Afi Escuela de Finanzas

Análisis Discriminante Lineal, Análisis Discriminante No Lineal, Naïve Bayes

Máster en Data Science y Big Data en Finanzas

Javier Nogales – PhD Matemáticas
Catedrático, Estadística e IO, UC3M
fcojavier.nogales@uc3m.es 2022

Objectives

- Develop skills for the main statistical tools in supervised learning
- Classify and predict observations based on parametric models (mainly Bayes-based classifiers): **probabilistic learning**
- Handle the R language for these tools

Organization

- Subject organized in 4 topics (QDA, LDA, Naïve Bayes, cost-sensitive learning)
- 8 hours in total: 2 sessions
- Practical course: 50% basic concepts + 50% computer labs (using R)
- Evaluation: 100% final exercise

Supervised Learning

- Take inputs $x_1, x_2, \ldots, x_k$ and map them to an output y
  - Statistical learning uses probability assumptions to find this map or function
  - Machine learning does not use probabilities, just the data
- To know whether the map is working correctly, we need some kind of metric to measure performance
  - Loss function: count number of times the model is working wrong (classification), or measure how close the prediction is to y (regression)
- The objective is to **minimize the loss**

Supervised Learning

Some notation:

- Output y: target, dependent variable, response, ...
- Input $x_1, x_2, \ldots, x_k$: predictors, features, regressors, covariates, independent variables, ...
- Data organized by rows: observations, samples, examples, instances, ...

One framework, but mainly two categories:

- Regression
- Classification

Supervised Learning Framework

- Usual framework in **Machine Learning / Statistics**:

$$\text{Data} = \text{Model} + \text{Noise}$$

- When we focus on one variable predicted by others: **supervised learning**

$$y = g(x_1, \ldots, x_k) + \text{Noise}$$

  - Statistical (linear) approximation: $y = \beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p + \epsilon$
  - Machine learning approximation: $y = \text{map}(x_1, \ldots, x_p) + \epsilon$

- Note the variables in the true model, g, may be different in the approximations

- Two main categories:
  - Classification: y is categorical
  - Regression: y is numeric (usually continuous)

Supervised Learning: Classification

- Traditional machine-learning classification: process to predict a categorical/qualitative variable (*response/target*) from various inputs (*predictors*)

  Example: predict whether tomorrow will rain or not

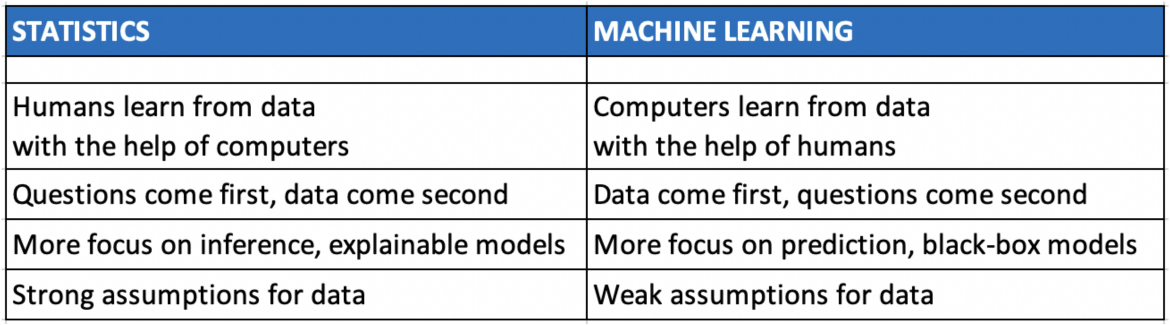
  Focus is on prediction, hence **better prediction** performance

- Probabilistic learning: the same but first predict the probability of each category, then predict the category

  Example: tomorrow there will be 70-percent chance of rain

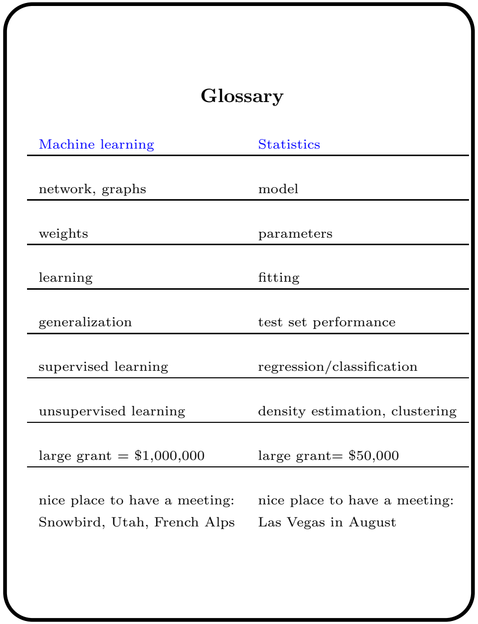
  Focus is on explanation + inference, hence **better understanding** (about relationship between the response and the predictors)

