



Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

Other Issues

# Processamento e Recuperação de Informação

## Classification

Departamento de Engenharia Informática  
Instituto Superior Técnico

1º Semestre  
2018/2019



# Outline

Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

Other Issues

- 1 Introduction
- 2 Supervised Learning
- 3 Text Classifiers
  - Nearest Neighbor Classifiers
  - Generative Bayesian Classifiers
  - Linear Discriminative Classifiers
  - Neural Networks
- 4 Other Issues



# Bibliography

Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

Other Issues

- Bing Liu, Web Data Mining - Exploring Hyperlinks, Contents, and Usage Data. Chapter 3.
- Ricardo Baeza-Yates and Berthier Ribeiro-Neto, Modern Information Retrieval. Chapter 8.
- Christopher D. Manning, Prabhakar Raghavan and Hinrich Schütze, Introduction to Information Retrieval. Chapters 13, 14 and 15.
- Jure Leskovec, Anand Rajaraman, and Jeff Ullman, Mining of Massive Datasets, Chapter 12



# Outline

Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

Other Issues

## 1 Introduction

## 2 Supervised Learning

## 3 Text Classifiers

## 4 Other Issues



# Organizing Knowledge

Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

Other Issues

- Organize into **systematic knowledge structures**
- Ontologies
  - Dewey Decimal System
  - ACM Computing Classification System
  - Patent Subject Classification
  - International Classification of Diseases
- Web catalogs
  - Yahoo Directory (RIP 2002–2014)
  - DMOZ Directory (RIP 1998–2017)
  - World Wide Web Virtual Library
  - Jasmine Directory



# Organizing Knowledge

Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

Other Issues

- Organize into **systematic knowledge structures**
- Ontologies
  - Dewey Decimal System
  - ACM Computing Classification System
  - Patent Subject Classification
  - International Classification of Diseases
- Web catalogs
  - Yahoo Directory (RIP 2002–2014)
  - DMOZ Directory (RIP 1998–2017)
  - World Wide Web Virtual Library
  - Jasmine Directory

Problem: Manual maintenance



# Outline

Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

Other Issues

1 Introduction

2 Supervised Learning

3 Text Classifiers

4 Other Issues



# Supervised Learning

Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

Other Issues

Given a set of **training data** as input, use **learning algorithm**  $A$  to discover the function  $\hat{h}$  that minimizes the **loss** (e.g. the error over the set of training instances)

**Input:**  $\{(x_i, y_i)\}_{i=1}^N, x_i \in \mathcal{R}^M, y_i \in \mathcal{R}$

**Hypothesis space:**  $h^* \in H$

**Loss function:**  $L(h(x), y)$

**Learning Algorithm:**  $\hat{h} = A(\{(x_i, y_i)\}_{i=1}^N)$ , such that  
$$\hat{h} = \operatorname{argmin}_h \sum_{i=1}^N L(h(x_i), y_i)$$





# An Example: Linear Regression

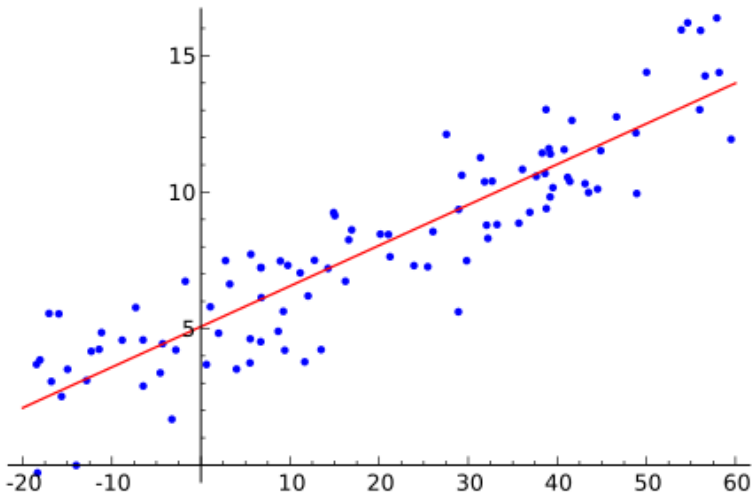
Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

