Lecture 18: Classification

Big Data and Machine Learning for Applied Economics
Econ 4676

Ignacio Sarmiento-Barbieri

Universidad de los Andes

October 19, 2021

Agenda

Elastic Net

- Naive Elastic Net

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

- Elastic Net: reescaled version. Double Shrinkage introduces "too" much bias, *final* version "corrects" for this

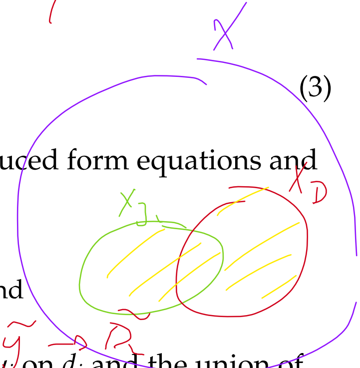
$$\hat{\beta}_{EN} = \frac{1}{\sqrt{1+\lambda_2}}\hat{\beta}_{naive\,EN} \quad (2)$$

Lasso for Causality

Inference with Selection among Many Controls

FWL

$$y_i = \alpha D_i + X_i'\theta_y + r_{yi} + \zeta_i \tag{3}$$



- 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

$y \in \mathbb{R}$

Classification

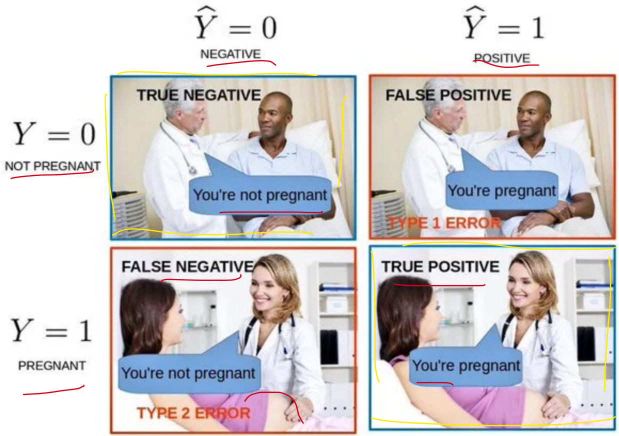
Classification: Motivation

- 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 $y \to n \in \{0,1\}$
- Two actions $(\hat{y}) \to a \in \{0,1\}$ Actions

Nature



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

Probability, Cost, and Classification

- Two states of nature $y \to n \in \{0, 1\}$ ✓
- Two actions $(\hat{y}) \to a \in \{0, 1\}$ ✓
- Probabilities
  - $p = Pr(y = 1|X)$
  - $1 - p = Pr(y = 0|X)$
- Loss: $L(a, n)$, penalizes being in bin (a, n)
- Risk: expected loss of taking action a

$$\underbrace{L = (\theta - a)^2}$$

$$L = \begin{cases} 1 \\ 0 \end{cases}$$

$$E(L(a, n))$$

Probability, Cost, and Classification

$n = \begin{cases} 1 \\ 0 \end{cases}$

- Risk: expected loss of taking action a

$- \int L f \, dx$

$L = (\theta - a)^2$

$$E[L(a,n)] = \sum_n p_n L(a,n) \quad (4)$$

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

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

$L \quad [0, \infty)$

$[0, 1]$

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

$L(a,n) = \begin{cases} 1 & a \neq n \\ 0 & a = n \end{cases}$

Probability, Cost, and Classification

- Which action do we choose?

Probability, Cost, and Classification

- Which action do we choose?
- We can compare the risk of each action
- We are going to choose to take action 1 when the risk is lower:

$R(1)$

$$\begin{aligned} R(1) &< R(0) \\ 1-p &< p \\ \boxed{p} &> \frac{1}{2} \end{aligned} \tag{5}$$

$R(0)$

- This is known as the Bayes Classifier, choose the estate that minimizes the risk

