



UNIVERSITY OF  
SAN FRANCISCO

Master of Science  
in Analytics

---

# Classification

Machine Learning 1

---

---



# Classification

- Response (Y) is qualitative — i.e. an enumerated class
- Examples:
  - Which candidate will win the 2016 presidential election?
  - What language is the following: 狗不喜欢吃蔬菜
- Algorithm grab bag
  - Now: Logistic Regression, Linear Discriminant Analysis (QDA), K Nearest Neighbours
  - Later: generalised additive models, trees, random forests, SVM
- Why not modify regression?
  - Yes, it's possible to enumerate classes and perform regression
  - But it's inadvisable
    - Ordering of classes may not be “natural”
    - Regression may predict value outside enumerated range



# Logistic Regression

- Given features, determine the probability that Y belongs to one of the defined categories
- Shares same theory as linear regression:
  - $p(X) = \beta_0 + \beta_1 X$
  - Use **maximum likelihood** b/c probabilities must be in interval [0 .. 1]:

$$p(X) = \frac{e^{\beta_0 + \beta_1 X}}{1 + e^{\beta_0 + \beta_1 X}}$$

- Maximum likelihood is a manipulation of (log) odds / SLR basis:

- Odds

$$\frac{p(X)}{1 - p(X)} = e^{\beta_0 + \beta_1 X}$$

- Log odds

$$\log \left( \frac{p(X)}{1 - p(X)} \right) = \beta_0 + \beta_1 X.$$



# Example: Default data

- Data from [ISLR dataset](#), including a person's:
  - Default status ("Yes" or "No")
  - Student status ("Yes" or "No")
  - Balance
  - Income
- Unbalanced data set
  - Total = 10K instances
  - Most people (96.67%) do not default (3.33% baseline error rate)
  - Bimodal income amounts
- Bayesian



# Implementation in weka

- 1) Coerce the data into [ARFF format](#) (eg. [Default](#))
- 2) In weka:
  - a) Explorer > [ Preprocess ] > Open file...
  - b) [ Classify ] > [ Choose ] (function)
  - c) Test options (test set / Cross-validation)

Default dataset

Weka Explorer

Classifier: Choose **Logistic -R 1.0E-8 -M -1 -num-decimal-places 4**

Test options:

- ☐ Use training set
- ☐ Supplied test set Set...
- ☒ Cross-validation Folds **10**
- ☐ Percentage split % **66**

More options...

(Nom) default

Start Stop

Result list (right-click for options)

17:37:32 - functions.Logistic

Classifier output:

=== Stratified cross-validation ===  
=== Summary ===

Correctly Classified Instances	9732	97.32 %
Incorrectly Classified Instances	268	2.68 %
Kappa statistic	0.4278	
Mean absolute error	0.0428	
Root mean squared error	0.1461	
Relative absolute error	66.4514 %	
Root relative squared error	81.4547 %	
Total Number of Instances	10000	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
Weighted Avg.	0.996	0.685	0.977	0.998	0.986	0.467	0.949	0.998	No
	0.315	0.004	0.724	0.315	0.439	0.467	0.040	0.528	Yes
	0.973	0.662	0.968	0.973	0.968	0.467	0.949	0.982	

=== Confusion Matrix ===

a	b	<-- classified as
9627	40	a = No
228	105	b = Yes

Status: OK

2.68% classification error

=== Confusion Matrix  
===

a b <--  
classified as  
9627 40 | a = No  
228 105 | b = Yes

Sensitivity / Specificity

F-Measure

0.986	...	No
0.439	...	Yes

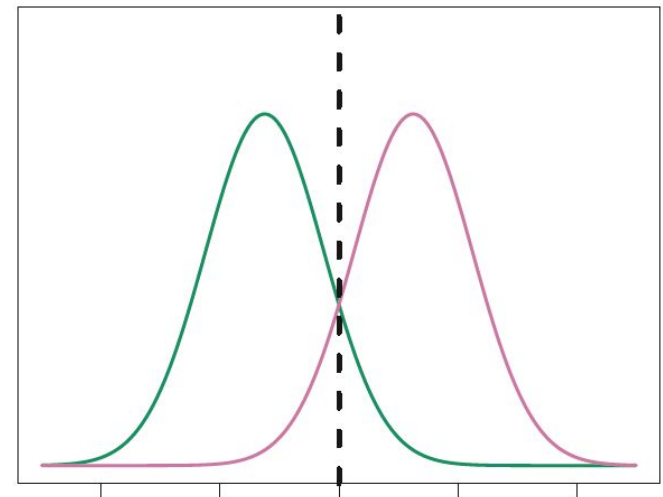


# Linear Discriminant Analysis

- Useful when:
  - Number of classes,  $K \geq 1$
  - Classes are well-separated
  - Distribution of each class is approximately normal
- Bayesian basis:

$$\Pr(Y = k|X = x) = \frac{\pi_k f_k(x)}{\sum_{l=1}^K \pi_l f_l(x)}$$

- $\pi_k$ : Prior probability
- $f_k(X) = \Pr(X=x | Y=k)$ : Density function
- Intuitively, looking for a class separator:





# Implementation in scikit-learn

- 1) Import data
  - a) If necessary, split data into train, test sets
- 2) Coerce data into:
  - a) test\_x, train\_x = List-of-lists / numpy matrix: all features
  - b) test\_y, train\_y = List / numpy vector: all targets

# 3) LDA

```
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
```

```
prior_vector = [0.2, 0.8]
```

```
algo = LinearDiscriminantAnalysis (priors=prior_vector) # "priors" vector is optional  
algo.fit (train_x, train_y)  
hypotheses = algo.predict (test_x)
```

- 4) Perform analysis on hypotheses

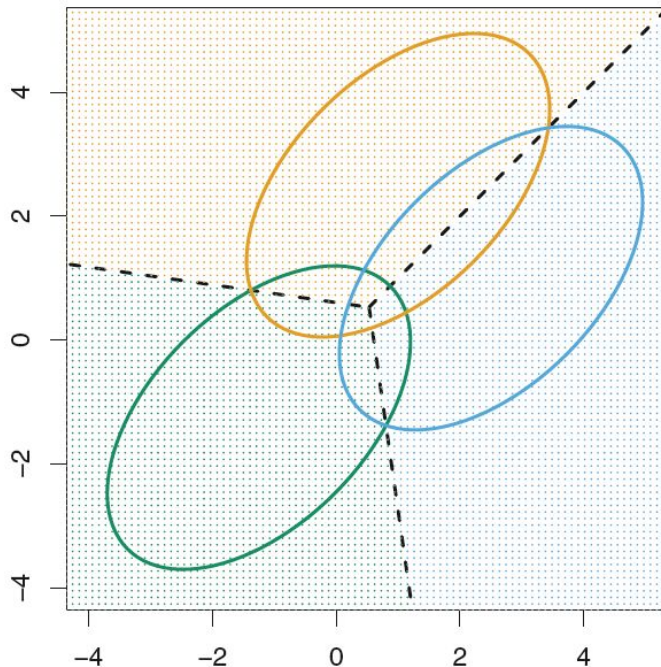


# LDA Decision Boundary

- Class separator is formally called “Decision Boundary”

$$\delta_k(x) = x \cdot \frac{\mu_k}{\sigma^2} - \frac{\mu_k^2}{2\sigma^2} + \log(\pi_k)$$