Other Issues



(source: [wikipedia](#))



# Linear Regression (cont.)

Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

Other Issues

- The hypothesis space:

$$h_{\vec{w}}(x) = w_0 + w_1x$$

where  $\vec{w} = [w_0, w_1]$

- The loss function:

$$L(h_{\vec{w}}, y) = \frac{1}{N} \sum_{i=1}^N (y_i - h_{\vec{w}}(x_i))^2$$

i.e. the sum of the squared error

- We want to find

$$w^* = \underset{w}{\operatorname{argmin}} L(h_{\vec{w}}, y)$$



# Minimizing the Loss

Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

Other Issues

- In simple cases, we can easily find one (or more) solution(s) to the problem of learning  $\hat{h}$ 
  - For linear regression, take the derivatives and equal to 0
- In many cases this is not possible (or we may want to enforce some constraints on the parameters)
- In practice, there are many ways to estimate  $w^*$



# An Example: Gradient Descent

Processamento  
e Recuperação  
de Informação

Introduction

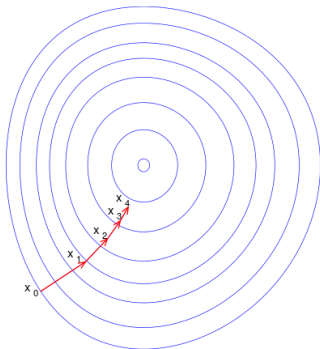
Supervised  
Learning

Text  
Classifiers

Other Issues

$w \leftarrow$  any point in the  
parameter space  
**loop** until convergence **do**  
  **for each**  $w_i$  **in**  $\vec{w}$  **do**  
     $w_i \leftarrow w_i - \alpha \frac{\partial}{\partial w_i} L(h_{\vec{w}}, y)$

$\alpha =$  learning rate



(source: [wikipedia](#))



# Classification

Processamento  
e Recuperação  
de Informação

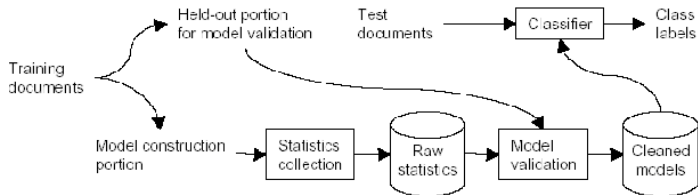
Introduction

Supervised  
Learning

Text  
Classifiers

Other Issues

- Learning to assign objects to classes given examples
- Learn a **classifier** (i.e., map the problem into supervised learning task)





# An Example: Logistic Regression

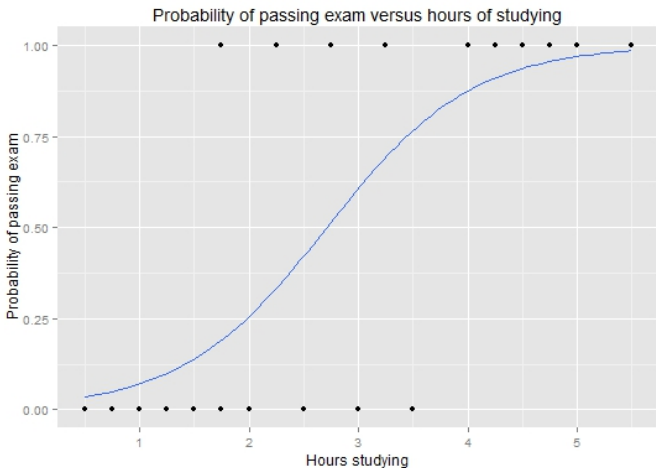
Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

Other Issues



(source: [wikipedia](#))



# Logistic Regression (cont.)

Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

Other Issues

- The hypothesis space:

$$h_{\vec{w}}(x) = \frac{1}{1 + e^{-(w_0 + w_1 x)}}$$

where  $\vec{w} = [w_0, w_1]$

- The loss function:

$$L(h_{\vec{w}}, y) = \frac{1}{N} \sum_{i=1}^N C(h_{\vec{w}}(x_i), y)$$

where

$$C(h_{\vec{w}}(x), y) = \begin{cases} -\log(h_{\vec{w}}(x)) & \text{if } y = 1 \\ -\log(1 - h_{\vec{w}}(x)) & \text{if } y = 0 \end{cases}$$

