Scientific programming: R, Rstudio and Rcpp

Agathe Guilloux
Professeure au LaMME - Université d'Évry - Paris Saclay

Goals of this lab

We will use the twoClass dataset from *Applied Predictive Modeling*, the book by Kuhn and Johnson 2013 to illustrate the some classical supervised classification algorithms.

We will use some *advanced* **R** packages:

- the **ggplot2** package for the figures and
- the **caret** package for the learning part. **caret** that provides an unified interface to many other packages.

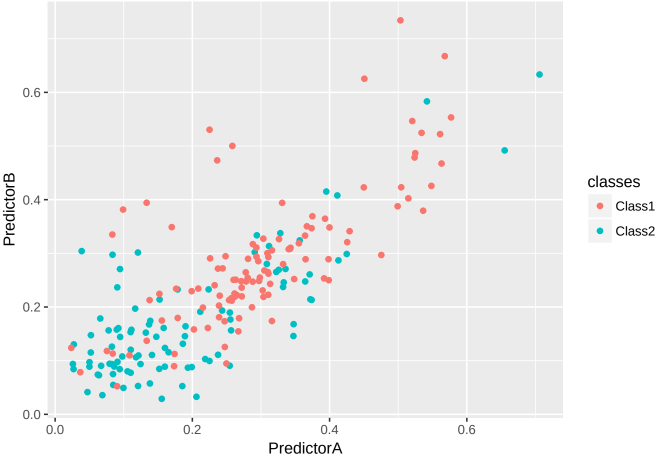
Goals:

- to review your R skills
- to review some classical classification algorithms
- to learn to use Rstudio and Rmarkdown
- to learn to create a R package
- and to use Rcpp to speed up your code.

All the material is on my webpage
http://www.math-evry.cnrs.fr/members/aguilloux/enseignements

The twoClass dataset

This is a synthetic dataset, which can be found in Kuhn and Johnson 2013 (or more simply in `AppliedPredictiveModeling` package).



Classification

Classification $=$ **supervised learning** with a binary label

Setting

- You have past/historical data, containing data about individuals $i = 1, \ldots, n$
- You have a **features** vector $x_i \in \mathbb{R}^d$ for each individual i
- For each i, you know if he/she clicked $(y_i = 1)$ or not $(y_i = -1)$
- We call $y_i \in \{-1, 1\}$ the **label** of i
- (x_i, y_i) are i.i.d realizations of (X, Y)

Aim

- Given a features vector x (with no corresponding label), predict a label $\hat{y} \in \{-1, 1\}$
- Use data $D = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ to construct a **classifier**

Probabilistic / statistical approach

- Model the distribution of $Y|X$
- Construct estimators $\hat{p}_1(x)$ and $\hat{p}_{-1}(x)$ of

$$p_1(x) = \mathbb{P}(Y = 1|X = x) \quad \text{and} \quad p_{-1}(x) = 1 - p_1(x)$$

- Given x, classify using

$$\hat{y} = \begin{cases} 1 & \text{if } \hat{p}_1(x) \geq t \\ -1 & \text{otherwise} \end{cases}$$

for some threshold $t \in (0, 1)$

Bayes formula. We know that

$$p_y(x) = \mathbb{P}(Y = y|X = x) = \frac{\mathbb{P}(X = x|Y = y)\mathbb{P}(Y = y)}{\mathbb{P}(X = x)}$$
$$= \frac{\mathbb{P}(X = x|Y = y)\mathbb{P}(Y = y)}{\sum_{y'=-1,1} \mathbb{P}(X = x|Y = y')\mathbb{P}(Y = y')}$$

If we know the distribution of $X|Y$ and the distribution of Y, we know the distribution of $Y|X$

Bayes classifier. Classify using Bayes formula, given that:

- We model $\mathbb{P}(X|Y)$
- We are able to estimate $\mathbb{P}(X|Y)$ based on data

Maximum a posteriori. Classify using the *discriminant* functions

$$\delta_y(x) = \log \mathbb{P}(X = x|Y = y) + \log \mathbb{P}(Y = y)$$

for $y = 1, -1$ and decide (largest, beyond a threshold, etc.)

Remark.

