University of Victoria Computer Science

ISMIR 2017 tutorial

Bayes and Markov Listen to Music

George Tzanetakis

University of Victoria

2017

Table of Contents I

Table of Contents II

Motivation

Probabilistic modeling is essential in understanding modern techniques in digital signal processing and machine learning. This course is a short but comprehensive introduction to probabilistic modeling through the lens of music information retrieval.

Overview

There are 6 units with durations ranging from 15-45 minutes. Each unit consists of a set of slides as well as associated hands-on demonstrations in the form of Python notebooks. All the source code will be provided to participants and there is considerable flexibility in how much time we spend on each unit.
The units are:

- Motivation - Overview
- Discrete Probabilities and Symbolic Music Processing
- Classification using probabilities
- Dealing with uncertainty and time
- Probabilistic graphical models
- Markov Logic Networks

Reading resources

University of Victoria Computer Science

There are many excellent books on probabilistic modeling and machine learning. Probably my favorite is **"Machine Learning: a Probabilistic Perspective"** by Kevin Murphy. Another good book is **"Probabilistic Graphical Models"** by Daphne Koller. A book with a similar probabilistic perspective that is available online is **"Model-Based Machine Learning"** by John Winn and Christopher Bishop (`http://www.mbmlbook.com/`).

Education and Academic Work Experience

University of Victoria Computer Science

- 1997 BSc in Computer Science (CS), University of Crete, Greece
- 1999 MA in CS, Princeton University, USA
- 2002 PhD in CS, Princeton University, USA
- 2003 PostDoc in CS, Carnegie Mellon University, USA
- 2004 Assistant Professor in CS, Univ. of Victoria, Canada
- 2010 Associate Professor in CS, Univ. of Victoria, Canada
- 2016 Professor in CS, Univ. of VIctoria, Canada
- 2010 (renewed 2015) Canada Research Chair (Tier II) in Computer Analysis of AUdio and Music
- Music theory, saxophone and piano performance, composition, improvisation both in conservatory and academic settings

Research

University of Victoria Computer Science

Inherently inter-disciplinary and cross-disciplinary work. Connecting theme: making computers better understand music to create more effective interactions with musicians and listeners. Audio analysis is challenging due to large volume of data - did big data before it became fashionable.

- Music Information Retrieval
- Digital Signal Processing
- Machine Learning
- Human-Computer Interaction
- Software Engineering
- Artifical Intelligence
- Multimedia
- Robotics
- Visualization
- Programming Languages

Research

University of Victoria Computer Science

Inherently inter-disciplinary and cross-disciplinary work. Connecting theme: making computers better understand music to create more effective interactions with musicians and listeners. Audio analysis is challenging due to large volume of data - did big data before it became fashionable.

- Music Information Retrieval
- Digital Signal Processing
- Machine Learning
- Human-Computer Interaction
- Software Engineering
- Artifical Intelligence
- Multimedia
- Robotics
- Visualization
- Programming Languages

Table of Contents I

Table of Contents II

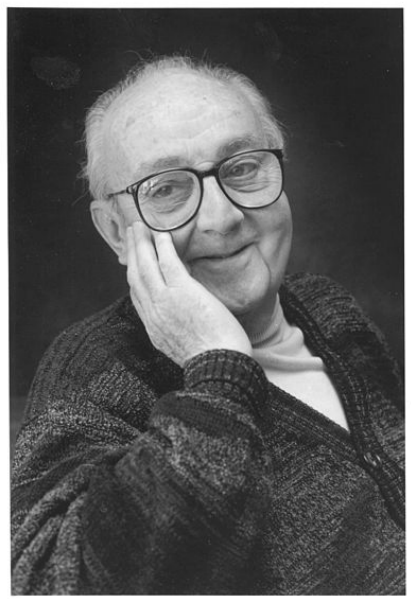
In a nutshell

University of Victoria Computer Science

Quote

Essentially, all models are wrong but some are useful.

George Box (1919, 2013). He got interested in statistics after performing experiments on the effect of poison gas to small animals in WW II.