- We want to find

$$w^* = \underset{w}{\operatorname{argmin}} L(h_{\vec{w}}, y)$$



# Outline

Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

Nearest Neighbor  
Classifiers

Generative Bayesian  
Classifiers

Linear Discriminative  
Classifiers

Neural Networks

Other Issues

## 1 Introduction

## 2 Supervised Learning

## 3 Text Classifiers

- Nearest Neighbor Classifiers
- Generative Bayesian Classifiers
- Linear Discriminative Classifiers
- Neural Networks

## 4 Other Issues





# Text Classification vs. Data Mining

Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

Nearest Neighbor  
Classifiers

Generative Bayesian  
Classifiers

Linear Discriminative  
Classifiers

Neural Networks

Other Issues

Leverage supervised learning together with method for representing textual information (e.g., VSM with TF-IDF)

- Lots of features and a lot of noise
- No fixed number of columns
- No categorical attribute values
- Data scarcity
- Larger number of class labels
- Hierarchical relationships between classes less systematic



# Text Classifiers

Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

Nearest Neighbor  
Classifiers

Generative Bayesian  
Classifiers

Linear Discriminative  
Classifiers

Neural Networks

Other Issues

- Nearest Neighbor Classifiers
  - Classify documents according to the class distribution of their neighbors
- Generative Bayesian classifiers (e.g., naïve Bayes)
  - Discover the class distribution most likely to have generated a test document
- Linear discriminative classifiers (e.g., the perceptron, logistic regression, or support vector machines):
  - Discover an hyperplane that separates classes
- Neural networks
  - Discover a non-linear function, often resulting from a composition of many functions, that separates classes



# Nearest Neighbor Classifiers

Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

Nearest Neighbor  
Classifiers

Generative Bayesian  
Classifiers

Linear Discriminative  
Classifiers

Neural Networks

Other Issues

- Intuition: similar documents are expected to be assigned the same class label
  - Similarity: vector space model + cosine similarity
- Training:
  - Index each document and remember class label
- Testing:
  - Fetch *k* most similar documents to the given document
  - Majority class wins
  - Alternatives:
    - Weighted counts: counts of classes weighted by the corresponding similarity measure
    - Per-class offset: tuned by testing the classifier on a portion of training data held out for this purpose



# kNN Classifier

Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

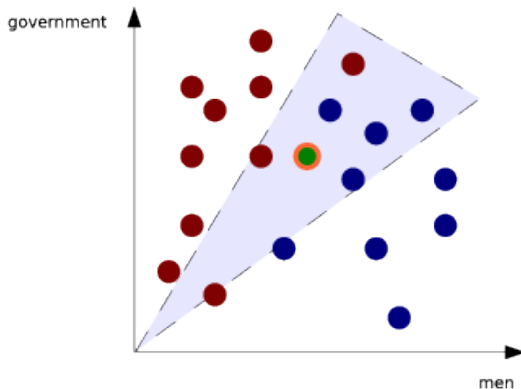
Nearest Neighbor  
Classifiers

Generative Bayesian  
Classifiers

Linear Discriminative  
Classifiers

Neural Networks

Other Issues



$$score(c, d_q) = b_c + \sum_{d \in kNN(d_q)} sim(d_q, d)$$



# Properties of $k$ NN

Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

Nearest Neighbor  
Classifiers

Generative Bayesian  
Classifiers

Linear Discriminative  
Classifiers

Neural Networks

Other Issues

- Advantages:
  - Reuse of standard vector space model and availability of associated technology (e.g., inverted indexes)
  - Collection updates are trivial
  - Accuracy comparable to best known classifiers
- Problems:
  - Classification efficiency
    - many lookups over the document collection/index
    - sorting by overall similarity
    - picking the best  $k$  documents
  - Space overhead and redundancy
    - Data stored at level of individual documents
    - Poor generalization
  - Choosing a value for  $k$



