

Lecture 18: Classification

Big Data and Machine Learning for Applied Economics
Econ 4676

Ignacio Sarmiento-Barbieri

Universidad de los Andes

October 15, 2020

Agenda

- 1 Recap
 - Elastic Net
 - Lasso for Causality
- 2 Classification
 - K-Nearest Neighbors
 - Logit
 - Logit Demo
- 3 Review & Next Steps
- 4 Further Readings

Elastic Net

- ▶ Naive Elastic Net

$$\min_{\beta} NEL(\beta) = \sum_{i=1}^n (y_i - x_i' \beta)^2 + \lambda_1 \sum_{s=2}^p |\beta_s| + \lambda_2 \sum_{s=2}^p \beta_s^2 \quad (1)$$

- ▶ Elastic Net: rescaled version
- ▶ Double Shrinkage introduces “too” much bias, *final* version “corrects” for this

$$\hat{\beta}_{EN} = \frac{1}{\sqrt{1 + \lambda_2}} \hat{\beta}_{naive EN} \quad (2)$$

- ▶ Careful sometimes software asks.

Lasso for Causality

Inference with Selection among Many Controls

$$y_i = \alpha d_i + x_i' \theta_y + r_{yi} + \zeta_i \quad (3)$$

- ▶ We apply variable selection methods to each of the two reduced form equations and then use all of the selected controls in estimation of α .
- ▶ We select
 - 1 A set of variables that are useful for predicting y_i , say x_{yi} , and
 - 2 A set of variables that are useful for predicting d_i , say x_{di} .
- ▶ We then estimate α by ordinary least squares regression of y_i on d_i and the union of the variables selected for predicting y_i and d_i , contained in x_{yi} and x_{di} .
- ▶ We thus make sure we use variables that are important for either of the two predictive relationships to guard against OVB

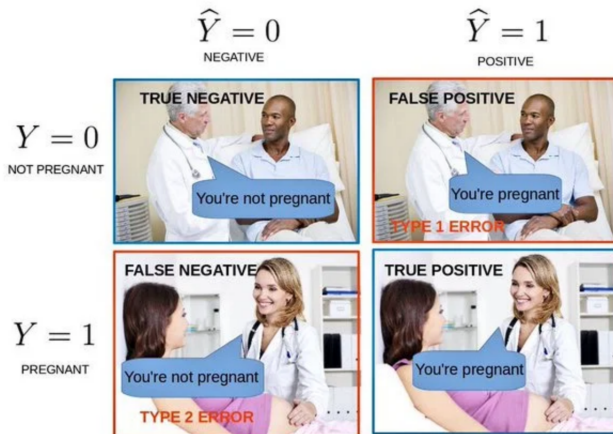
Classification

Classification: Motivation

- ▶ Admit a student to *PEG* based on their grades and LoR
- ▶ Give a credit, based on credit history, demographics?
- ▶ Classifying emails: spam, personal, social based on email contents
- ▶ Aim is to classify y based on X 's
- ▶ y can be
 - ▶ qualitative (e.g., spam, personal, social)
 - ▶ Not necessarily ordered
 - ▶ Not necessarily two categories, but will start with the binary case

Motivation

- ▶ Two states of nature $y \rightarrow i \in \{0, 1\}$
- ▶ Two actions $(\hat{y}) \rightarrow j \in \{0, 1\}$



Source: <https://dzone.com/articles/understanding-the-confusion-matrix>

Probability, Cost, and Classification

- ▶ Two states of nature $y \rightarrow i \in \{0, 1\}$
- ▶ Two actions $(\hat{y}) \rightarrow j \in \{0, 1\}$
- ▶ Probabilities
 - ▶ $p = \Pr(Y = 1|X)$
 - ▶ $1 - p = \Pr(Y = 0|X)$
- ▶ Loss: $L(i, j)$, penalizes being in bin i, j
- ▶ Risk: expected loss of taking action i

Probability, Cost, and Classification

- Risk: expected loss of taking action i

$$E[L(i, j)] = \sum_j p_j L(i, j) \quad (4)$$

- The objective is the same as before: minimize the risk
- We have to define $L(i, j)$