Statistics vs Machine Learning



STATISTICS	MACHINE LEARNING
Humans learn from data with the help of computers	Computers learn from data with the help of humans
Questions come first, data come second	Data come first, questions come second
More focus on inference, explainable models	More focus on prediction, black-box models
Strong assumptions for data	Weak assumptions for data

Statistics vs Machine Learning (by R. Tibshirani)



Glossary

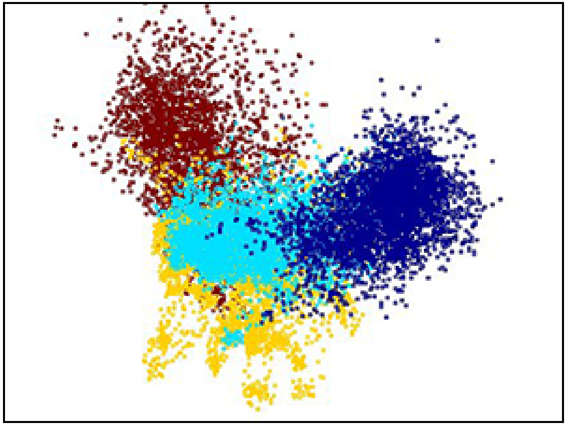
Machine learning	Statistics
network, graphs	model
weights	parameters
learning	fitting
generalization	test set performance
supervised learning	regression/classification
unsupervised learning	density estimation, clustering
large grant = \$1,000,000	large grant= \$50,000
nice place to have a meeting: Snowbird, Utah, French Alps	nice place to have a meeting: Las Vegas in August

Classification

- We have a data matrix $X \in \mathbb{R}^{n \times p}$ for the predictors or features. The observations in X come from different known sub-populations (categories)
- We know a sample of well-classified elements that will allow to classify new observations
- We will use *labels* y to identify the categories
- Main question: if we have a new observation, can we predict to which category it will belong?

Classification

- Four groups, two variables, thousands of observations



- For a new observation, can we predict to which group it will belong?

Classification

- Many applications:
  - Credit scoring, credit risk
  - Stock trading
  - Electronic fraud detection
  - Spam filtering or text classification
  - Diagnosing medical conditions
  - Churn Analytics
  - Pattern recognition: voice recognition, text classification, image recognition

Classification

- Statistical Classification:
  - Logistic Regression
  - Discriminant Analysis: Bayes classifiers
    - LDA
    - QDA
    - Naïve Bayes
    - Shrinkage classification
- Machine-Learning Classification:
  - Nearest Neighbors
  - Neural Networks
  - SVMs
  - Decision trees, Random Forests, Gradient Boosting, . . .

Probabilistic Learning

Mainly two families:

- Logistic Regression
- Discriminant Analysis: Bayes Classifiers

Classification: Logistic Regression

Remember:

- In logistic regression, if we have 2 groups (0 and 1), we model the conditional distribution of the response given the predictors: $p = p(y = 1|x_1, \ldots, x_p)$
- Assuming that the logit is linear: $\text{logit}(p) = \log \frac{p}{1-p} = \beta_0 + \beta^T x$
- Hence, $p = \frac{\exp(\beta_0+\beta^T x)}{1+\exp(\beta_0+\beta^T x)} = \frac{1}{1+\exp(-\beta_0-\beta^T x)}$
- Logistic regression provides a linear classification rule: for a probability threshold 0.5, the decision boundary separating the two predicted classes is the solution of $\hat{\beta}_0 + \hat{\beta}^T x = 0$

Classification: Bayes Classifiers

- **Probabilistic learning:** instead of focusing on the conditional mean of of the response given the predictors, $y|x_1, \ldots, x_p$, focus on the conditional probability: $p(y|x_1, \ldots, x_p)$
- In logistic regression, this conditional probability is modelled directly
- In Bayes classifiers, the conditional probability is modelled in a different and less direct way:
  - First, we model the predictors X separately for each given class y: $p(x|y)$
  - Then, we apply the Bayes formula to get $p(y|x)$
- This idea is more stable than logistic regression when the classes are well-separated
- It is popular when we have more than two classes
- Bayes classifiers perform well if variables are somehow Gaussian
  Note this implies the predictors should be numeric, but not mandatory