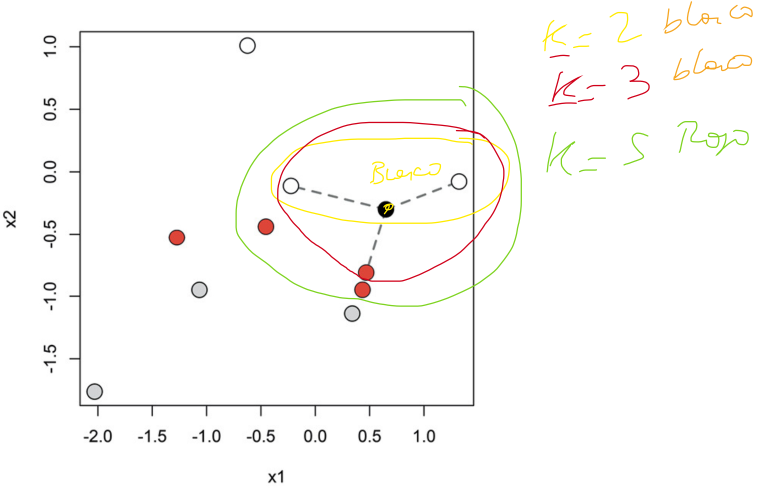
Probability, Cost, and Classification

- 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)

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 like to predict the class label
  - Find the K nearest neighbors in the dataset of labeled observations, $x_i, y_i{}_{i=1}^{n}$, the most common distance is the Euclidean distance (units):

$$d(x_i, x_f) = \sqrt{\sum_{j=1}^{p} (x_{ij} - x_{fj})^2} \tag{6}$$

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

$$[x_{i1}, y_{i1}], \ldots, [x_{iK}, y_{iK}] \tag{7}$$

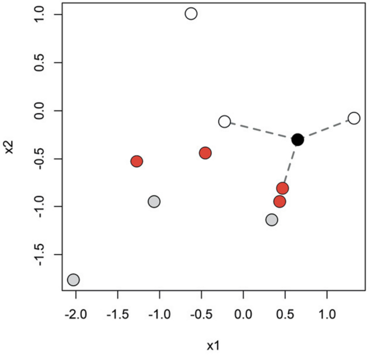
  - The predicted class of x_f is the most common class in this set

$$\hat{y}_f = mode\{y_{i1}, \ldots, y_{iK}\} \tag{8}$$

K-Nearest Neighbors

- There are some major problems with practical applications
  - Knn predictions are unstable as a function of K

$$K = 1 \implies \hat{p}(white) = 0$$
$$K = 2 \implies \hat{p}(white) = 1/2$$
$$K = 3 \implies \hat{p}(white) = 2/3$$
$$K = 4 \implies \hat{p}(white) = 1/2$$



Source: Taddy (2019)

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%)

→ Demo

- 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

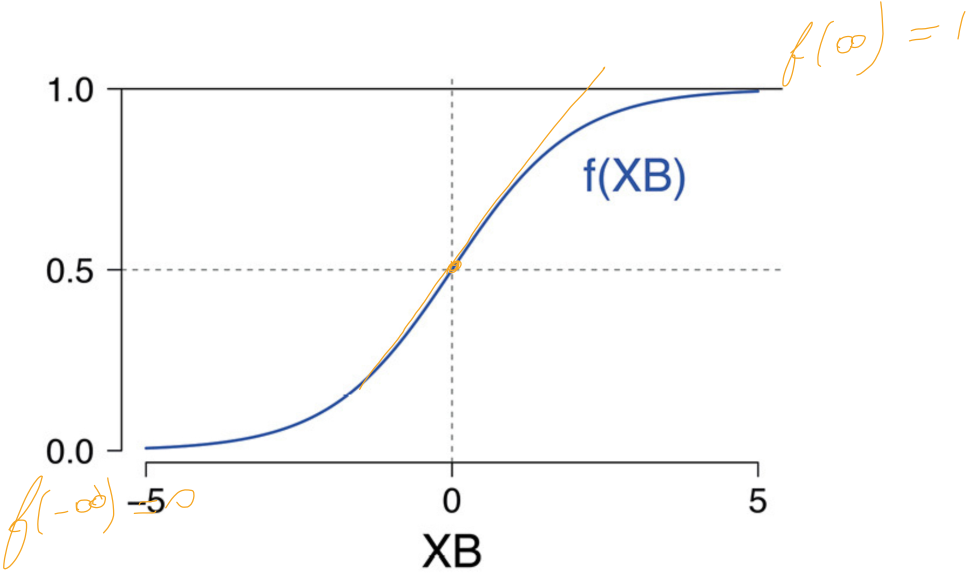
We have a conditional probability

$$Pr(y = 1|X) = f(X'\beta) \tag{9}$$

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

$$log \underbrace{\left(\frac{p(y = 1|X)}{1 - p(y = 1|X)} \right)}_{\text{odds ratio}} = \beta_0 + \beta_1 x_1 + \cdots + \beta_k x_k \tag{10}$$