Probability, Cost, and Classification

- Risk: expected loss of taking action i

$$E[L(i, j)] = \sum_j p_j L(i, j) \quad (5)$$

$$R(i) = (1 - p)L(i, 0) + pL(i, 1)$$

- Loss 0-1: $L(i, j) = 1[i \neq j]$
- the expected loss will be negative (i.e. you will expect to make a profit (min costs))

$$R(1) < R(0) \quad (6)$$

$$1 - p < p$$

$$p > \frac{1}{2}$$

- This is known as the Bayes Classifier

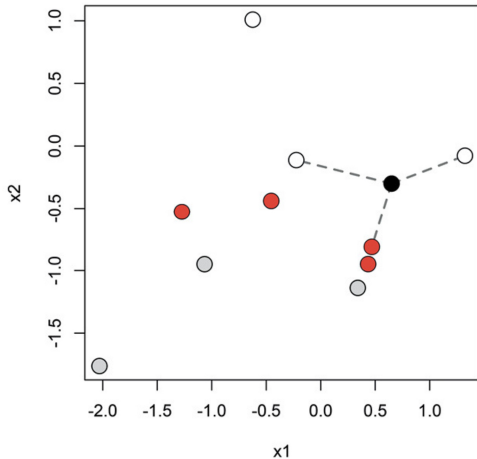
Probability, Cost, and Classification

- ▶ Under a 0-1 penalty the problem boils down to finding $p = PR(Y = 1|X)$
- ▶ We then predict 1 if $p > 0.5$ and 0 otherwise (Bayes classifier)
- ▶ We can think 3 ways of finding this probability in binary cases
 - ▶ K-Nearest Neighbors
 - ▶ Logistic
 - ▶ LDA
- ▶ Why not $p = X\beta$?

K-Nearest Neighbors

K-Nearest Neighbors

- K nearest neighbor (K-NN) algorithm predicts class \hat{y} for x by asking *What is the most common class for observations around x ?*



Source: Taddy (2019)

Lecture 18

K-Nearest Neighbors

- ▶ K nearest neighbor (K-NN) algorithm predicts class \hat{y} for x by asking *What is the most common class for observations around x ?*
- ▶ Algorithm: given an input vector x_f where you would loik to predict the class label
 - ▶ Find the K nearest neighbors in the dataset of labeled observations, $x_i, y_{i=1}^n$, the most common distance is the Euclidean distance:

$$d(x_i, x_f) = \sqrt{\sum_{j=1}^p (x_{ij} - x_{fj})^2} \quad (7)$$

- ▶ This yields a set of the K nearest observations with labels:

$$[x_{i1}, y_{i1}], \dots, [x_{iK}, y_{iK}] \quad (8)$$

- ▶ The predicted class of x_f is the most common class in this set

$$\hat{y}_f = \text{mode}\{y_{i1}, \dots, y_{iK}\} \quad (9)$$

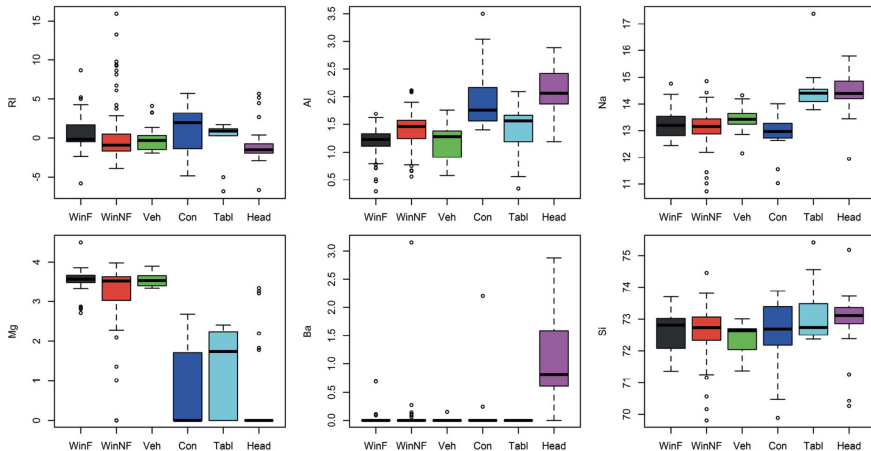
K-Nearest Neighbors

```
#Load the required packages
library("dplyr") #for data wrangling
library("class") #for KNN
library("MASS") # a library of example datasets
#Read the data
data(fgl) ## loads the data into R; see help(fgl)
str(fgl)
```

```
## 'data.frame':    214 obs. of  10 variables:
## $ RI : num  3.01 -0.39 -1.82 -0.34 -0.58 ...
## $ Na : num  13.6 13.9 13.5 13.2 13.3 ...
## $ Mg : num  4.49 3.6 3.55 3.69 3.62 3.61 3.6 3.61 3.58 3.6 ...
## $ Al : num  1.1 1.36 1.54 1.29 1.24 1.62 1.14 1.05 1.37 1.36 ...
## $ Si : num  71.8 72.7 73 72.6 73.1 ...
## $ K : num  0.06 0.48 0.39 0.57 0.55 0.64 0.58 0.57 0.56 0.57 ...
## $ Ca : num  8.75 7.83 7.78 8.22 8.07 8.07 8.17 8.24 8.3 8.4 ...
## $ Ba : num  0 0 0 0 0 0 0 0 0 0 ...
## $ Fe : num  0 0 0 0 0 0.26 0 0 0 0.11 ...
## $ type: Factor w/ 6 levels "WinF","WinNF",...: 1 1 1 1 1 1 1 1 1 1 ...
```