A review: Bayes' Theorem

- $P(A|B)$ probability of A given B is true/known, also called **posterior probability**
- In practice, A is an unknown event and B is known data/information
- $P(A)$ is the **prior probability** (what we know about A independently of B) before collecting data
- $P(B)$ is called the marginal distribution (independent of all events)
- $P(B|A)$ is the conditional probability of B knowing A, also known as the likelihood (of the data given A is true)
- Bayes' Theorem:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

- In many cases, we have several mutually disjoint events: $A_1, \ldots, A_K$. In that case,

$$P(B) = \sum_k P(B \cap A_k) = \sum_k P(B|A_k)P(A_k)$$

A review: Bayes' Theorem

Example: drug test

- 0.4% of a given population use some type of drug
- Conventional drug test: produce 99% true positives for drug users and 99% true negatives for non-drug users
- If one individual is tested positive, what is the probability of having used the drug?
- Bayes' rule:

$$P(\text{user}|+) = \frac{P(+|\text{user})P(\text{user})}{P(+|\text{user})P(\text{user}) + P(+|!\text{user})P(!\text{user})} = \frac{0.99 \times 0.004}{0.99 \times 0.004 + 0.01 \times 0.996} = 28.4\%$$

- Very small...why?

Bayes Classifiers

- Imagine we have G groups, and the output can take G distinct values, coded as $g = 1, \dots, G$
- Let $\pi_g = P(y \in g)$ denote the prior probability (known), for $g = 1, \dots, G$, that an observation belongs to group g
- Note previous probabilities are independent of predictors X
- First, model the (multivariate) distribution of predictors for each g: $f_g(x)$

Bayes Classifiers

- Then, apply Bayes Theorem to obtain the posterior probabilities given a new observation x:

$$p_g(x) = P(y \in g \mid X = x) = \frac{f_g(x)\pi_g}{\sum_k f_k(x)\pi_k}$$

- Finally, apply Bayes' rule: assign new observation x to that group with largest $p_g(x)$, i.e. $\max_g \; f_g(x)\pi_g$

- Note that $p_1(x) + \cdots + p_G(x) = 1$

- The posterior probabilities are the same as in logistic regression but estimated in a different way

Bayes Classifiers

- Bayes classifiers are optimal in the sense they minimize the classification error rate
- This is just in theory: we need to know exactly, without any error, $f_g(x)$ and π_g for each g
- In practice, we just approximate or estimate $f_g(x)$ and π_g
- For instance, π_g can be estimated using the proportion of training observations that belong to class g: $\hat{\pi}_g = n_g/n$
- But without previous knowledge, $\hat{\pi}_g = 1/G$ is a good choice
- Next, we will see some useful approximations for $f_g(x)$ in practice

Bayes Classifiers

- If each $f_g(x)$ is assumed to be multivariate normal, $f_g \sim \mathcal{N}_p(\mu_g, \Sigma_g)$, then
- Bayes rule: $\max_g \; f_g(x)\pi_g \equiv$

$$\max_g \quad \pi_g \, (2\pi)^{-p/2} \, \det(\Sigma_g)^{-1/2} \, \exp(-0.5(x-\mu_g)^T \Sigma_g^{-1}(x-\mu_g))$$

  which is equivalent (taking logs) to:

$$\min_g \quad (x-\mu_g)^T \Sigma_g^{-1}(x-\mu_g) + \log\det(\Sigma_g) - 2\log\pi_g$$

- This is called Quadratic Discriminant Analysis (QDA)
- We can estimate parameters using the training set, but may be too complex in high dimension (p/n large): we need to estimate G vector of means and G covariance matrices
- In practice, it is an unbiased estimator with high variance

Classification: QDA

- Assuming the data matrix X is grouped in G submatrices, each with n_g observations, parameters are estimated as follows: $\hat{\mu}_g = \bar{x}_g$, $\hat{\Sigma}_g = S_g$, and $\hat{\pi}_g = \frac{n_g}{n}$

- If we do not have prior knowledge about π_g, then assume $\pi_1 = \cdots = \pi_G$, and focus on

$$\min_g \quad (x - \bar{x}_g)^T S_g^{-1} (x - \bar{x}_g) + \log\det(S_g)$$

- Note QDA may be too complex in high dimension (p large): from n observations, we need to estimate G vector of means and G covariance matrices