- Different models on the distribution of $X|Y$ leads to different classifiers
- The simplest one is the Naive Bayes
- Then, the most standard are Linear Discriminant Analysis (LDA) and Quadratic discriminant Analysis (QDA)

Naive Bayes

Naive Bayes. A crude modeling for $\mathbb{P}(X|Y)$: assume features X^j are independent conditionally on Y:

$$\mathbb{P}(X = x|Y = y) = \prod_{j=1}^{d} \mathbb{P}(X^j = x^j|Y = y)$$

Model the univariate distribution $X^j|Y$: for instance, assume that

$$\mathbb{P}(X^j|Y) = \text{Normal}(\mu_{j,k}, \sigma^2_{j,k}),$$

parameters $\mu_{j,k}$ and $\sigma^2_{j,k}$ easily estimated by MLE

- If the feature X^j is discrete, use a Bernoulli or multinomial distribution
- Leads to a classifier which is very easy to compute
- Requires only the computation of some averages (MLE)

Discriminant analysis

Discriminant Analysis. Assume that

$$\mathbb{P}(X|Y=y) = \text{Normal}(\mu_y, \Sigma_y),$$

where we recall that the density of $\text{Normal}(\mu, \Sigma)$ is given by

$$f(x) = \frac{1}{(2\pi)^{d/2}\sqrt{\det \Sigma}} \exp\Big(-\frac{1}{2}(x-\mu)^\top \Sigma^{-1}(x-\mu)\Big)$$

In this case, discriminant functions are

$$\begin{aligned}\delta_y(x) &= \log \mathbb{P}(X=x|Y=y) + \log \mathbb{P}(Y=y) \\ &= -\frac{1}{2}(x-\mu_y)^\top \Sigma_y^{-1}(x-\mu_y) - \frac{d}{2}\ln(2\pi) \\ &\quad - \frac{1}{2}\log\det \Sigma_y + \log \mathbb{P}(Y=y)\end{aligned}$$

Estimation. Use "natural" estimators, obtained by maximum likelihood estimation. Define for $y \in \{-1, 1\}$

$$I_y = \{i = 1, \ldots, n \ : \ y_i = y\} \quad \text{and} \quad n_y = |I_y|$$

MLE estimators are given by

$$\hat{\mathbb{P}}(Y = y) = \frac{n_y}{n}, \quad \hat{\mu}_y = \frac{1}{n_y} \sum_{i \in I_y} x_i,$$

$$\hat{\Sigma}_y = \frac{1}{n_y} \sum_{i \in I_y} (x_i - \hat{\mu}_y)(x_i - \hat{\mu}_y)^\top$$

for $y \in \{-1, 1\}$. These are simply the proportion, sample mean and sample covariance within each group of labels

Linear Discriminant Analysis (LDA)

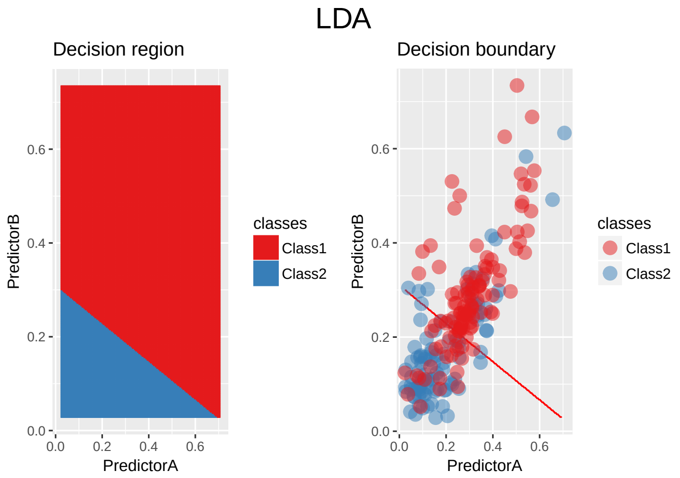
- Assumes that $\Sigma = \Sigma_1 = \Sigma_{-1}$
- All groups have the same correlation structure
- In this case decision function is linear $\langle x, w \rangle \geq c$ with

$$\begin{aligned} w &= \Sigma^{-1}(\mu_1 - \mu_{-1}) \\ c &= \frac{1}{2}(\langle \mu_1, \Sigma^{-1}\mu_1 \rangle - \langle \mu_{-1}, \Sigma^{-1}\mu_{-1} \rangle) \\ &\quad + \log\left(\frac{\mathbb{P}(Y = 1|X = x)}{\mathbb{P}(Y = -1|X = x)}\right) \end{aligned}$$