Logit



Source: Taddy (2019)

Logit

We have a conditional probability

$$Pr(y = 1|X) = f(X'\beta) \tag{11}$$

Can recover predictions:

$$p(y = 1|X) = \frac{e^{X'\beta}}{1 + e^{X'\beta}} = \frac{exp(\beta_0 + \beta_1 x_1 + \cdots + \beta_k x_k)}{1 + exp(\beta_0 + \beta_1 x_1 + \cdots + \beta_k x_k)} \tag{12}$$

Linear Discriminant Analysis

Linear Discriminant Analysis

Reverend Bayes to the rescue: Bayes Theorem

$$p(y=1|X) = \frac{f(X|y=1)p(y=1)}{m(X)} \tag{13}$$

$f(x/y=0)$

with $m(X)$ is the marginal distribution of X, i.e.

$$m(X) = \int f(X|y=1)p(y=1)dy \tag{14}$$

Recall that there are two states of nature $y \to i \in \{0,1\}$

$$\begin{aligned} m(X) &= f(X|y=1)p(y=1) + f(X|y=0)p(y=0) \\ &= f(X|y=1)p(y=1) + f(X|y=0)(1-p(y=1)) \end{aligned} \tag{15}$$

Linear Discriminant Analysis

- This is basically an empirical Bayes approach
- We need to estimate $f(X|y=1), f(X|y=0)$ and $p(y=1)$
  - Let's start by estimating $p(y=1)$. We've done this before

$$p(y=1) = \frac{\sum_{i=1}^{n} 1[y_i = 1]}{N} \tag{16}$$

  - Next $f(X|y=j)$ with $j = 0, 1$.
    - if we assume one predictor and $X|y \sim N(\mu_j, \sigma_j)$
    - the problem boils down to estimating μ_j, σ_j
    - LDA makes it simpler, assumes $\sigma_j = \sigma \; \forall j$
    - then partition the sample in two $y = 0$ and $y = 1$, estimate the moments and get $\hat{f}(X|y=j)$
  - Plug everything into the Bayes Rule and you're done

μ_0 $\sigma_0 = \sigma_1$

μ_1

Linear Discriminant Analysis
Extensions

- If we have k predictors?
- then $X|y \sim NM(\mu, \Sigma)$

$$f(X|y=j) = \frac{1}{(2\pi)^{k/2}|\Sigma|^{1/2}} exp(-\frac{1}{2}(x-\mu_j)'\Sigma_j(x-\mu_j)) \quad (17)$$

- μ_j is the vector of the sample means in each partition $j = 0, 1$
- Σ_j is the matrix of variance and covariances of each partition $j = 0, 1$
- Can we lift normality?

Linear Discriminant Analysis

- Why is it call linear?
- Note

$$p > \frac{1}{2} \iff ln(\frac{p}{(1-p)}) \quad (18)$$

- Logit with one predictor

$$\beta_1 + \beta_2 X \quad (19)$$

- Classification: in the probability of space
- Discrimination: in the space of X
- $\beta_1 + \beta_2 X$ is the discrimination function for logit (it is lineal)

Linear Discriminant Analysis

HW ?
ISLR

- LDA?
- One predictor with $\sigma_0 = \sigma_1$ (equal variance)

$$p(y=1|X) = \frac{f(X|y=1)p(y=1)}{f(X|y=1)p(y=1) + f(X|y=0)(1-p(y=1))} \tag{20}$$

- Then under the equal variance assumption

$$\frac{p(y=1|X)}{1-p(y=1|X)} = \frac{f(X|y=1)p(y=1)}{f(X|y=0)(1-p(y=1))} \tag{21}$$

