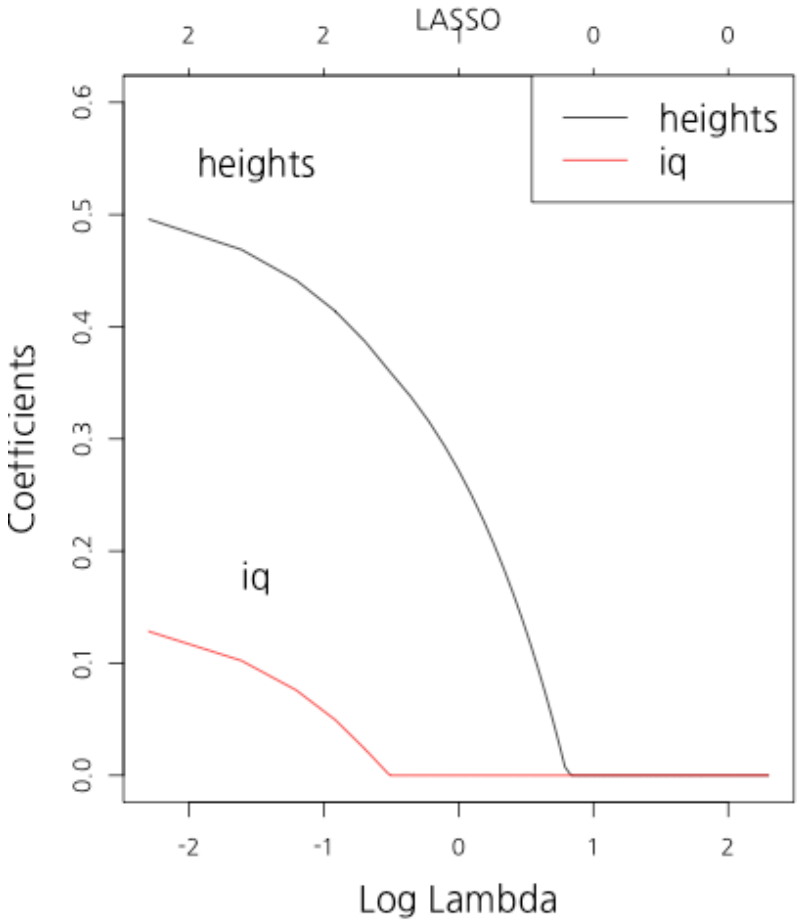
- Drop Out, Bagging, Feature Bagging

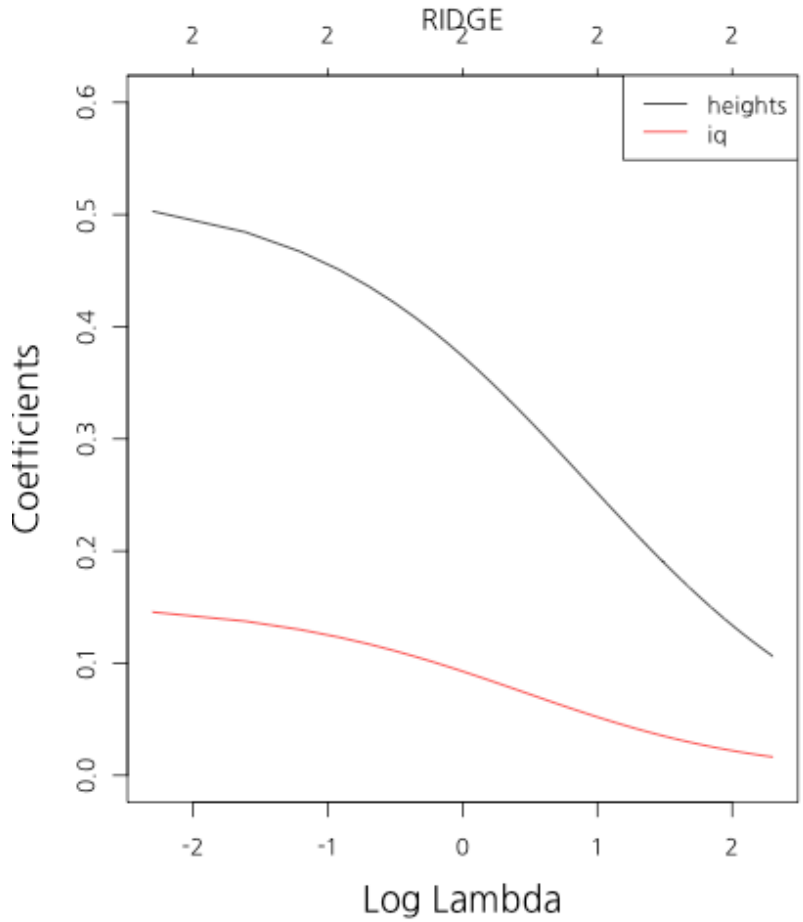
Lasso Vs Ridge





Lasso Vs Ridge





감정분석

Data

25,000 IMDB movie reviews 중에서 1,000개만

Training Vs Test = 7 Vs 3

Traing Set 과 Test Set 분할

```
fileName <- "data/IMDBmovie/labeledTrainData.tsv"
data <- read.csv(fileName, header=T, sep="\t", quote="")
nrow(data)
```

```
## [1] 25000
```

```
data <- data[1:2000, ]
```

Traing Set 과 Test Set 분할

```
totalNum <- 1:nrow(data)
set.seed(12345)
shuffledNum <- sample(totalNum, nrow(data), replace = F)
trainingNum <- shuffledNum[1:1400]
testNum <- shuffledNum[1401:2000]
data.train <- data[trainingNum, ]
data.test <- data[testNum, ]
```

Term-DocumentMatrix

```