Refractive index and chemical composition for six possible glass types: float glass window (WinF), nonfloat glass window (WinNF), vehicle window (Veh), container (Con), tableware (Tabl), vehicle headlamp (Head)

K-Nearest Neighbors



K-Nearest Neighbors

- ▶ Units matter
 - ▶ Since distance is measured on the raw x values, units matter.
 - ▶ As we did for regularization, we will standardized observations.
 - ▶ R `scale` function does this, i.e., convert columns to mean-zero sd-one

```
x <- scale(fgl[,1:9]) # column 10 is the class label
apply(x,2,sd) # see ?apply
```

```
## RI Na Mg Al Si K Ca Ba Fe
## 1 1 1 1 1 1 1 1 1
```

K-Nearest Neighbors

- ▶ Run Knn
- ▶ Make sure you have numeric matrices of training data x values, with labels y
- ▶ Also need to provide new *test* values where you would like to predict
- ▶ Note that there's no model do fit, Knn, just counts neighbors for each observation in test

```
set.seed(1010101)
test <- sample(1:214,10)
nearest1 <- knn(train=x[-test,], test=x[test,], cl=fgl$type[-test], k=1)
nearest5 <- knn(train=x[-test,], test=x[test,], cl=fgl$type[-test], k=5)
data.frame(fgl$type[test],nearest1,nearest5)
```

```
##      fgl.type.test. nearest1 nearest5
## 1           WinF         WinF   WinNF
## 2           Head         Head     Head
## 3          WinNF         WinNF   WinNF
## 4           WinF         WinF   WinF
## 5          WinNF         WinNF   WinNF
## 6          WinNF         WinNF   WinNF
## 7           Head          Con     Con
## 8           Head         WinNF   WinNF
## 9          WinNF         WinNF   WinNF
## 10          WinNF         WinF    WinF
```

K-Nearest Neighbors

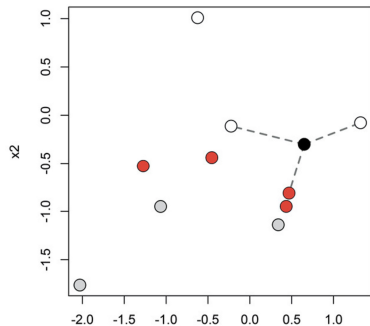
- ▶ In this case
 - ▶ 1-Knn manages 70% accuracy
 - ▶ 5-Knn manages 60% accuracy
 - ▶ H.W. try for different seed, (Taddy's is 80% and 70%)
- ▶ There are some major problems with practical implications
 - ▶ Knn predictions are unstable as a function of K

$$K = 1 \implies \hat{p}(\text{white}) = 0$$

$$K = 2 \implies \hat{p}(\text{white}) = 1/2$$

$$K = 3 \implies \hat{p}(\text{white}) = 2/3$$

$$K = 4 \implies \hat{p}(\text{white}) = 1/2$$



K-Nearest Neighbors

- ▶ In this case
 - ▶ 1-Knn manages 70% accuracy
 - ▶ 5-Knn manages 60% accuracy
 - ▶ H.W. try for different seed, (Taddy's is 80% and 70%)
- ▶ There are some major problems with practical implications
 - ▶ Knn predictions are unstable as a function of K
 - ▶ This instability of prediction makes it hard to choose the optimal K and cross validation doesn't work well for KNN
 - ▶ Since prediction for each new x requires a computationally intensive counting, KNN is too expensive to be useful in most big data settings.
 - ▶ KNN is a good idea, but too crude to be useful in practice