- Calculation is tractable for  $p > 1$  &  $k > 2$

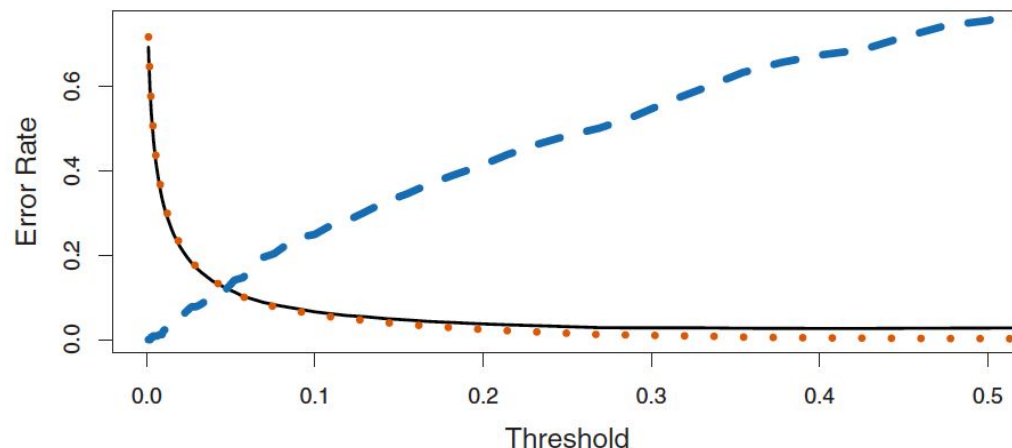




# When you care about minorities...

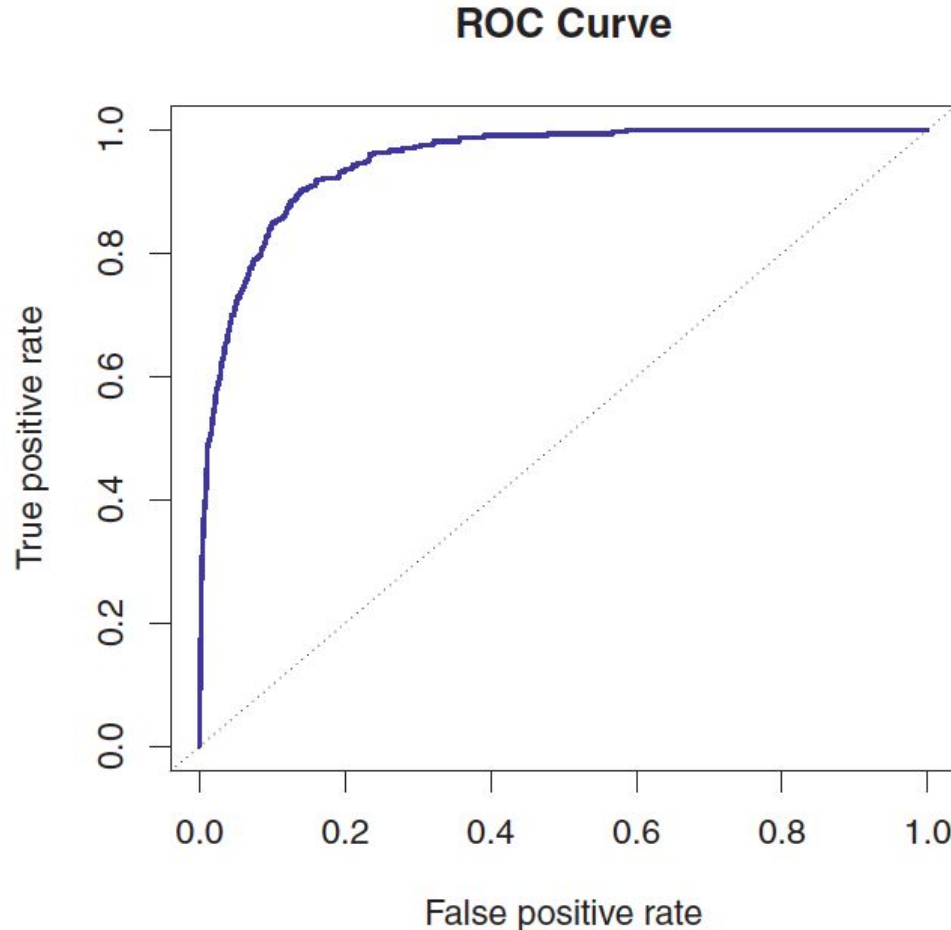


- Sometimes minority performance is more important:
  - Disease detection
  - Credit card fraud
- Many functions increase performance this way
  - Assigns observations to class with highest posterior probability
  - Default 50% (or higher) will be assigned to majority class
  - Can change that threshold; see improvement in confusion matrix





# Making the tradeoffs visible



- ROC curve
  - Shows both errors for all possible thresholds
  - Defines AUC = .95?
  - Curve ideally hugs top left corner



# Non-linear classification

- Quadratic Discriminative Analysis

- Relaxes assumptions:
  - Each class has a unique (Gaussian) distribution  $(\mu_k, \Sigma_k)$
  - Assigns label which is the max of:

$$\begin{aligned}\delta_k(x) &= -\frac{1}{2}(x - \mu_k)^T \Sigma_k^{-1}(x - \mu_k) + \log \pi_k \\ &= -\frac{1}{2}x^T \Sigma_k^{-1}x + x^T \Sigma_k^{-1}\mu_k - \frac{1}{2}\mu_k^T \Sigma_k^{-1}\mu_k + \log \pi_k\end{aligned}$$

- Quadratic?
  - Note “x” is quadratic function
  - Class boundaries may be non-linear

- K Nearest Neighbours

- Assigns label according to majority (plurality) of neighbours
- Can produce highly non-linear boundaries