library(tm)
```

```
corpus <- Corpus(VectorSource(data.train$review))
tdm.train <- TermDocumentMatrix(corpus,
                                control=list(tolower = T,
                                             removePunctuation = T,
                                             removeNumbers = T,
                                             stopwords=stopwords("SMART")))

dim(tdm.train)
```

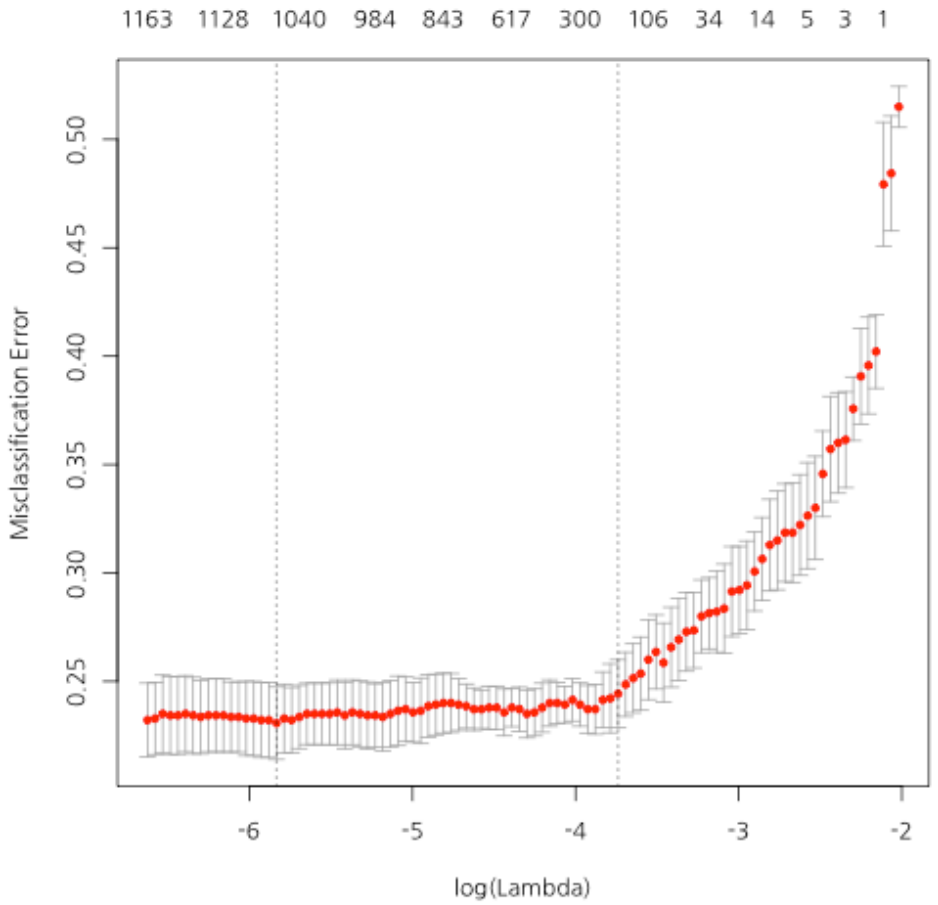
```
## [1] 24865  1400
```

LASSO Regression

```
alpha <- 1
cv.lasso <- cv.glmnet(as.matrix(t(tdm.train)), data.train$sentiment,
                      type.measure = "class",
                      nfolds = 4,
                      family = "binomial",
                      alpha = alpha)
```