$$= \frac{p(y=1)exp((x-\mu_1)^2)}{(1-p(y=1))exp((x-\mu_0)^2)} \tag{22}$$

Linear Discriminant Analysis

- Taking logs

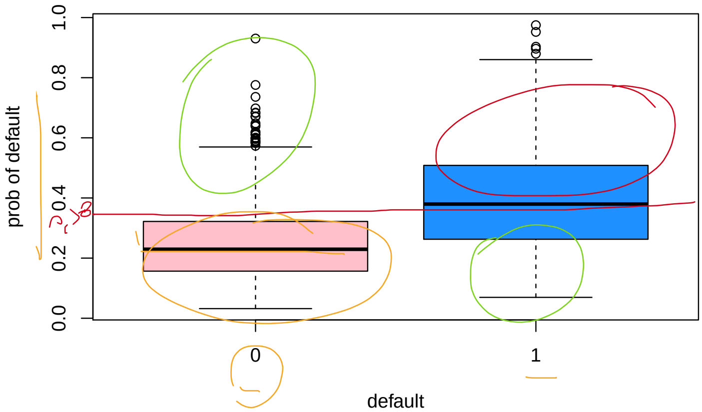
$$
\begin{aligned}
log\left(\frac{p(y=1|X)}{1-p(y=1|X}\right) &= log(\frac{p(y=1)}{(1-p(y=1))} + (x-\mu_1)^2 - (x-\mu_0)^2 && (23)\\
&= log(\frac{p(y=1)}{(1-p(y=1))} + \mu_1^2 - \mu_0^2 - 2(\mu_1-\mu_0)x && (24)\\
&= \gamma_1 + \gamma_2 X && (25)
\end{aligned}
$$

- under the assumption of equal variance the discrimination function is lineal
- Note: logit estimates γ_1 and γ_2

Misclassification Rates

Misclassification Rates

- Predicted probabilities from Logit model



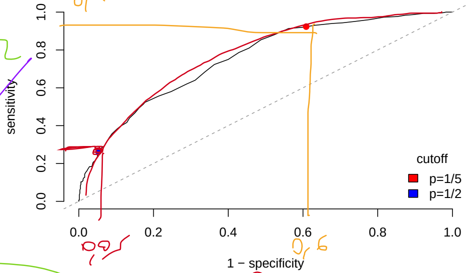
Misclassification Rates

ROC

- 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 $p_i \cancel{<} p$
- Measures of performance
  - *1-Specificity:* False Positive Rate, Type I error
  - *Sensitivity:* True Positive Rate, power, (1-Type II error)

TPR

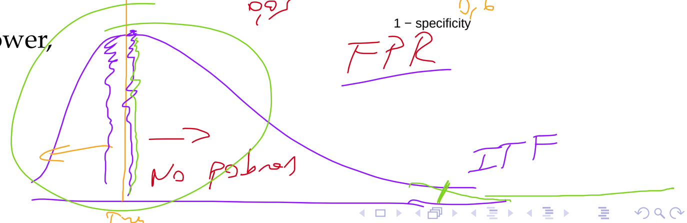
0,90



0,05

0,6

FPR



No pobres

ITF

ROC

- ROC curve: Receiver operating characteristic curve
- ROC curve illustrates the trade-off of the classification rule
- Gives us the ability
  - Measure the predictive capacity of our model
  - Compare between models – → ver apendice demostracion
- Some definitions
  - $P = \sum y_i$ positives
  - $N = \sum (1 - y_i)$ negatives
  - $T = P + N$ all observations
  - True Positives: $TP = \sum \hat{y}_i y_i$, True Positive Rate = $\frac{TP}{P}$
  - False Positives: $FP = \sum \hat{y}_i (1 - y_i)$, False Positive Rate = $\frac{FP}{N}$

Algebra

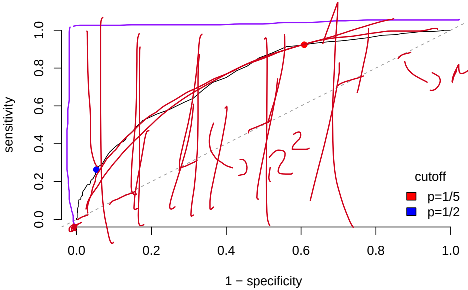
ROC

Roc = 1

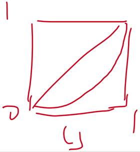
Roc < 1

- Binary Classifier: $\hat{y}_i = 1[p_i > c], c \in [0, 1]$
- Bayes fixes $c = 0.5$
- Ideally $TPR = 1$ and $FPR = 0$
- ROC curve give us the locus of all possible TPR and FPR for all possible $c \in [0, 1]$

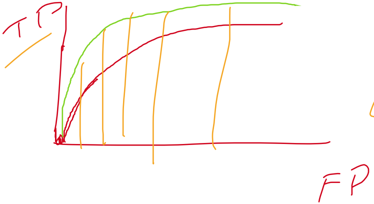
TPR



FPR



ROC



c = ?