- Moreover, some S_g could be singular (or close to)

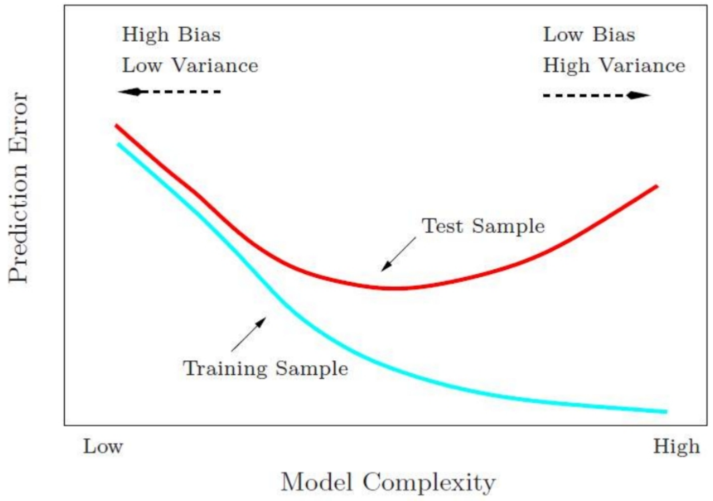
Classification: QDA

- For these reasons, QDA is relatively unstable in practice (except for very large samples and low dimension)
- Moreover, it is very sensitive to deviations from Gaussian distribution
- Hence, convenient to impose restrictions in higher dimension
- Linear discriminant analysis (LDA): simplified version, much less parameters

$\longrightarrow$ less variance but higher bias than QDA

Classification: simple models

When we simplify: we introduce some bias to reduce variance (less prediction error)



Classification: LDA

- Idea: introduce some bias to reduce variance (less prediction error)
- The most common restriction is to assume all covariance matrices, Σ_g, are equal. Then, the only covariance matrix to be estimated is $\hat{\Sigma} = S_w = \sum_{g=1}^{G} (\frac{n_g - 1}{n - G}) S_g$
- Bayes rule: $\max_g \; f_g(x)\pi_g \equiv$

$$\min_g \quad (x - \bar{x}_g)^T S_w^{-1} (x - \bar{x}_g) - 2 \log \hat{\pi}_g$$

- This is indeed a linear rule (in x), equivalent to

$$\min_g \quad 2\bar{x}_g^T S_w^{-1} x - \bar{x}_g^T S_w^{-1} \bar{x}_g - 2 \log \hat{\pi}_g$$

- This is called Linear Discriminant Analysis (LDA)

Classification: LDA

- It is linear because the terms $x^T S_w^{-1} x$ are constant for all groups
- Similar to the linear rule in logistic regression:
  - If we have 2 groups (0 and 1), $\text{logit}(p) = \log \frac{p}{1-p} = \beta_0 + \beta^T x$
  - In LDA, $\text{logit}(p) = (\mu_1 - \mu_0)^T \Sigma^{-1} x + \log \frac{\pi_0}{\pi_1} = \alpha_0 + \alpha^T x$
  - Both approaches give linear logit, but they estimate parameters in different ways

Classification: LDA

- If we do not have prior knowledge about π_g, then assume $\pi_1 = \cdots = \pi_G$, and focus on

$$\min_g \quad (x - \bar{x}_g)^T S_w^{-1} (x - \bar{x}_g)$$

- I.e., classify x in the population whose mean is closest (using Mahalanobis distance)

LDA

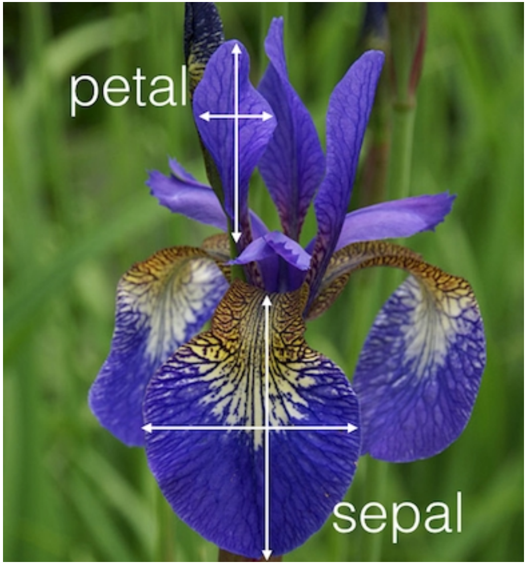
Some recommendations:

- LDA can be sensitive to outliers: hence, try to identify and remove them previously
- LDA assumes all groups have the same covariance matrix: hence, standardize the variables previously
- LDA assumes predictor distribution is multivariate gaussian: hence, transform the predictors previously to make them somehow symmetric
- For binary classification, better logistic regression (unless very well-separated groups)
- For multi-class classification, better LDA (unless very-unbalanced groups)

Classification: Example

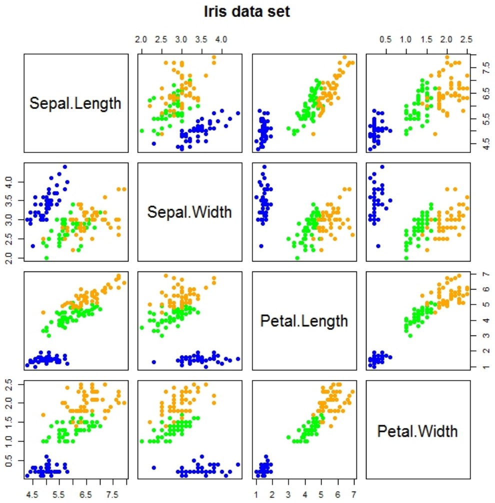
- Fisher’s Iris data set: best known data set in classification
- Data collected by Edgar Anderson in the 30’s to quantify the morphologic variation of Iris flowers of three related species
- It consists of 150 Iris flowers of three different species: setosa, versicolor, virginica
- Four types of measurements for each flower, the length and width of sepals and petals in centimeters, respectively
- Popularized by Ronald Fisher in 1936
- Based on the combination of the four features, Fisher developed a linear discriminant model to distinguish the species from each other

Iris Flower



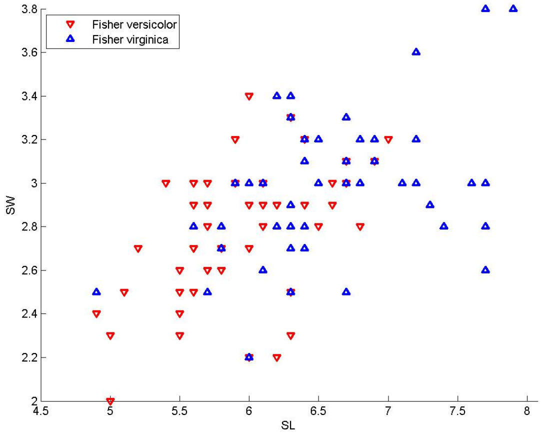
Iris Scatter Plot

In blue, setosa Iris; in green, versicolor Iris; in orange, virginica Iris



Classification: Example

- Iris dataset: focus first on two species (versicolor and virginica)



- And consider just two features (length and width of sepals)

Classification: performance measures

- Confusion matrix:

	Classify in P_1	$\cdots$	Classify in P_G
Belongs to P_1	n_{11}	$\cdots$	n_{1G}
$\vdots$	$\vdots$	$\ddots$	$\vdots$
Belongs to P_G	n_{G1}	$\cdots$	n_{GG}

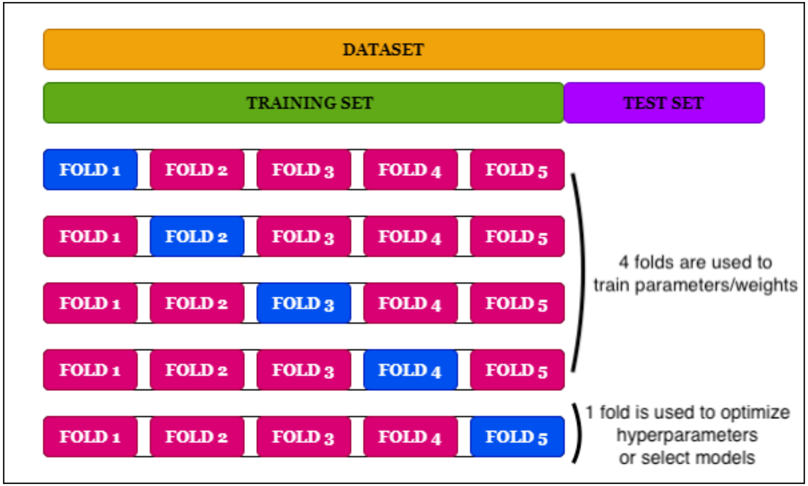
where n_{ij} is the amount of observations coming from group P_i and classified in group P_j