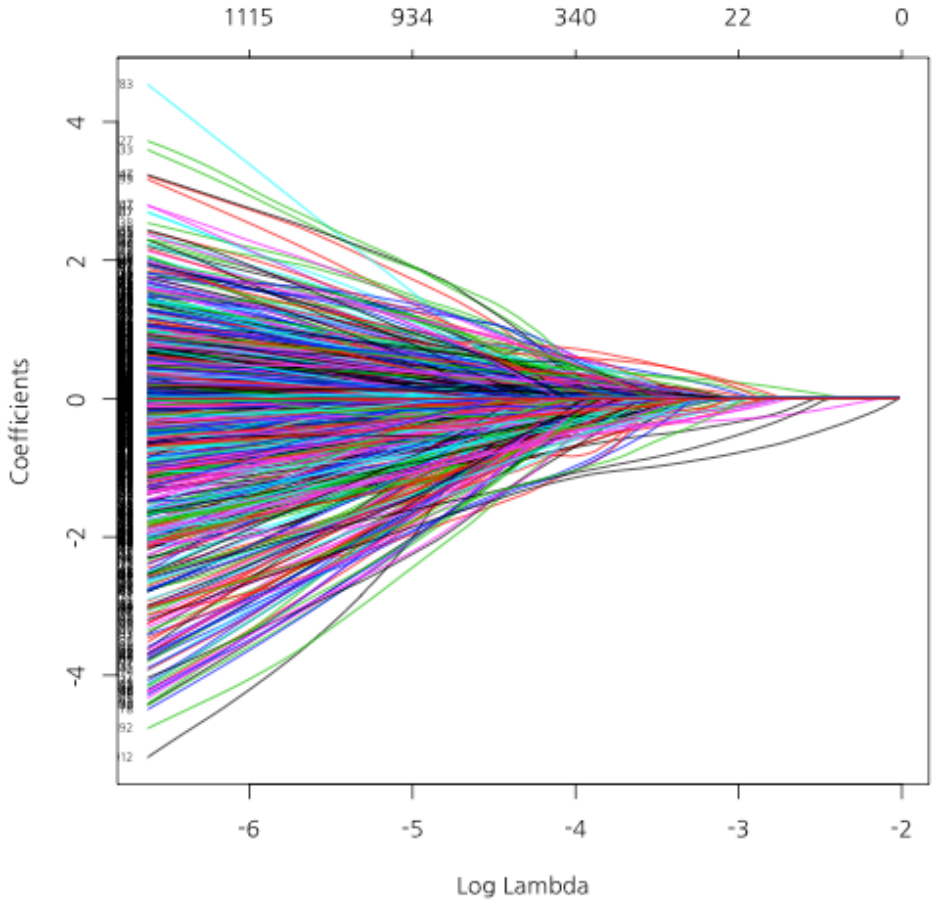
LASSO Regression

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



LASSO Regression

```
plot(cv.lasso$glmnet.fit, "lambda", label=TRUE)
```

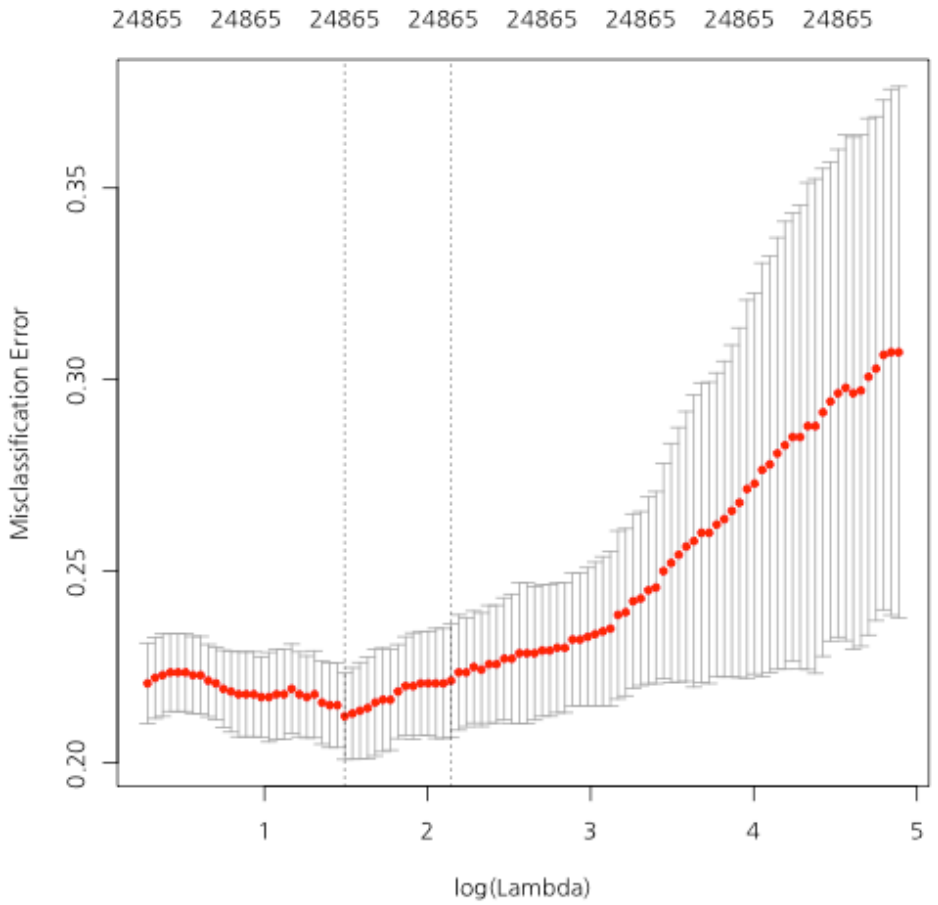


Ridge Regression

```
alpha <- 0
cv.ridge <- cv.glmnet(as.matrix(t(tdm.train)), data.train$sentiment,
                      type.measure = "class",
                      nfolds = 4,
                      family = "binomial",
                      alpha = alpha)
```