# Improvements for $k$ NN

Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

Nearest Neighbor  
Classifiers

Generative Bayesian  
Classifiers

Linear Discriminative  
Classifiers

Neural Networks

Other Issues

- To reduce space requirements and speed up classification
  - Find clusters in the data and start by comparing instances against clusters (**clustering covered in the next lecture**)
  - Store only a few statistical parameters per cluster
  - In second step, compare with documents in only the most promising clusters
- However...
  - Ad-hoc choices for number and size of clusters and parameters
  - Number of clusters depends on the data



# Bayesian Classifiers

Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

Nearest Neighbor  
Classifiers

Generative Bayesian  
Classifiers

Linear Discriminative  
Classifiers

Neural Networks

Other Issues

- Probabilistic document classifier
- Assumptions:
  - 1 A document can belong to **exactly one class**
  - 2 Each class  $c$  has an associated prior probability  $P(c)$
  - 3 There is a class-conditional document distribution  $P(d|c)$  for each class (i.e., the likelihood)
- Given a document  $d$ , the probability of it being generated by class  $c$  is:

$$P(c|d) = \frac{P(d|c)P(c)}{\sum_{\gamma} P(d|\gamma)P(\gamma)}$$

- The class with the highest probability is assigned to  $d_q$  (i.e., we use a *maximum a-posteriori* rule)



# Learning the Document Distribution

Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

Nearest Neighbor  
Classifiers

Generative Bayesian  
Classifiers

Linear Discriminative  
Classifiers

Neural Networks

Other Issues

- $P(d|c)$  is estimated based on parameters  $\Theta$
- $\Theta$  are estimated based on two factors:
  - 1 Prior knowledge before seeing any documents
  - 2 Terms in the training documents
- Bayes Optimal Classifier

$$P(c|d) = \int_{\Theta} \frac{P(d|c, \Theta)P(c|\Theta)}{\sum_{\gamma} P(d|\gamma, \Theta)P(\gamma|\Theta)} P(\Theta|D)$$

- This can be hard to compute
- Maximum Likelihood Estimate:  $P(d|c, \hat{\Theta})$

$$\hat{\Theta} = \operatorname{argmax}_{\Theta} P(d|c, \Theta)$$





# Naïve Bayes Classifier

Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

Nearest Neighbor  
Classifiers

Generative Bayesian  
Classifiers

Linear Discriminative  
Classifiers

Neural Networks

Other Issues

- Naïve assumption
  - assumption of **independence between terms**
  - joint term distribution is the product of the marginals
- Widely used owing to
  - simplicity and speed of training, applying, and updating
- Two kinds of widely used marginals for text
  - Binary model (Bernoulli)
  - Multinomial model



# Naïve Bayes Models

Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

Nearest Neighbor  
Classifiers

Generative Bayesian  
Classifiers

Linear Discriminative  
Classifiers

Neural Networks

Other Issues

**Binary Model:** Each parameter  $\theta_{c,t}$  indicates the probability that a document in class  $c$  will mention term  $t$  at least once

$$P(d|c, \Theta) = \prod_{t \in d} \theta_{c,t} \prod_{t \notin d} (1 - \theta_{c,t})$$

$$\theta_{c,t} = \frac{N_{c,t}}{N_c}$$

$N_{c,t}$  = n. of docs in class  $c$  containing term  $t$

$N_c$  = n. of docs in class  $c$



# Naïve Bayes Models (cont.)

Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

Nearest Neighbor  
Classifiers

Generative Bayesian  
Classifiers

Linear Discriminative  
Classifiers

Neural Networks

Other Issues

## Multinomial Model:

- each class has an associated die with  $|W|$  faces
- each parameter  $\theta_{c,t}$  denotes probability of the face turning up on tossing the die, i.e.  $\sum_{d \in c} n(d, t) / \sum_{d \in c} \ell_d$
- term  $t$  occurs  $n(d, t)$  times in document  $d$
- document length is a random variable denoted  $L$

$$\begin{aligned} P(d|c, \Theta) &= P(L = \ell_d | c) P(d | \ell_d, c) \\ &= P(L = \ell_d | c) \frac{\ell_d!}{\prod_{t \in d} n(d, t)!} \prod_{t \in d} \theta_{c,t}^{n(d,t)} \\ &\sim P(L = \ell_d | c) \prod_{t \in d} \theta_{c,t}^{n(d,t)} \end{aligned}$$