Quadratic Discriminant Analysis (QDA)

- Assumes that $\Sigma_1 \neq \Sigma_{-1}$
- Decision function is quadratic

Example: LDA



LDA
Decision region
Decision boundary
PredictorB
PredictorA
0.0
0.2
0.4
0.6
classes
Class1
Class2

Logistic regression

Logistic regression

- By far the most widely used classification algorithm
- We want to explain the label y based on x, we want to "regress" y on x
- Models the distribution of $Y|X$

For $y \in \{-1, 1\}$, we consider the model

$$\mathbb{P}(Y = 1 | X = x) = \sigma(x^\top w + b)$$

where $w \in \mathbb{R}^d$ is a vector of model **weights** and $b \in \mathbb{R}$ is the **intercept**, and where σ is the **sigmoid** function $\sigma(z) = \dfrac{1}{1 + e^{-z}}$

Compute $\hat{w}$ and $\hat{b}$ as follows:

$$(\hat{w}, \hat{b}) \in \operatorname*{argmin}_{w \in \mathbb{R}^d, b \in \mathbb{R}} \frac{1}{n} \sum_{i=1}^{n} \log(1 + e^{-y_i(\langle x_i, w \rangle + b)})$$

- It is a convex and smooth problem
- Many ways to find an approximate minimizer
- Convex optimization algorithms (more on that later)

If we introduce the **logistic loss** function

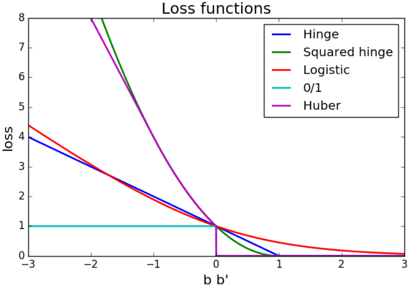
$$\ell(y, y') = \log(1 + e^{-yy'})$$

then

$$(\hat{w}, \hat{b}) \in \operatorname*{argmin}_{w \in \mathbb{R}^d, b \in \mathbb{R}} \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, \langle x_i, w \rangle + b)$$

Other classical loss functions for binary classication

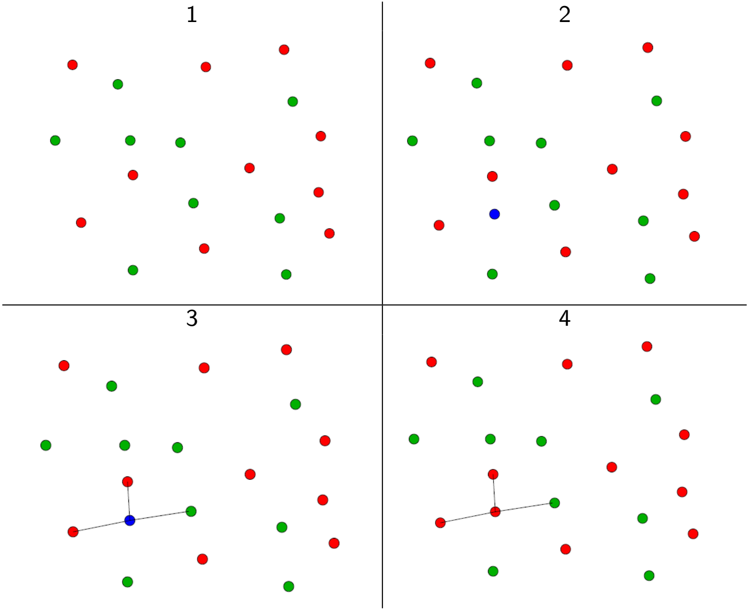
- Hinge loss (SVM), $\ell(y, y') = (1 - yy')_+$
- Quadratic hinge loss (SVM), $\ell(y, y') = \frac{1}{2}(1 - yy')_+^2$
- Huber loss $\ell(y, y') = -4yy'\mathbf{1}_{yy' < -1} + (1 - yy')_+^2 \mathbf{1}_{yy' \geq -1}$