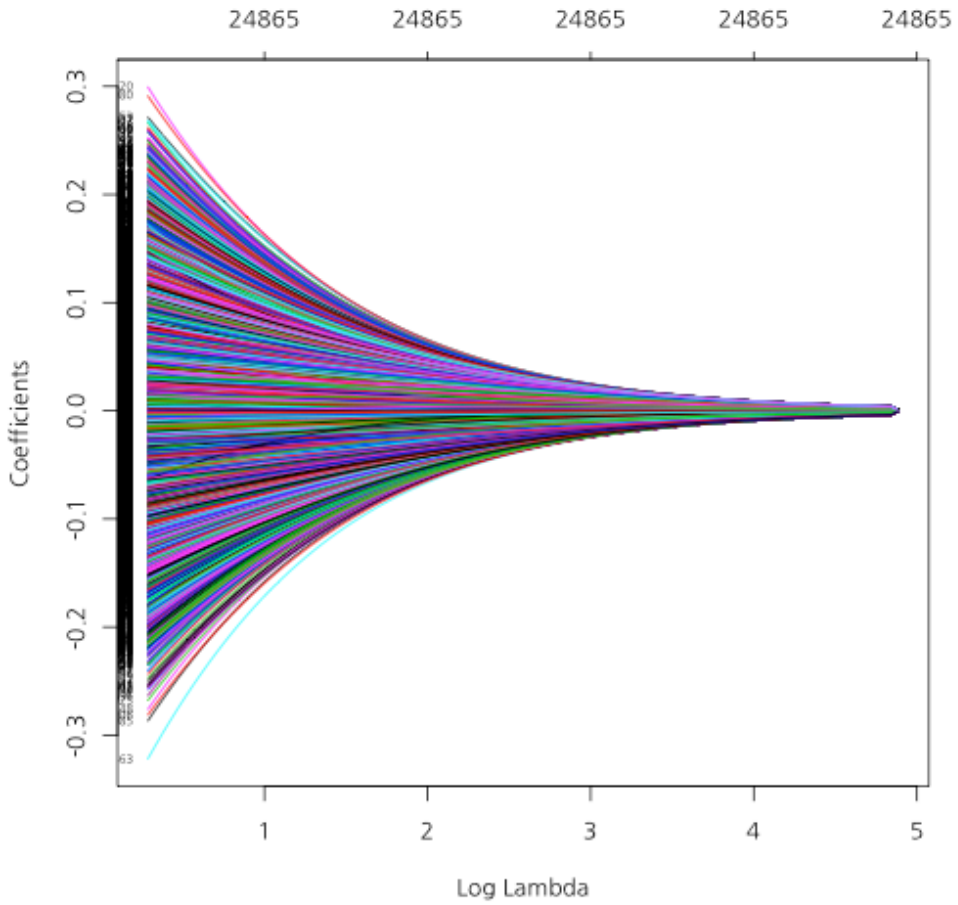
RIDGE Regression

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



RIDGE Regression

```
plot(cv.ridge$glmnet.fit, "lambda", label=TRUE)
```

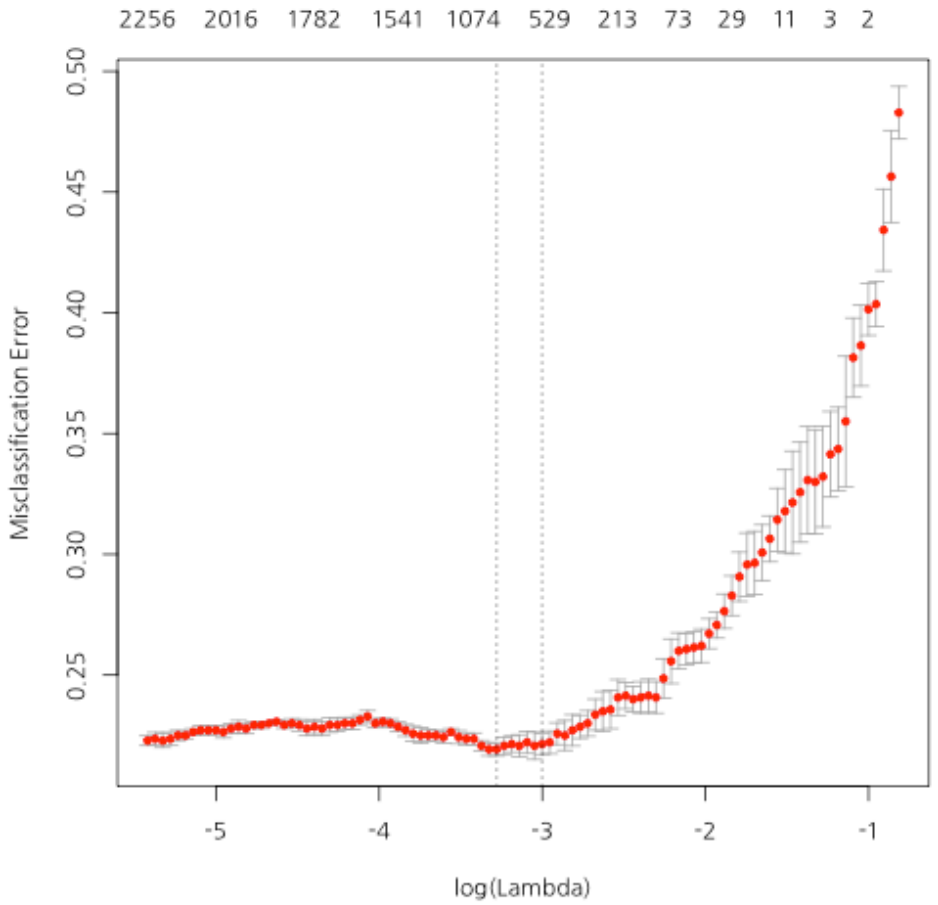


ElasticNet Regression

```
alpha <- .3
cv.elastic <- cv.glmnet(as.matrix(t(tdm.train)), data.train$sentiment,
                        type.measure = "class",
                        nfolds = 4,
                        family = "binomial",
                        alpha = alpha)
```