- The **error** is then: $\frac{n_{12}+\cdots+n_{G,G-1}}{n}$, the sum of off-diagonal elements

- The **accuracy** is the sum of the diagonal terms

- The confusion matrix should be computed using cross-validation or testing sets (not training ones)

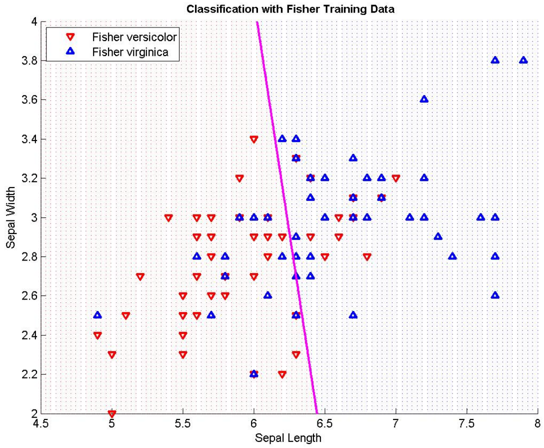
Performance measures


DATASET
TRAINING SET
TEST SET
FOLD 1
FOLD 2
FOLD 3
FOLD 4
FOLD 5
4 folds are used to train parameters/weights
1 fold is used to optimize hyperparameters or select models

In the validation or testing set is where we use performance measures like: Accuracy, Kappa, Sensitivity (TPs), Specificity (TNs), etc.

Example

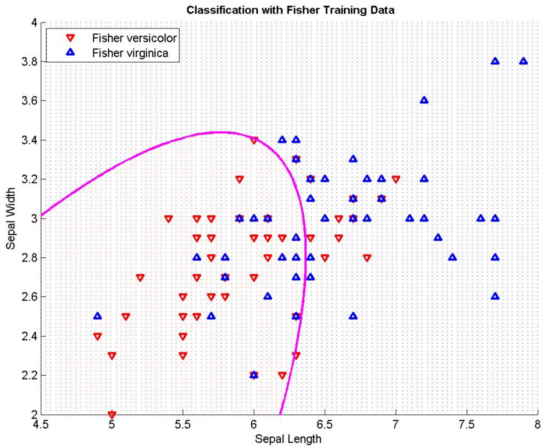
- LDA classifier:



- In-sample error = 25%

Example

- QDA classifier:



- In-sample error = 29%

Example

- Iris dataset: consider now the three species and the four features
- In-sample error for LDA: 2%
- CV error for LDA: 2%
- Out-of-sample error for LDA: 2.3%

- In-sample error for QDA: 2.7%
- CV error for QDA: 2.7%
- Out-of-sample error for QDA: 2.7%

Naïve Bayes

- Another restriction, successful in high dimension, is to assume variables (features) are independent
- This is called Naïve Bayes Classification
- It assumes: $f_g(x) = f_g(x_1) \times f_g(x_2) \times \cdots \times f_g(x_p)$
- Hence, there are many versions depending on distribution f_g:
  Gaussian, kernel, multinomial, bernoulli, etc.
- It is easy to build and particularly useful for very large data sets
- Widely used in text classification (high dimensional training data sets): spam filters, sentimental analysis, news articles classification, etc.

Naïve Bayes

- Gaussian Naïve Bayes: useful for continuous predictors
    - Linear naïve Bayes:
      Impose the pooled covariance matrix is diagonal:

$$\hat{\Sigma} = \text{diag}(S_w)$$

    - Quadratic naïve Bayes:
      Impose each covariance matrix is diagonal:

$$\hat{\Sigma}_g = \text{diag}(S_g), \quad g = 1, \ldots, G$$

Naïve Bayes

- Binomial Naïve Bayes: useful for discrete counts
  - number of times outcome number x_i is observed over the n trials
  - used in text classification word count vectors
- Bernoulli Naïve Bayes: useful for binary predictors
  - used in text classification word occurrence vectors
- Categorical Naïve Bayes: useful for categorical features
  - we should label-encode the features with dummies

Naïve Bayes

General comments:

- Easy and fast to predict and understand: very large datasets
- Perform well in multi-class prediction, especially when we have a large number of categorical predictors
- But depends heavily on independence of predictors
- Posterior probabilities not reliable
- Mainly used in: text classification (spam filters), sentiment analysis (is text negative, positive or neutral?), and recommendation systems (would a user like a given product?)