A model-based view of machine learning

The basic recipe:

- **Generate/Sample:** Describe how the data is generated and the assumptions you make using a probabilistic model
- **Learn/Estimate:** Estimate the parameters of the probabilistic model using available data (the learning part)
- **Reason/Infer:** Use the estimated probabilistic model to reason about the problem at hand by inference i.e assigning probabilities to events of interest that are not known before hand.
- **Evaluate:** Examine how well the model performs and what types of errors it makes

Important insights

- Understanding notation in addition to the underlying concepts is important
- Separating model from inference
- Understanding the connection between statistics and probability
- Thinking the generative way
- Probabilistic modeling is all about how to calculate probabilities of events that are "hard" to estimate from probabilities of events that are "easier" to estimate
- Focus on the basic concepts and don't get bogged down in the implementation details and the multiple variants
- Misleading use of language is frequently why probability problems can be difficult (for example Monty Hall). In most actual applications that's not a problem.

Table of Contents I

Table of Contents II

Overview

The main goal of this section is to introduce notation and the very basic concepts underlying probabilities. We will start with a very simple example and then use the learned concepts to do some simple symbolic music modeling using Python.

- Probability - baysian vs frequentist interpretation
- Random variables
- Probabilistic Model
- Inference

Probability as a framework for quantifying uncertainty

Almost every computational model of some process has to account for uncertainty. There are many sources for this uncertainty including: measurement noise, wrong model assumptions, lack of knowledge and many others. *Probability* probably provides the best principled framework for quantifying the degree of uncertainty for various phenomena in order to build applications and tools that interact with the real world.

Frequentist interpretation

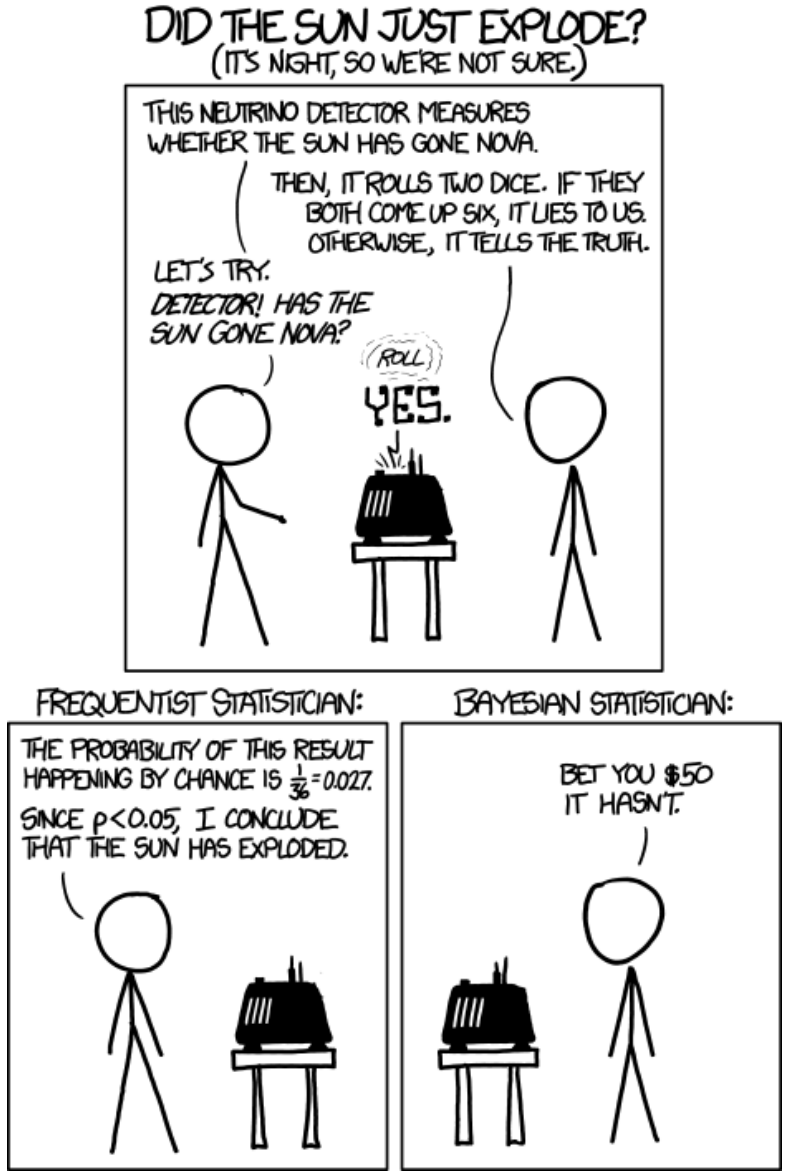
An event's probability is the limit of its relative frequency in a large number of trials. This connects to statistics and empirical experiments. The initial classical definition of probability was based on physical idealized symmetry (dice, coins, cards). The axiomatic formulation of probability by Kolmogorov (1903-1987) in 1933 focuses on operations on probability values rather than the initial assignment of values.