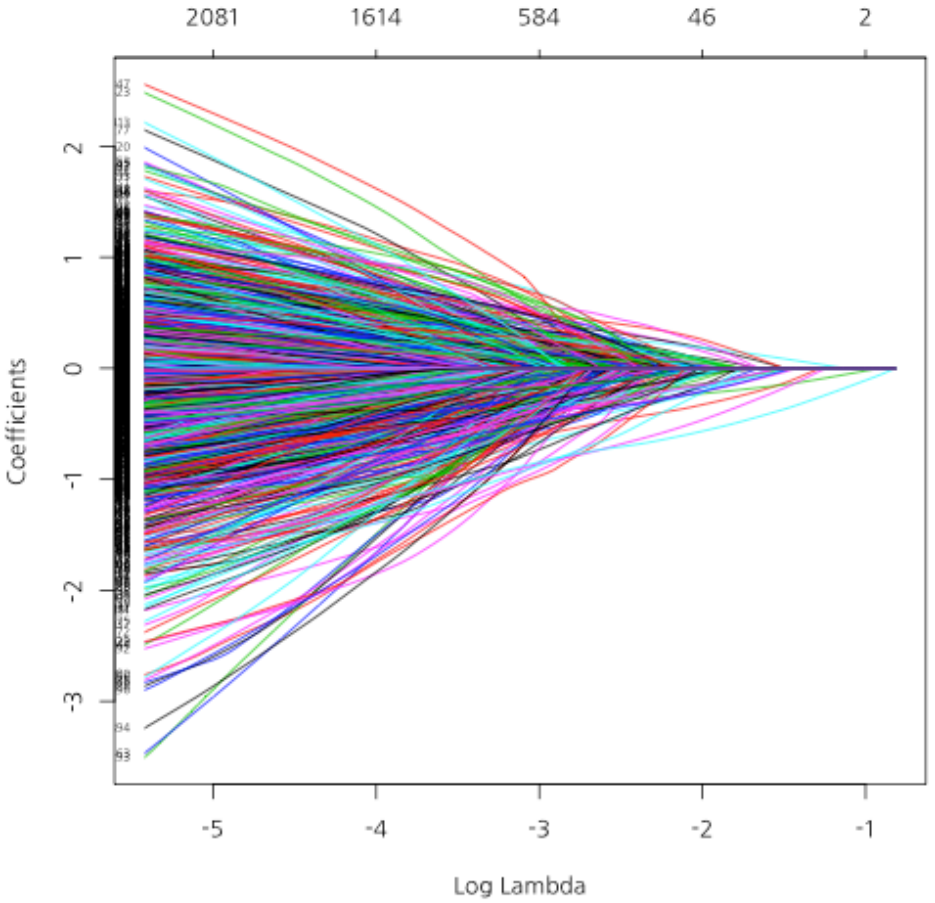
ElasticNet Regression

```
plot(cv.elastic)
```



ElasticNet Regression

```
plot(cv.elastic$glmnet.fit, "lambda", label=TRUE)
```



감정 단어 추출

```
coef.lasso <- coef(cv.lasso, s = "lambda.min")[,1]
coef.ridge <- coef(cv.ridge, s = "lambda.min")[,1]
coef.elastic <- coef(cv.elastic, s = "lambda.min")[,1]
```

감정 단어 추출

```
pos.lasso <- sort(coef.lasso[coef.lasso > 0])
neg.lasso <- sort(coef.lasso[coef.lasso < 0])

length(pos.lasso)
```

```
## [1] 475
```

```
length(neg.lasso)
```

```
## [1] 606
```

감정 단어 추출

```
pos.lasso[1:5]
```

```
##  patricyoure        tonks       ahmads   unorthodox         lori
## 2.331122e-15 5.989465e-15 6.367617e-13 1.169503e-05 1.214698e-05
```

```
neg.lasso[1:5]
```

```
## differentokay           pic distinguished    decoration        inform
##     -3.918694     -3.835755     -3.390117     -3.359009     -3.290011
```

감정 단어 추출

```
pos.ridge <- sort(coef.ridge[coef.ridge > 0])
neg.ridge <- sort(coef.ridge[coef.ridge < 0])

length(pos.ridge)
```

```
## [1] 12963
```

```
length(neg.ridge)
```

```
## [1] 11903
```

감정 단어 추출

```
pos.ridge[1:5]
```

```
##          jew      streets          rob         plan        stall
## 3.042999e-07 2.091328e-05 2.313824e-05 2.362472e-05 2.417472e-05
```

```
neg.ridge[1:5]
```

```
##        evolving     typecasting       cancelled     cameosimans
##     -0.10789309     -0.10340770     -0.10157122     -0.09999705
## selfdeprecating
##     -0.09974657
```

감정 단어 추출

```
pos.elastic <- sort(coef.elastic[coef.elastic > 0])
neg.elastic <- sort(coef.elastic[coef.elastic < 0])

length(pos.elastic)
```

```
## [1] 463
```

```
length(neg.elastic)
```

```
## [1] 612
```