# Parameter Smoothing

Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

Nearest Neighbor  
Classifiers

Generative Bayesian  
Classifiers

Linear Discriminative  
Classifiers

Neural Networks

Other Issues

- What if a test document  $d_q$  contains a term  $t$  that never occurred in any training document in class  $c$ ?



# Parameter Smoothing

Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

Nearest Neighbor  
Classifiers

Generative Bayesian  
Classifiers

Linear Discriminative  
Classifiers

Neural Networks

Other Issues

- What if a test document  $d_q$  contains a term  $t$  that never occurred in any training document in class  $c$ ?
  - $P(c|d_q) = 0$
  - Even if many other terms clearly hint at a high likelihood of class  $c$  generating the document
- Thus, MLE cannot be used directly
- We can use **Laplace smoothing**
  - Simply adds 1 to each count

$$\theta_{c,t} = \frac{\sum_{d \in c} n(d, t) + 1}{\sum_{d \in c} \ell_d + |W|}$$



# Performance Analysis

Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

Nearest Neighbor  
Classifiers

Generative Bayesian  
Classifiers

Linear Discriminative  
Classifiers

Neural Networks

Other Issues

- Multinomial naïve Bayes classifier generally outperforms the binary variant
- $k$ NN may outperform Naïve Bayes
- Naïve Bayes is faster and more compact
- Determines **decision boundaries**
  - Regions of the term-space where different classes have similar probabilities
  - Documents in these regions are hard to classify
  - Strongly biased



# Discriminative Classification

Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

Nearest Neighbor  
Classifiers

Generative Bayesian  
Classifiers

Linear Discriminative  
Classifiers

Neural Networks

Other Issues

- Naïve Bayes classifiers are **generative**
- Differently, **discriminative** classifiers:
  - Directly map the feature space to class labels
  - Class labels are encoded as numbers
    - e.g: +1 and -1 for two a class problem
- For instance, we can try to find a vector  $\alpha$  such that the sign of  $\alpha \cdot d + b$  directly predicts the class of a document  $d$
- Possible solutions:
  - Linear least-square regression
  - **The Perceptron**
  - **Logistic Regression**
  - **Support Vector Machines**



# What is a Linear Discriminative Classifier?

Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

Nearest Neighbor  
Classifiers

Generative Bayesian  
Classifiers

Linear Discriminative  
Classifiers

Neural Networks

Other Issues

- Essentially:

- Classification decision is based on the value of a linear combination of the features
- Can be seen as the splitting of a high-dimensional input space with a hyperplane

$$y(d_1, \dots, d_n) = f(\alpha_1 d_1 + \alpha_2 d_2 + \dots + \alpha_n d_n)$$

- $\alpha_i$  are parameters (i.e., the weight of each feature  $d_i$ )
- $f$  is the activation function (e.g.,  $f(d) = 1_{x \geq 0}(d)$ )
- The result of  $y(d_1, \dots, d_n)$  corresponds to the estimated class





# The Bias Term

Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

Nearest Neighbor  
Classifiers

Generative Bayesian  
Classifiers

Linear Discriminative  
Classifiers

Neural Networks

Other Issues

- Notice that the decision hyperplane must go through the origin
- Could be achieved by preprocessing the input, but this is not always desirable or possible
- Solution : Add a bias input:

$$y(d_1, \dots, d_n) = f(b + \alpha_1 d_1 + \dots + \alpha_n d_n)$$

- Same as an input connected to the constant 1
- We consider this *ghost* input implicit henceforth