Shrinkage Methods in Classification

- If the dimension is large, then better to use LDA than QDA
- And better to use Naïve Bayes than LDA
- Shrinkage classification is a way to combine above ideas:
  - $\hat{\Sigma}_g(\lambda) = (1 - \lambda)S_g + \lambda S_w$, where $0 \leq \lambda \leq 1$
    (combine LDA and QDA)
  - $\hat{\Sigma}_g(\alpha, \lambda) = (1 - \alpha)\hat{\Sigma}_g(\lambda) + \alpha \nu_\lambda I$, where $0 \leq \alpha \leq 1$
    and $\nu_\lambda = \text{tr}(\hat{\Sigma}_g(\lambda))/p$
    (and now combine with naive)
- Parameters α and λ need to be optimized to improve performance

Bayes classifiers: advanced tools and packages

- Factor-Based Linear Discriminant Analysis: HiDimDA
- Linear Discriminant Analysis with Stepwise Feature Selection: klaR, MASS
- Penalized Discriminant Analysis: mda, penalizedLDA, klaR, rlda, sda, sparsediscrim
- Sparse Linear Discriminant Analysis: sparseLDA, sparsediscrim
- Naive Bayes: naivebayes, klaR,

Bayes classifiers: pros and cons

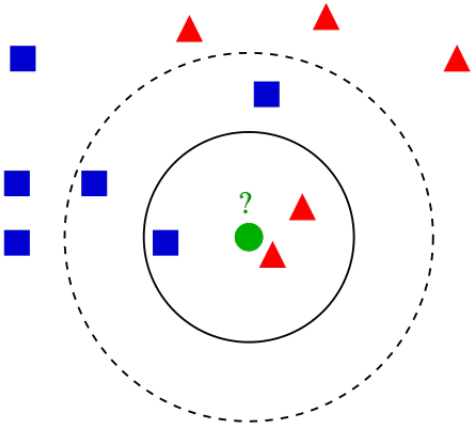
- Bayes classifiers are optimal in the sense they minimize the classification error (theoretically)
- Work well if variables are somehow symmetric and unimodal (somehow linearly separable)
- Then, we need to help the tools: transform the features and standardize the variables
- For binary classification, better logistic regression (unless very well separated groups). For multiclass classification, better LDA (unless very unbalanced groups)
- If dataset is clearly non-linear, use ML: but no closed-form solution and hyper-parameters appear
- To avoid collinearity problems, use advanced versions: sparse, shrinkage, etc.
- They are sensitive to outliers: try to identify them first and remove

k-Nearest Neighbors

k-Nearest Neighbors

- Simple method with good performance for highly non-linear data and moderate p/n
- The k-NN algorithm is as follows:
  - Define a distance between observations (Euclidean, Mahalanobis, Manhattan, . . .)
  - Compute the distance between the new observation to classify, x, and all the observations in the sample
  - Select the k closest observations to x and compute the proportion of them that belongs to each group
  - Then, classify x with the largest proportion (mode)

k-NN for classification



Classification: k-NN

- This method is a data-mining tool: no assumptions needed
- But it cannot tell us which predictors are important
- Well-suited for highly non-linear data
- But bad performance if the dimension, p, is high
- The key element in k-NN is the selection of k (neighbors)
- Usually, cross-validation is used to select k with less prediction error (bias-variance tradeoff)
- Preferable to try odd numbers for k

Bayes Rule and Cost-Sensitive Learning

Probabilistic Learning: Bayes rule

- Bayes rule is optimal in the sense it minimizes the overall error rate, i.e. the number of elements off the diagonal in the confusion matrix (if the underlying assumptions are true, or in the training set)

- Hence, we are assuming all the classification errors are equally important
  But what happens if we are not interested in minimizing such overall error?

- **Unbalanced classes** and/or **cost-sensitive learning**

Probabilistic Learning: unbalanced classes

How to deal with unbalanced classes?