감정 단어 추출

```
pos.elastic[1:5]
```

```
##            theni       mutilating              ive             ride
##     0.0002650247     0.0003448974     0.0003601619     0.0003829296
## weirdstrangeeven
##     0.0009908731
```

```
neg.elastic[1:5]
```

```
##          pic     returned       commit economically     enamored
##    -1.332630    -1.270371    -1.207446    -1.156321    -1.140002
```

감정 단어 점수화

```
library(tm.plugin.sentiment)
```

```
score.lasso <- polarity(tdm.train, names(pos.lasso), names(neg.lasso))
score.ridge <- polarity(tdm.train, names(pos.elastic), names(neg.elastic))
score.elastic <- polarity(tdm.train, names(pos.elastic), names(neg.elastic))
```

CUT-POINT

```
findCutpoint(data.train$sentiment, score.lasso)
```

```
## [1] 0.1428571
```

```
findCutpoint(data.train$sentiment, score.ridge)
```

```
## [1] 0.106383
```

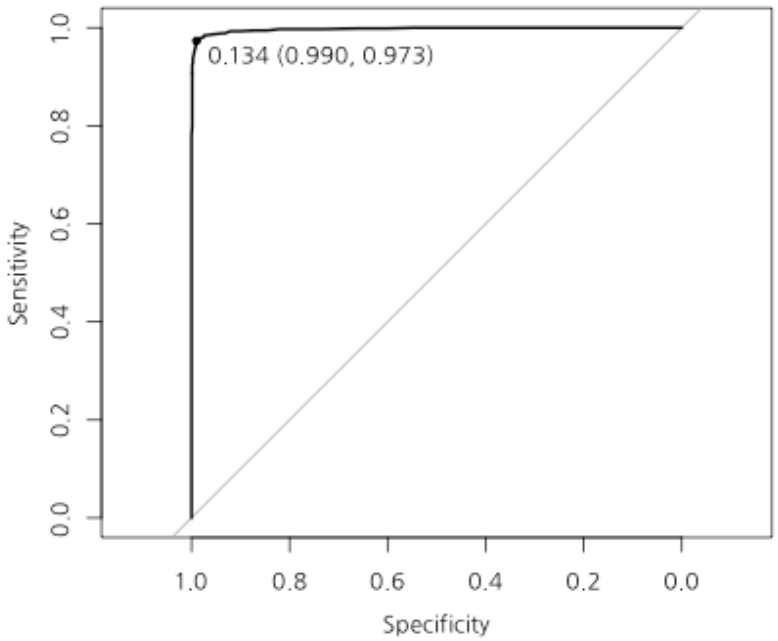
```
findCutpoint(data.train$sentiment, score.elastic)
```

```
## [1] 0.106383
```

CUT-POINT

```
library(pROC)
```

```
plot.roc(data.train$sentiment, score.lasso, print.thres = T)
```




```
##
```

CUT-POINT

```
cut.lasso <- findCutpoint(data.train$sentiment, score.lasso)
cut.ridge <- findCutpoint(data.train$sentiment, score.ridge)
cut.elastic <- findCutpoint(data.train$sentiment, score.elastic)
```

Test Set

```
corpus <- Corpus(VectorSource(data.test$review))

tdm.test <- TermDocumentMatrix(corpus,
                               control=list(dictionary = Terms(tdm.train),
                                            tolower = T,
                                            removePunctuation = T,
                                            removeNumbers = T,
                                            stopwords=stopwords("SMART")))
```

Test Set

```
score.lasso <- polarity(tdm.test, names(pos.lasso), names(neg.lasso))
score.ridge <- polarity(tdm.test, names(pos.elastic), names(neg.elastic))
score.elastic <- polarity(tdm.test, names(pos.elastic), names(neg.elastic))
```

Test Set

```
library(caret)
```

```
score.lasso.b <- rep(0, length(score.lasso))
score.lasso.b[score.lasso >= cut.lasso] <- 1
confusionMatrix(score.lasso.b, data.test$sentiment)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 251  68
##          1  58 223
##
##                Accuracy : 0.79
##                  95% CI : (0.7552, 0.8219)
##     No Information Rate : 0.515
##     P-Value [Acc > NIR] : <2e-16
##
```