# Training : The Perceptron Algorithm

Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

Nearest Neighbor  
Classifiers

Generative Bayesian  
Classifiers

Linear Discriminative  
Classifiers

Neural Networks

Other Issues

- Switching to vector notation:

$$y(\mathbf{d}) = f(\alpha \mathbf{d}) = f_{\alpha}(d) \quad (1)$$

- Assume we need to separate sets of points  $A$  (i.e., the positive examples) and  $B$  (i.e., the negative examples)

$$E(\alpha) = \sum_{\mathbf{d} \in A} (1 - f_{\alpha}(\mathbf{d})) + \sum_{\mathbf{d} \in B} f_{\alpha}(\mathbf{d}) \quad (2)$$

- Goal:  $E(\alpha) = 0$
- Start from a random  $\alpha$  and improve it iteratively



# Algorithm Pseudo-Code

Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

Nearest Neighbor  
Classifiers

Generative Bayesian  
Classifiers

Linear Discriminative  
Classifiers

Neural Networks

Other Issues

- ➊ Start with random  $\alpha$ , set  $t = 0$
  - ➋ Select a vector  $\mathbf{d} \in A \cup B$
  - ➌ If  $\mathbf{d} \in A$  and  $\alpha \mathbf{d} \leq 0$ , then  $\alpha_{t+1} = \alpha_t + \mathbf{d}$
  - ➍ Else if  $\mathbf{d} \in B$  and  $\alpha \mathbf{d} \geq 0$ , then  $\alpha_{t+1} = \alpha_t - \mathbf{d}$
  - ➎ Conditionally go to step 2
- Guaranteed to converge iff  $A$  and  $B$  are linearly separable!



# Problems of Simple Perceptrons

Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

Nearest Neighbor  
Classifiers

Generative Bayesian  
Classifiers

Linear Discriminative  
Classifiers

Neural Networks

Other Issues

- The standard Perceptron returns the most recent version of the weight vector
- Intuitively, this version is over-adapted to the last few instances, and may work less well for other instances
- The **Averaged Perceptron** returns the average of all versions (or the last few versions) of the weight vector
- An implementation trick involves setting a learning step that takes the averaging effect into account

- Several problems involve multi-class classification
- Multi-class classification requires one weight vector for each category
- In practice, we can represent this with one giant weight vector and repeated features for each category
- Update rule involves changing the weights for the true



# Summary of Simple/Averaged Perceptrons

Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

Nearest Neighbor  
Classifiers

Generative Bayesian  
Classifiers

Linear Discriminative  
Classifiers

Neural Networks

Other Issues

- Simple and reasonably efficient online training
- Easy to extend in order to consider multi-class classification
- Works well for document classification, and more generally for problems with many features
- Limited capabilities (e.g., does not try to optimize the separation “distance” between classes)
  - Just looks for a hyperplane that separates the two sets
  - Methods such as **Support Vector Machines**, on the other hand, try to maximize the distance between two closest opposite sample points (i.e., the **margin of the separating hyperplane**)



# Linear Discriminative Classifiers and SVMs

Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

Nearest Neighbor  
Classifiers

Generative Bayesian  
Classifiers

Linear Discriminative  
Classifiers

Neural Networks

Other Issues

- Hypothesis:
  - The classes can be separated by an **hyperplane**
  - The hyperplane that is close to many training data points has a greater chance of misclassifying test instances
  - An hyperplane that passes through a "no-man's land", has lower chances of misclassifications
- Make a decision by thresholding
  - Seek an hyperplane that maximizes the distance to any training point
  - Choose the class on the same side of the hyperplane as the test document (i.e., same as in the Perceptron)



# Discovering the Hyperplane

Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

Nearest Neighbor  
Classifiers

Generative Bayesian  
Classifiers

Linear Discriminative  
Classifiers

Neural Networks

Other Issues

- Assume the training documents are separable by an hyperplane perpendicular to a vector  $\alpha$
- Seek a vector  $\alpha$  which maximizes the distance of any training point to the hyperplane
- This corresponds to solving the following **quadratic programming** problem:

$$\begin{array}{ll}\text{Minimize} & \frac{1}{2} \alpha \cdot \alpha \\ \text{subject to} & c_i(\alpha \cdot d_i + b) \geq 1, \forall i = 1, \dots, n\end{array}$$