- Adjust prior probabilities, π_g: instead of using the proportions found in the training set, be more conservative (shrink towards 0.5)
- Change threshold in Bayes rule (0.5 in binary problems) for posterior probabilities
- Use other performance measures (rather than accuracy and error rate)
- Cost-sensitive learning
- Sub-sampling (for instance sampling more from minority class)

Probabilistic Learning: unbalanced classes

Performance measures, especially useful when 2 classes are unbalanced:

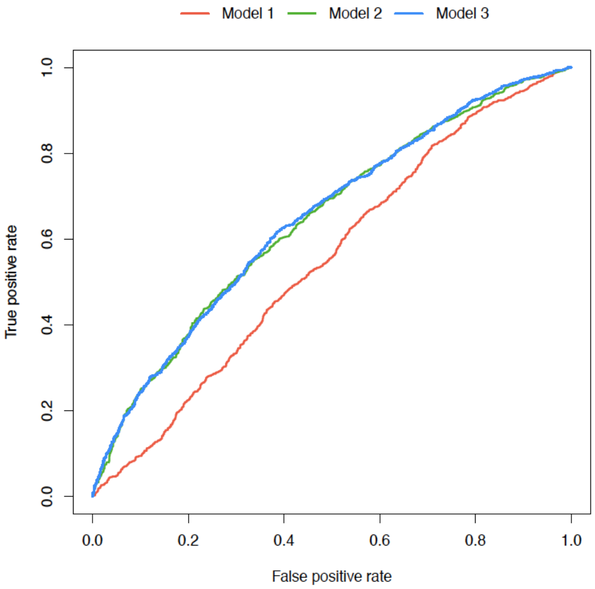
	Predicted	
Actual	yes	no
yes	TP	FN
no	FP	TN

- kappa: adjusts accuracy by the possibility of a correct prediction obtained by chance. The greater the better
- sensitivity: true positive rate, TP/(TP+FN). The greater the better
- specificity: true negative rate, TN/(TN+FP). The greater the better
- precision: positive predictive value, TP/(TP+FP). The greater the better
- recall: TP/(TP+FN) = sensitivity, but used with different interpretation. The greater the better
- F1 = 2*precision*recall/(precision+recall). The greater the better. Good measure to compare models

Probabilistic Learning: ROC curves

- Used for binary problems by varying the cutoff to assign groups (Bayes' rule)
- Compare false positive rates, $P(\hat{y} = 1 | y = 0)$, with true positive ones, $P(\hat{y} = 1 | y = 1)$
- The true positive rate is also called the sensitivity and 1 minus the false positive rate is also called the specificity
- The receiver operating characteristic (ROC) curve is created by varying the cutoff from 0 to 1

ROC curves


Model 1
Model 2
Model 3
True positive rate
False positive rate
0.0
0.2
0.4
0.6
0.8
1.0

AUC

- In previous graph, models 2 and 3 have higher true positive rates than model 1
- But difficult to compare models 2 and 3
- A useful summary of the overall quality is the area under the curve (AUC)
- An AUC of 0.5 indicates a random guessing while an AUC of 1 indicates perfect classification

Probabilistic Learning: cost-sensitive learning

- Cost matrix:

	Predicted	
Actual	yes	no
yes	$-c_{11}$	c_{12}
no	c_{21}	$-c_{22}$

- Bayes rule is optimal when $c_{11} = c_{22} = 0$ and $c_{21} = c_{12}$
- In many applications, the losses from classification errors are unbalanced: $c_{11} \neq c_{22}$ and $c_{21} \neq c_{12}$
- The ROC curve (Receiver Operating Characteristic) can also help: tradeoff between true positives and false positives. The closer the curve to the top-left position, the better
- Cost-sensitive learning implies domain knowledge for the specific application, as we can reduce one specific error rate but at the cost of increasing the others
- Only a few packages in Caret, mainly for some SVMs (e1071) and trees (C5.0 and CART)

Probabilistic Learning: final remarks

- Logistic regression performs well for binary classification, and it allows categorical predictors (features). Recommendable to regularize to avoid overfitting
- For more than two groups, LDA performs better if predictors are somehow Gaussian
- In Bayes classifiers, feature selection should be done (to choose the important features) or shrinkage
- Validate conveniently the classification tools (using testing sets) and consider always misclassification losses
- Try as many classifications tools as you can: SVMs, RFss, kNN, etc.
- SVM is indeed similar to logistic regression with shrinkage

Objectives

- Develop skills for the main statistical tools in supervised learning
- Classify and predict observations based on parametric models (mainly Bayes-based classifiers): **probabilistic learning**
- Handle the R language for these tools

“It is much more interesting to live with uncertainty than to live with answers that might be wrong.”

Richard Feynman, Nobel laureate in Physics

Afi Escuela de Finanzas