- ROC Properties
  - Has positive slope
    - In $(0,0)$, $c=1$. When $c \downarrow$, $TP \uparrow$ and $FP \uparrow$. Then

$$TPR = \sum \frac{\hat{y}_i y_i}{P} \quad FPR = \sum \frac{\hat{y}_i(1-y_i)}{T-P} \tag{26}$$

    - Is easy to show

HW

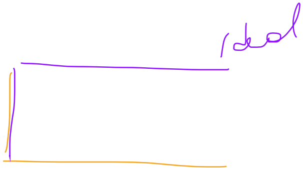
$$TPR = \frac{\sum \hat{y}_i}{P} - \frac{T-P}{P} FPR \tag{27}$$

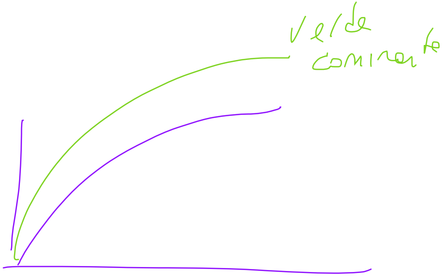
    - ROC is the locus of all possible TPR and FPR for all possible $c \in [0,1]$

$$TPR = \frac{\sum \hat{y}_i(c)}{P} - \frac{T-P}{P} FPR(c) \tag{28}$$

ROC: Summary

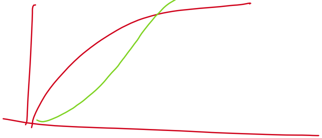
- Ideal ROC curve
- AUC: area under the curve, is like an R^2
- Help us compare between classifiers
- Dominated classifiers?
- Which c? Choose a max ~~F~~PR





MIN FPR

o alguna condición?



Review & Next Steps

- Review Classification:
  - KNN
    - Intuitive
    - Not very useful in practice, curse of dimensionality
  - Logit
  - Linear Discriminant Analysis $\sigma_j = \sigma$ Link
  - Misclassification Rates: ROC curve
  - QDA? $\sigma_j \neq \sigma_i$
  - Multiple Classes?
- Next class: ~~Problem Sets, Text Data!~~ Arboles < Reg / Classif
- 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.

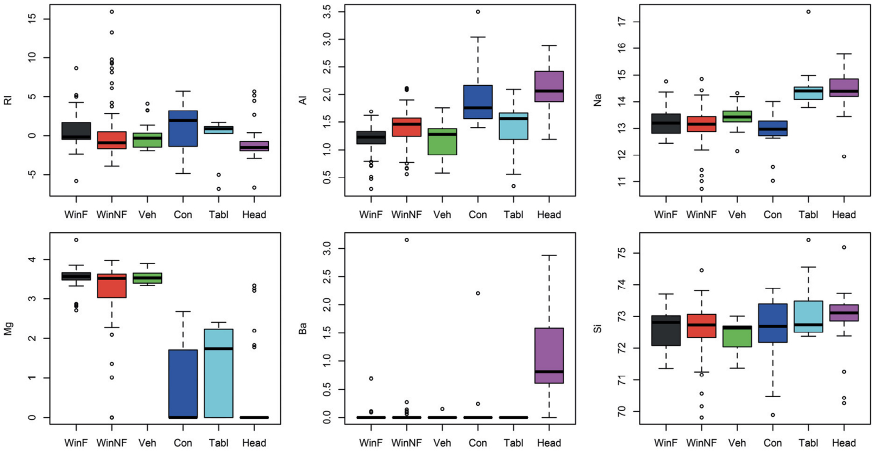
K-Nearest Neighbors

```r
#Load the required packages
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 standarized 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

- Before running `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`

```r
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
```

Logit Demo

```r
set.seed(101010) #sets a seed
credit<-readRDS("credit_class.rds")
#70% train
indic<-sample(1:nrow(credit),floor(.7*nrow(credit)))
#Partition the sample
train<-credit[indic,]
test<-credit[-indic,]
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
```

```r
dim(credit)
```

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

Logit Demo

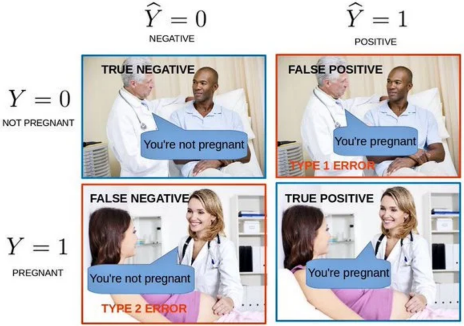
```r
mylogit <- glm(Default~duration + amount + installment + age
               + factor(history) + factor(purpose) + factor(foreign) + factor(rent),
               data = train, family = "binomial")
summary(mylogit)
```