Bayesian Probabilities

University of Victoria Computer Science

Probability summarizes (with a value between 0 and 1) the uncertainty we have about the world. Probabilities (between 0 and 1) correspond to intermediate degrees of belief in the truth of a sentence. Important: the sentence itself is either true or false. Degree of belief is different than degree of truth and depends on our current state of knowledge.
Consider drawing a card and asking what is the probability it is the card is the ace of spades. Before looking at the card the probability is $\frac{1}{52}$ but after it will either be 0 or 1. Therefore, all probability statements should indicate the evidence used to obtain the probability estimate. The probability before evidence is obtained is termed the prior (or unconditional) and after evidence the posterior (or conditional).

Frequentists vs Bayesians (xkcd)



Random variables

In order to specify the uncertainty about different parts of a problem that we wish to model we will use **Random Variables**. We specify a RV with a name and a domain from which it takes values. Discrete random variables take specific discrete values. For example in music modeling a random variable for *genre* might take the values *country*, *jazz*, and the random variable *hasLyrics* might take the values *yes*, *no*.

Assigning probabilities to values of RV

We can associate a probability, a number between 0 and 1 to an assignment of a random variable to a value. For example we can have:

$$P(genre = country) = 0.7$$

Notice that the sum of the probabilities of all assignments of values to random variable must add up to 1. This is a normalization constraint and is essential to being able to combine probability estimates. Based on it we can state that:

$$P(genre = jazz) = 0.3$$

Notice that I have not said anything about how this number 0.7 came about. For now you can think of it as an educated guess. Later we will talk about how these numbers can be learned from data.

Probability Notation

Random Variables represents a "part" of the world whose "status" is initially unknown. Each random variable has a domain that it can take on. For example, the **RV** *Weather* can have the values: *sun*, *rain*, *cloud*, *snow*. Domains can be boolen, discrete, continuous.

Probability notation continued

Probabilities are assigned over values in the domain. The notation $\mathbf{P}(\mathit{Weather})$ denotes a vector of values for the probabilities of each individual state of the weather:

$$
\begin{aligned}
P(\mathit{Weather} = \mathit{sunny}) &= 0.65 \\
P(\mathit{Weather} = \mathit{rain}) &= 0.25 \\
P(\mathit{Weather} = \mathit{cloudy}) &= 0.07 \\
P(\mathit{Weather} = \mathit{snow}) &= 0.03 \\
\mathbf{P}(\mathit{Weather}) &= (0.65, 0.25, 0.07, 0.03)
\end{aligned}
$$

Probability

Set of all possible outcomes of a random experiment is called the **sample space**. $\Omega = 1, 2, 3, 4, 5, 6$ is the sample space of rolling a die. An **event** is a subspace of these outcomes. For example $E = 1, 3, 5$ is the event of observing an odd number when rolling a die. A probability P is a real-valued function defined on the sample space Ω that informally assigns values between 0 and 1 to all events.

Probability properties

- For any event $E \subset \Omega, \quad 0 \leq P(E) \leq 1$
- $P(\Omega) = 1$
- For any set of disjoint events, $E_1, E_2, ..., E_k \in \Omega$, $P(\cup_{i=1}^{k} E_i) = \sum_{i=1}^{k} P(E_i)$

Note: These properties help us calculate unknown probabilities of events based on known probabilities of other events. How the numbers are assigned to particular events is problem and domain dependent. They can be assigned based on degrees of belief or they can be estimated by statistical frequency of occurence.

Probability distribution and sampling

The probabilities associated with every possible value of a random variable consitute a **probability distribution**. The process of selecting a value randomly according to the probability distribution is called **sampling**. It can be viewed as a process of generating a sequence of random **samples** and it can help us better understand how a particular probabilistic model works.