Test Set

```
score.ridge.b <- rep(0, length(score.ridge))
score.ridge.b[score.ridge >= cut.ridge] <- 1
confusionMatrix(score.ridge.b, data.test$sentiment)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 248  58
##          1  61 233
##
##                Accuracy : 0.8017
##                  95% CI : (0.7675, 0.8329)
##     No Information Rate : 0.515
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.6031
##  Mcnemar's Test P-Value : 0.8545
##
```

Test Set

```
score.elastic.b <- rep(0, length(score.elastic))
score.elastic.b[score.elastic >= cut.elastic] <- 1
confusionMatrix(score.elastic.b, data.test$sentiment)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 248  58
##          1  61 233
##
##                Accuracy : 0.8017
##                  95% CI : (0.7675, 0.8329)
##     No Information Rate : 0.515
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.6031
##  Mcnemar's Test P-Value : 0.8545
##
```

glmnet 활용

```
score.lasso <- predict(cv.lasso, as.matrix(t(tdm.train)), s = "lambda.min")
score.ridge <- predict(cv.ridge, as.matrix(t(tdm.train)), s = "lambda.min")
score.elastic <- predict(cv.elastic, as.matrix(t(tdm.train)), s = "lambda.min")
```

glmnet 활용

```
findCutpoint(data.train$sentiment, score.lasso)
```

```
## [1] 0.6222667
```

```
findCutpoint(data.train$sentiment, score.ridge)
```

```
## [1] -0.01393038
```

```
findCutpoint(data.train$sentiment, score.elastic)
```

```
## [1] -0.06572849
```

```
cut.lasso <- findCutpoint(data.train$sentiment, score.lasso)
cut.ridge <- findCutpoint(data.train$sentiment, score.ridge)
cut.elastic <- findCutpoint(data.train$sentiment, score.elastic)
```

glmnet 활용

```
score.lasso <- predict(cv.lasso, as.matrix(t(tdm.test)), s = "lambda.min")
score.ridge <- predict(cv.ridge, as.matrix(t(tdm.test)), s = "lambda.min")
score.elastic <- predict(cv.elastic, as.matrix(t(tdm.test)), s = "lambda.min")
```

Test Set

```
score.lasso.b <- rep(0, length(score.lasso))
score.lasso.b[score.lasso >= cut.lasso] <- 1
confusionMatrix(score.lasso.b, data.test$sentiment)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 260  68
##          1  49 223
##
##                Accuracy : 0.805
##                  95% CI : (0.771, 0.836)
##     No Information Rate : 0.515
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.6089
##  Mcnemar's Test P-Value : 0.09609
##
```

Test Set

```
score.ridge.b <- rep(0, length(score.ridge))
score.ridge.b[score.ridge >= cut.ridge] <- 1
confusionMatrix(score.ridge.b, data.test$sentiment)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 238  40
##          1  71 251
##
##                Accuracy : 0.815
##                  95% CI : (0.7816, 0.8453)
##     No Information Rate : 0.515
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6308
##  Mcnemar's Test P-Value : 0.004407
##
```

Test Set

```
score.elastic.b <- rep(0, length(score.elastic))
score.elastic.b[score.elastic >= cut.elastic] <- 1
confusionMatrix(score.elastic.b, data.test$sentiment)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 248  34
##          1  61 257
##
##                Accuracy : 0.8417
##                  95% CI : (0.81, 0.87)
##     No Information Rate : 0.515
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6839
##  Mcnemar's Test P-Value : 0.007641
##
```

다른 데이터에 적용

Data

Amazon Books Reviews 중에서 2,000개

Test Set

```
books.review <- read.csv("data/Amazon_books.csv", stringsAsFactors = F)
```

Test Set

```
corpus <- Corpus(VectorSource(books.review$texts))

tdm.test <- TermDocumentMatrix(corpus,
                               control=list(dictionary = Terms(tdm.train),
                                            tolower = T,
                                            removePunctuation = T,
                                            removeNumbers = T,
                                            stopwords=stopwords("SMART")))
```

Test Set

```
score.elastic <- polarity(tdm.test, names(pos.elastic), names(neg.elastic))
```

Test Set

```
score.elastic.b <- rep(0, length(score.elastic))
score.elastic.b[score.elastic >= cut.elastic] <- 1
confusionMatrix(score.elastic.b, books.review$sentiment)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 490 154
##          1 510 846
##
##                Accuracy : 0.668
##                  95% CI : (0.6469, 0.6886)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.336
##  Mcnemar's Test P-Value : < 2.2e-16
##
```