# SVM Classifier

Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

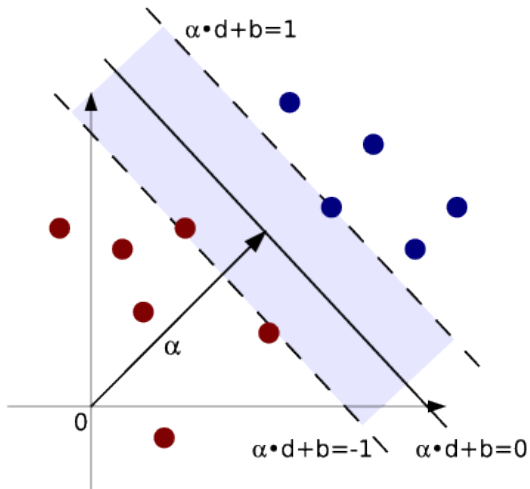
Nearest Neighbor  
Classifiers

Generative Bayesian  
Classifiers

Linear Discriminative  
Classifiers

Neural Networks

Other Issues







# Non Separable Classes

Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

Nearest Neighbor  
Classifiers

Generative Bayesian  
Classifiers

Linear Discriminative  
Classifiers

Neural Networks

Other Issues

- Classes in the training data not always separable
- We introduce **slack variables**

$$\begin{aligned} &\text{Minimize} && \frac{1}{2}\alpha \cdot \alpha + C \sum_i \xi_i \\ &\text{subject to} && c_i(\alpha \cdot d_i + b) \geq 1 - \xi_i, \forall i = 1, \dots, n \\ &&& \text{and } \xi_i \geq 0, \forall i = 1, \dots, n \end{aligned}$$

- Implementations often solve the equivalent dual problem

$$\begin{aligned} &\text{Maximize} && \sum_i \lambda_i - \frac{1}{2} \sum_{i,j} \lambda_i \lambda_j c_i c_j (d_i \cdot d_j) \\ &\text{subject to} && \sum_i c_i \lambda_i = 0 \\ &&& \text{and } 0 \leq \lambda_i \leq C, \forall i = 1, \dots, n \end{aligned}$$



# Analysis of SVMs

Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

Nearest Neighbor  
Classifiers

Generative Bayesian  
Classifiers

Linear Discriminative  
Classifiers

Neural Networks

Other Issues

- Complexity:
  - Quadratic optimization problem
  - Requires on-demand computation of inner-products
  - Recent SVM packages work in linear time
- Performance:
  - Amongst most accurate classifier for text
  - Better accuracy than Naïve Bayes and most classifiers
  - Linear SVMs suffice
    - Standard text classification tasks have classes almost separable using a hyperplane in feature space
  - Non-linear SVMs can be achieved through **kernel functions**



# Neural Networks

Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

Nearest Neighbor  
Classifiers

Generative Bayesian  
Classifiers

Linear Discriminative  
Classifiers

Neural Networks

Other Issues



# Outline

Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

Other Issues

- 1 Introduction
- 2 Supervised Learning
- 3 Text Classifiers
- 4 Other Issues**



# Other Issues (1)

Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

Other Issues

- Tokenization and feature extraction
  - E.g.: replacing monetary amounts by a special token, part-of-speech tagging, representations based on  $n$ -grams, etc.
- Handling scenarios with multiple classes, or with multiple labels per test document, with binary classifiers like SVMs
  - E.g., one-vs.-rest heuristic
    - e.g. “sports” vs. “not-sports”, “science” vs. “not-science”, etc.
    - Create a classifier for each case
    - Assign class(es) with the highest confidence



## Other Issues (2)

Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

Other Issues

- Evaluating text classifiers
  - Accuracy
  - Training speed and scalability
  - Simplicity, speed, and scalability for document modifications
  - Ease of diagnosis, interpretation of results, and adding human judgment and feedback
- Many other practical issues...



Processamento  
e Recuperação  
de Informação

Introduction

Supervised  
Learning

Text  
Classifiers

Other Issues

# Questions?