```
##
## ...
##
## Coefficients:
##                              Estimate Std. Error z value Pr(>|z|)
## (Intercept)                -3.285e-01  5.597e-01  -0.587 0.557264
## duration                    1.625e-02  9.538e-03   1.704 0.088369 .
## amount                      1.518e-04  4.325e-05   3.511 0.000447 ***
## installment                 3.335e-01  9.216e-02   3.619 0.000296 ***
## age                        -1.762e-02  8.851e-03  -1.990 0.046554 *
## factor(history)poor        -1.212e+00  3.126e-01  -3.876 0.000106 ***
## factor(history)terrible    -1.989e+00  3.552e-01  -5.598 2.17e-08 ***
## factor(purpose)usedcar     -1.813e+00  4.067e-01  -4.459 8.23e-06 ***
## factor(purpose)goods/repair -7.163e-01  2.254e-01  -3.177 0.001486 **
## factor(purpose)edu          1.207e-01  3.858e-01   0.313 0.754450
## factor(purpose)biz         -9.862e-01  3.440e-01  -2.867 0.004147 **
## factor(foreign)german      -2.057e+00  8.213e-01  -2.505 0.012254 *
## factor(rent)TRUE            7.554e-01  2.355e-01   3.208 0.001337 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## ...
```

Logit Demo

```r
test$phat<- predict(mylogit, test, type="response")
test$Default_hat<-ifelse(test$phat>.5,1,0)
with(test,prop.table(table(Default,Default_hat)))
```

```
##          Default_hat
## Default          0          1
##       0 0.63666667 0.06666667
##       1 0.22666667 0.07000000
```



LDA: Demo

$$p(y=1) = \frac{\sum_{i=1}^{n} 1[y_i = 1]}{N} \tag{29}$$

```
p1<-sum(train$Default)/dim(train)[1]
p1
```

```
## [1] 0.3014286
```

$$\hat{\mu}_k = \frac{1}{n_k} \sum_{i:y_i=k} x_i \tag{30}$$

```
mu1<-mean(train$duration[train$Default==1])
mu1
```

```
## [1] 24.78673
```

```
mu0<-mean(train$duration[train$Default==0])
mu0
```

```
## [1] 19.79346
```

LDA: Demo

$$\hat{\sigma}^2 = \frac{1}{N-K}\sum_{k=1}^{K}\sum_{i:y_i=k}(x_i-\hat{\mu}_k)^2 \tag{31}$$

```
g1<-sum((train$duration[train$Default==1]-mu1)^2)
g0<-sum((train$duration[train$Default==0]-mu0)^2)


sigma<-sqrt((g1+g0)/(dim(train)[1]-2))
```

$$\hat{f}_k \sim N(\hat{\mu}_k, \hat{\sigma}) \tag{32}$$

```
f1<-dnorm(test$duration,mean=mu1,sd=sigma)
f0<-dnorm(test$duration,mean=mu0,sd=sigma)
```

LDA: Demo

```r
library("MASS")      # LDA
lda_simple <- lda(Default~duration, data = train)
lda_simple_pred<-predict(lda_simple,test)
names(lda_simple_pred)
```

```
## [1] "class"     "posterior" "x"
```

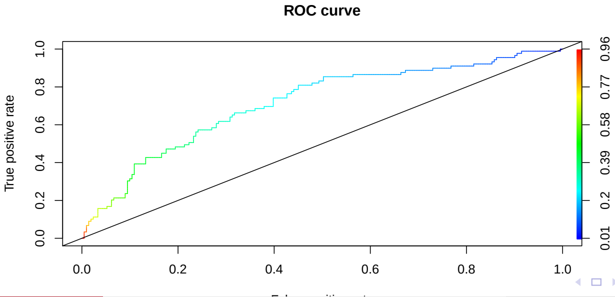
```r
posteriors<-data.frame(lda_simple_pred$posterior)
posteriors$hand<-f1*p1/(f1*p1+f0*(1-p1))
head(posteriors)
```