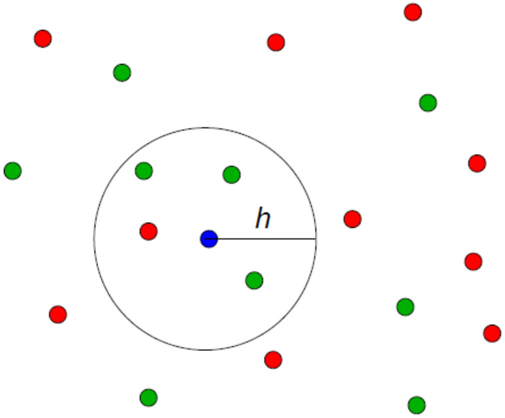
- These losses can be understood as a convex approximation of the 0/1 loss $\ell(y, y') = \mathbf{1}_{yy' \leq 0}$

k Nearest-Neighbors

Example: k Nearest-Neighbors (with $k = 3$) I



Example: k Nearest-Neighbors (with $k = 4$) I



k Nearest-Neighbors

- Neighborhood $\mathcal{V}_{\mathbf{x}}$ of $\mathbf{x}$: k closest from $\mathbf{x}$ learning samples.

k-NN as local conditional density estimate

$$\widehat{p}_{+1}(\mathbf{x}) = \frac{\sum_{\mathbf{x}_i \in \mathcal{V}_{\mathbf{x}}} \mathbf{1}_{\{y_i = +1\}}}{|\mathcal{V}_{\mathbf{x}}|}$$

- KNN Classifier:

$$\widehat{f}_{KNN}(\mathbf{x}) = \begin{cases} +1 & \text{if } \widehat{p}_{+1}(\mathbf{x}) \geq \widehat{p}_{-1}(\mathbf{x}) \\ -1 & \text{otherwise} \end{cases}$$

- **Remark:** You can also use your favorite kernel estimator...

Metrics

Confusion matrix

Definitions : Confusion matrix

For all individual $i = 1, \ldots, n$, define Y_i^P as the prediction (of Y_i). The confusion matrix is defined as

		Observed labels	
		$Y_i = -1$	$Y_i = 1$
Predictions	$Y_i^P = -1$	TN	FN
	$Y_i^P = 1$	FP	TP
	total	N	P

where P=POSITIVE, N=NEGATIVE, F=FALSE, T=TRUE.

Metrics from the confusion matrix

Define

- the **true positive rate or sensitivity or recall** as TP/P
- the **false discovery rate** as FP/(FP+TP)
- the **true negative rate or specificity** as TN/N
- the **false positive rate** as FP/(FP+TN)=FP/N = 1 - specificity
- the **precision** as
$$\frac{TP}{TP + FP}$$
- the **accuracy** as
$$\frac{TP + TN}{P + N}$$
- the False-Discovery-Rate (FDR) as 1−precision.

The ROC curve

To define the predictions (Y_i^P), we consider a $1/2$ threshold. Now, let the threshold varies from 0 to 1.

For each value of the threshold s, compute

- the true positive rate TPR_s
- the false-discovery-rate FDR_s.

The ROC curve and AUC

The ROC (receiver operating characteristic) curve is define as the curve

$$\Big\{(TPR_s, FDR_s), \forall s \in [0,1]\Big\}.$$

The AUC is the area under the ROC curve.

A classification rule constructed purely at random has an AUC of around 0.5.

Assigments

Your assigment

Due September 21, 2017

1. Send me a (complete...) html version of the part 1 of the Lab.
2. Send me a zip with your package. It **must** contain a working example (the .Rdm and .html files must be included in the zipped folder).

All files and zipped folder must be named after your name. For example, my file corresponding to the first assignment would be called **guilloux_TPpart1.html**.

Your homework

If your not familiar with these ML methods, see Friedman, Hastie, and Tibshirani 2001, available on https://web.stanford.edu/ hastie/Papers/ESLII.pdf

References I

Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*. Vol. 1. Springer series in statistics New York, 2001.

Max Kuhn and Kjell Johnson. *Applied predictive modeling*. Vol. 810. Springer, 2013.