Logit

Logit

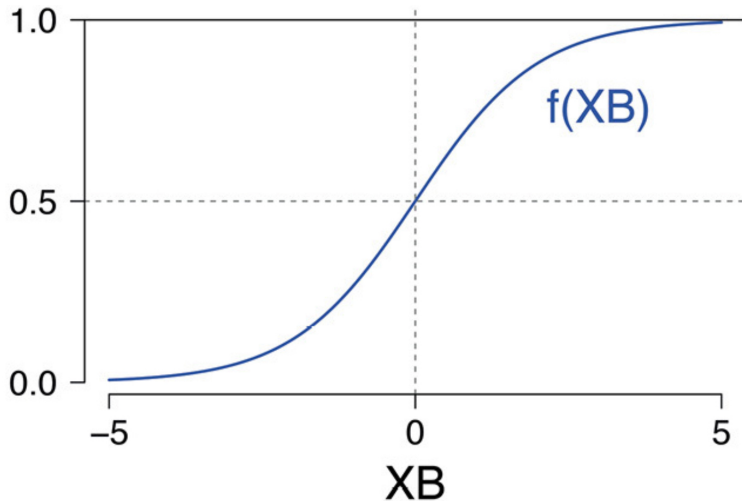
We have a conditional probability

$$Pr(y = 1|X) = f(X'\beta) \quad (10)$$

Logistic regression uses a *logit* (sigmoid, softmax) link function

$$p(y = 1|X) = \frac{e^{X'\beta}}{1 + e^{X'\beta}} = \frac{\exp(\beta_0 + \beta_1 x_1 + \cdots + \beta_k x_k)}{1 + \exp(\beta_0 + \beta_1 x_1 + \cdots + \beta_k x_k)} \quad (11)$$

Logit



Source: Taddy (2019)

Logit Demo

```
#Load the required packages  
library("dplyr") #for data wrangling  
library("gamlr") #ML
```

```
#Read the data  
credit<-readRDS("credit_class.rds")  
dim(credit)
```

```
## [1] 1000    9
```

```
head(credit)
```

```
##   Default duration amount installment age  history      purpose foreign  rent  
## 1      0         6    1169           4  67 terrible goods/repair foreign FALSE  
## 2      1        48    5951           2  22      poor goods/repair foreign FALSE  
## 3      0        12    2096           2  49 terrible          edu foreign FALSE  
## 4      0        42    7882           2  45      poor goods/repair foreign FALSE  
## 5      1        24    4870           3  53      poor    newcar foreign FALSE  
## 6      0        36    9055           2  35      poor          edu foreign FALSE
```


Logit Demo

```
mylogit <- glm(Default~duration + amount + installment + age +  
               factor(history) + factor(purpose) + factor(foreign) + factor(rent),  
               data = credit, family = "binomial")  
summary(mylogit,type="text")
```

```
## ...  
##  
## Coefficients:  
##              Estimate Std. Error z value Pr(>|z|)  
## (Intercept)    -1.936e-01  4.579e-01  -0.423  0.67239  
## duration        2.666e-02  8.152e-03   3.270  0.00108 **  
## amount          9.793e-05  3.670e-05   2.668  0.00763 **  
## installment     2.361e-01  7.687e-02   3.072  0.00213 **  
## age            -1.598e-02  7.348e-03  -2.175  0.02964 *  
## factor(history)poor    -1.101e+00  2.490e-01  -4.424  9.67e-06 ***  
## factor(history)terrible -1.849e+00  2.837e-01  -6.518  7.13e-11 ***  
## factor(purpose)usedcar   -1.702e+00  3.273e-01  -5.201  1.98e-07 ***  
## factor(purpose)goods/repair -7.551e-01  1.867e-01  -4.044  5.25e-05 ***  
## factor(purpose)edu       -1.473e-01  3.263e-01  -0.451  0.65166  
## factor(purpose)biz       -8.501e-01  2.801e-01  -3.036  0.00240 **  
## factor(foreign)german  -1.322e+00  5.814e-01  -2.274  0.02298 *  
## factor(rent)TRUE       5.702e-01  1.944e-01   2.934  0.00335 **  
## ---  
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1  
##  
## ...
```