Hands-on Example of Probability Distribution and Sampling

Jupyter notebook in Python exploring sampling of probability distributions

Incorporating evidence

Suppose we introduce another random variable *hasLyrics* with two values: *no* and *yes*. We expect that more country songs will have lyrics than jazz songs. That means that the probability distribution of *hasLyrics* depends on whether the *genre* is *country* or *jazz*. This is known as a conditional probability distribution and is notated as follows:

$$P(hasLyrics = no/genre = jazz) = 0.9$$

This implies that $P(hasLyrics = yes|genre = jazz) = 0.1$

Conditional probability

If $genre = country$ then we have:

$$P(hasLyrics = no|genre = country) = 0.2$$

We can use the short-hand notation $P(hasLyrics|genre)$ to denote the conditional probability distribution that in this case can be specified by providing four probabilities (or two using normalization). We will call these numbers and in general any numbers used to "specify" a particular probabilistic model **parameters** and use θ to denote a vector containing them.

Conditional Probability Table

We can display all the relevant probabilities using a conditional probability table. Notice that the sum of row entries must be equal to 1 but NOT the sum of column entries.

genre	hasLyrics = no	hasLyrics = yes
country	0.2	0.8
jazz	0.9	0.1

A note about notation

Frequently when notating a conditional probablity distribution the short hand $P(hasLyrics|genre)$ is used. Conceptually this expands to all possible combinations of values of the two random variables involved. Also some times when the values of random variables in a problem are unique the name of the random variable is omitted i.e $P(country)$ instead of $P(genre = country)$. It is important to keep in mind these conventions as our examples get more complicated.

Independent variables

Definition

Two variables A and B are said to be **independent** if $P(A, B) = P(A)P(B)$ (or equivalently $P(A) = P(A|B)$)

Suppose that you toss the same coin twice. If A represent the random variable of the first toss and B represents the random variable of the second toss then we can assume that A and B are independent. Evidence about B will not change our belief about A.

Now consider as A the event of rolling a 5 the first time a die is rolled and B the event that the sum of numbers of the first roll and the second roll is 10. In this case B is not independent of A.

Our first probabilistic model

An illuminating way to think about what we have looked so far is as a description of a process that takes into account the various sources of uncertainty. First we randomly choose whether the song is jazz or country based on the prior distribution (equivalent to randomly picking a record from a shelf). There is a 70% chance the song will be country and a 30% it will be jazz. Let's suppose that it is a country song. Then there is a 20% chance it does not have lyrics and and a 80% it has lyrics. Let's consider the event that the piece is country and it does not have lyrics. The probability of this event is $70\% \times 20\% = 14\%$.

Joint probability

University of Victoria Computer Science

This is the **joint probability** of choosing *hasLyrics=yes* and *genre=jazz*. We can use the short hand notation $P(hasLrics, genre)$ to refer to this joint probability distribution. Note that the total sum over all consituent events of a joint probaiblity should be 1:

$$\sum_A \sum_B P(A, B) = 1$$

Probabilistic models

University of Victoria Computer Science

Definition

Probabilistic Model

- A set of random variables,
- A joint probability distribution over these variables (i.e. a distribution that assigns a probability to every configuration of these variables such that the probabilities add up to 1 over all possible configurations).

Inference

When we have a probabilistic model we can make predictions, learn about the values of some random variables given the values of others, and in general, answer any possibly questions that can be stated about the random variables. The probablistic model expresses the set of assumptions we are making about the problem we are tryng to solve and our uncertainty about them is expressed through probabilities. Typically we will know the values of some random variables in our model (evidene) and based on this knowledge we will want to **infer** something about the probability distribution of some other variables.

Product rule

We have seen that:

$$P(hasLyrics, genre) = P(genre)P(hasLyrics/genre)$$

This is an example of the **product rule**:

$$P(A, B) = P(A)P(B|A).$$

Sum rule I

We can sum the joint probabilities for all possible values of genre to "eliminate" that variable.

$$\sum_{hasLyrics} P(hasLyrics, genre = country) = P(genre = country).$$

More generally using short-hand notation we can express that this holds for all values of *genre* :

$$\sum_{hasLyrics} P(hasLyrics, genre) = P(genre).$$

Sum rule II

More generally the sum rule of probability states:

$$P(A) = \sum_{B} P(A, B)$$

In this context, the distribution $P(A)$ is known as the marginal distribution for A and the act of summing out B is called marginalisation.