```
##           X0        X1      hand
## 1  0.8013656 0.1986344 0.1986344
## 3  0.7668614 0.2331386 0.2331386
## 14 0.6861792 0.3138208 0.3138208
## 16 0.6861792 0.3138208 0.3138208
## 28 0.7668614 0.2331386 0.2331386
## 33 0.7283950 0.2716050 0.2716050
```

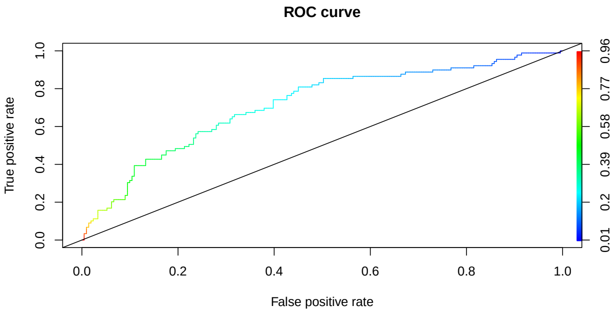
Roc Demo

```
library("ROCR") #Roc

mylogit <- glm(Default~duration + amount + installment + age
               + factor(history) + factor(purpose) + factor(foreign) + factor(rent),
               data = train, family = "binomial")
test$phat<- predict(mylogit, test, type="response")
pred <- prediction(test$phat, test$Default)
roc_ROCR <- performance(pred,"tpr","fpr")
plot(roc_ROCR, main = "ROC curve", colorize = T)
abline(a = 0, b = 1)
```



Roc Demo




```r
auc_ROCR <- performance(pred, measure = "auc")
auc_ROCR@y.values[[1]]
```

```
## [1] 0.714415
```

Roc Demo

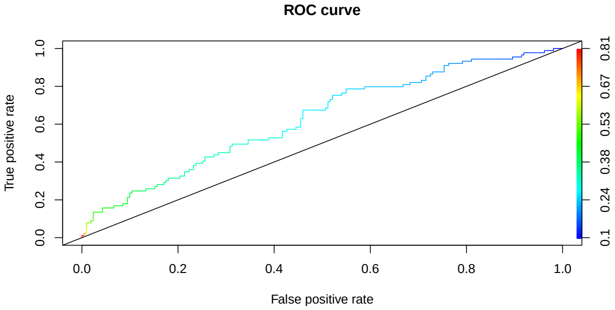
```r
mylda <- lda(Default~duration + amount + installment + age , data = train)
mylda
```

```
## Call:
## lda(Default ~ duration + amount + installment + age, data = train)
##
## Prior probabilities of groups:
##         0         1
## 0.6985714 0.3014286
##
## Group means:
##   duration   amount installment      age
## 0 19.79346 3062.888    2.885481 36.40900
## 1 24.78673 4057.791    3.109005 33.85782
##
## Coefficients of linear discriminants:
##                      LD1
## duration     0.0296041361
## amount       0.0002055164
## installment  0.4821242957
## age         -0.0386710882
```

Roc Demo

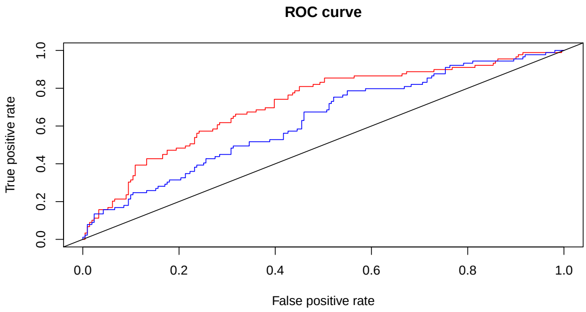
```
phat_mylda<- predict(mylda, test, type="response")
pred_mylda <- prediction(phat_mylda$posterior[,2], test$Default)

roc_mylda <- performance(pred_mylda,"tpr","fpr")
plot(roc_mylda, main = "ROC curve", colorize = T)
abline(a = 0, b = 1)
```



Roc Demo

```
plot(roc_ROCR, main = "ROC curve", colorize = FALSE, col="red")
plot(roc_mylda,add=TRUE, colorize = FALSE, col="blue")
abline(a = 0, b = 1)
```



Roc Demo

- Area under the curve (AUC)

```
auc_ROCR <- performance(pred, measure = "auc")
auc_ROCR_lda_simple <- performance(pred_mylda, measure = "auc")
auc_ROCR@y.values[[1]]
```

```
## [1] 0.714415
```

```
auc_ROCR_lda_simple@y.values[[1]]
```

```
## [1] 0.6291602
```