Logit Demo: Build a design matrix

```
str(credit$foreign)
```

```
## Factor w/ 2 levels "foreign","german": 1 1 1 1 1 1 1 1 1 1 ...
```

```
source("naref.R")  
credit<-naref(credit)  
str(credit$foreign)
```

```
## Factor w/ 3 levels NA,"foreign","german": 2 2 2 2 2 2 2 2 2 2 ...
```

```
credx <- model.matrix( Default ~ .^2, data=credit)[,-1]  
dim(credx)
```

```
## [1] 1000 121
```

```
colnames(credx)[c(1,2,16,17,18)]
```

```
## [1] "duration"          "amount"            "rentTRUE"  
## [4] "duration:amount"   "duration:installment"
```

Logit Demo

```
credx <- sparse.model.matrix( Default ~ .^2, data=credit)[,-1]  
head(credx)
```

```
## 6 x 121 sparse Matrix of class "dgCMatrix"
```

```
##
```

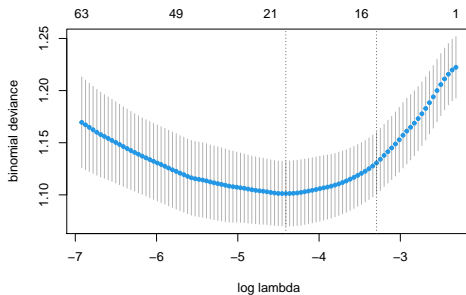
```
## 1  6 1169 4 67 . . 1 . . 1 . . 1 . 1 . 7014 24 402 . . 6 . . 6 . . 6 .  
## 2 48 5951 2 22 . 1 . . . 1 . . 1 . 1 . 285648 96 1056 . 48 . . . 48 . . 48 .  
## 3 12 2096 2 49 . . 1 . . . 1 . 1 . 1 . 25152 24 588 . . 12 . . . 12 . 12 .  
## 4 42 7882 2 45 . 1 . . . 1 . . 1 . 1 . 331044 84 1890 . 42 . . . 42 . . 42 .  
## 5 24 4870 3 53 . 1 . 1 . . . 1 . 1 . 116880 72 1272 . 24 . 24 . . . 24 .  
## 6 36 9055 2 35 . 1 . . . . 1 . 1 . 1 . 325980 72 1260 . 36 . . . . 36 . 36 .  
##
```

Logit Demo

```
default <- credit$Default  
credscore <- cv.gamlr(credx, default, family="binomial", verb=TRUE)
```

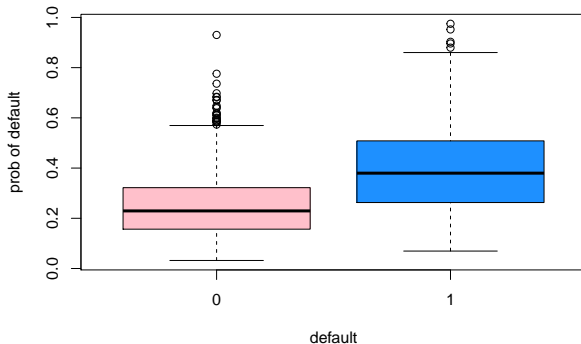
```
## fold 1,2,3,4,5,done.
```

```
plot(credscore)
```



Logit Demo

```
## What are the underlying default probabilities  
## In sample probability estimates  
pred <- predict(credscore$gamlr, credx, type="response")  
pred <- drop(pred) # remove the sparse Matrix formatting  
boxplot(pred ~ default, xlab="default", ylab="prob of default", col=c("pink","dodgerblue"))
```



Logit Demo Misclassification Rates

- ▶ A classification rule, or cutoff, is the probability p at which you predict
 - ▶ $\hat{y}_i = 0$ if $p_i < p$
 - ▶ $\hat{y}_i = 1$ if $p_i \geq p$
- ▶ We have two types of error associated with this that we can use as a measure of performance

$$\text{False Positive Rate} = \frac{\text{expected \# false positives}}{\text{\# classified positive}}$$

$$\text{False Negative Rate} = \frac{\text{expected \# false negatives}}{\text{\# classified negative}} \quad (12)$$

- ▶ Another measure of performance is using the number of *correct* classifications
 - ▶ *Sensitivity* proportion of true $y = 1$ classified as such
 - ▶ *Specificity* proportion of true $y = 0$ classified as such

Logit Demo

```
rule <- 1/2  
sum( (pred>rule)[default==0] )/sum(pred>rule) ## false positive rate
```

```
## [1] 0.3189655
```

```
sum( (pred<rule)[default==1] )/sum(pred<rule) ## false negative rate
```

```
## [1] 0.25
```

```
sum( (pred>rule)[default==1] )/sum(default==1) ## sensitivity
```

```
## [1] 0.2633333
```

```
sum( (pred<rule)[default==0] )/sum(default==0) ## specificity
```

```
## [1] 0.9471429
```

Logit Demo

```
rule <- 1/5
```

```
sum( (pred>rule)[default==0] )/sum(pred>rule) ## false positive rate
```

```
## [1] 0.6059744
```

```
sum( (pred<rule)[default==1] )/sum(pred<rule) ## false negative rate
```

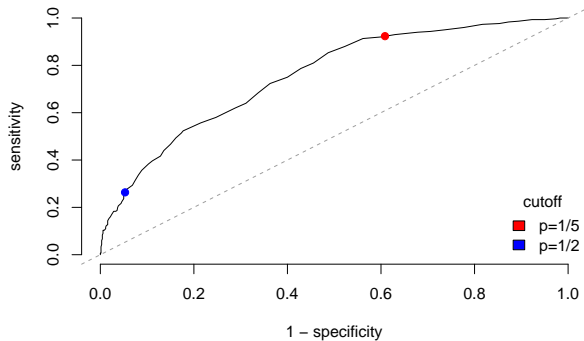
```
## [1] 0.07744108
```

```
sum( (pred>rule)[default==1] )/sum(default==1) ## sensitivity
```

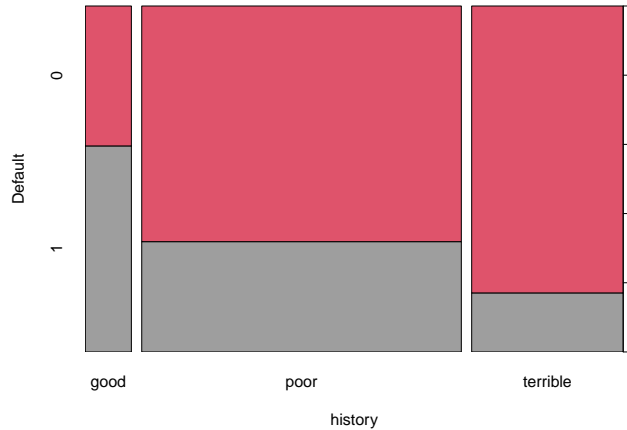
```
## [1] 0.9233333
```

```
sum( (pred<rule)[default==0] )/sum(default==0) ## specificity
```

```
## [1] 0.3914286
```

Logit Demo



Review & Next Steps

- ▶ Today:
 - ▶ KNN
 - ▶ Intuitive
 - ▶ Not very useful in practice, curse of dimensionality
 - ▶ Logit
 - ▶ `gamlr`
 - ▶ Exploit sparsity
 - ▶ Model evaluation
 - ▶ Careful with naive models
- ▶ Next class: Classification (cont.)
- ▶ Questions? Questions about software?

Further Readings

- ▶ Friedman, J., Hastie, T., & Tibshirani, R. (2001). The elements of statistical learning (Vol. 1, No. 10). New York: Springer series in statistics.
- ▶ James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). An introduction to statistical learning (Vol. 112, p. 18). New York: springer.
- ▶ Kuhn, M. (2012). The caret package. R Foundation for Statistical Computing, Vienna, Austria. <https://topepo.github.io/caret/index.html>
- ▶ Taddy, M. (2019). Business data science: Combining machine learning and economics to optimize, automate, and accelerate business decisions. McGraw Hill Professional.
- ▶ Zou, H. y Hastie, T., 2005, Regularization and variable selection via the elastic net, Journal of the Royal Statistical Society, 67, 2, 301-320.