The sum and product rules

The sum and product rules are very general. They apply not just when A and B are binary random variables, but also when they are multi-state random variables, and even when they are continuous (in which case the sums are replaced by integrations). Furthermore, A and B could each represent sets of several random variables. For example if $B = C, D$:

$$P(A, C, D) = P(A)P(C, D|A) \qquad P(A) = \sum_C \sum_D P(A, C, D)$$

Inference Example

Recall that the process of obtaining revised probability distributions after the values of some random variables have been observed, is called **inference**. Let's look at an example. We know that the probability of a song is jazz is 30%. Suppose that we observe that the song does not have lyrics. How does this evidence affect the probability that the song is jazz ? We have:

$$P(genre = jazz|hasLyrics = no) = \frac{0.3 * 0.9}{0.3 * 0.9 + 0.7 * 0.2} \approx 0.66$$

Notice that this **posterior** probability after incorporating evidence is more than twice the original **prior** probability.

Hands-on Example Conditional Probabilities

Jupyter notebook in Python exploring conditional probability distributions.

Table of Contents I

Table of Contents II

Continuous RVs

University of Victoria Computer Science

Continuous Random Variables

For continuous variables it is not possible to write out the distribution as a table as it would have infinite many values. Instead, the probability that a random variable takes on some value x is represented as a parameterized function of x. For example we could have $P(X = x) = U[5, 20](x)$ express the belief that the temperature tomorrow will be uniformly distributed between 5 and 20 degrees Celcius. Notice that P(X = 18) should not be interpreted as the probability that the temperature is exactly 18 degrees which is zero. Instead, the temperature in a small region around 18 degrees is equal to the value in the limit. Sometimes lower case $p(x)$ is used to differentiate continuous (density) from discrete distributions.

Continuous probability functions

University of Victoria Computer Science

	Probability Function	Parameters
Gaussian	$p(x) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}}$	μ, σ
Binomial	$p(x) = \binom{n}{x} p^x (1-p)^{n-x}$	n, p
Poisson	$p(x) = \frac{1}{x!}\theta^x e^{-\theta}$	θ
Exponential	$p(x) = \theta e^{-\theta x}$	θ

Maximum Likelihood Estimation of Parameters

University of Victoria Computer Science

Definition

A parametric statistical model is a collection of probability distribution functions each of which is uniquely characterized by a finite dimensional vector of parameters (θ). For example Gaussian distributions are characterized by their mean and covariance matrix.

Definition

Given a dataset and set of models find the set of parameters that maximizes the likelihood of observing the data given the model. In well-behaved cases this estimation can be performed analytically rather than numerically using optimization methods. For example the ML estimate of the mean and covariance matrix are the sample mean and sample covariance.

Likelihood

University of Victoria Computer Science

Definition

The likelihood of a set of data for a particular statistical model assumes that the data is i.i.d (independent and identically distributed). That means that:

$$P(\mathbf{x}|M) = \prod_i P(x_i|M) = \prod_i P(x_i|\theta), \tag{1}$$

where θ is a vector of parameters characterizing the model M and x_i are the data points. If we consider the equation above as a function of the θ for the data point x_i we obtain:

$$L(\theta|x_1, \ldots, x_n) = \prod_i P(x_i|\theta) \tag{2}$$

$$\ln(\theta|x_1, \ldots, x_n) = \ln \prod_i P(x_i|\theta) = \sum_i \ln P(x_i|\theta) \tag{3}$$

Hands-on demonstration of continuous probability functions and ML estimation

Definition

University of Victoria Computer Science

Definition

Classification (or Supervised Learning) is the machine learning task of inferring the label (class) of an object (typically represented as a vector of numbers) by analyzing training data consisting of feature vectors (representing objects) and associated labels.

Example

University of Victoria Computer Science

Example

You are given the height, weight of 1000 people as well as a binary attribute that indicates whether they are professional basketball (or if you prefer hockey) players or not (training set). You are then given the height and weight of 100 new people and are asked to predict whether they are professional basketball players or not (testing set). The number of correct predictions is the performance metric.

Train Set and Test Set

$$\left(\begin{array}{cc|c} H(cm) & W(kgr) & NBA \\ 190 & 100 & 0 \\ 195 & 110 & 0 \\ 180 & 85 & 0 \\ \dots & & \\ 178 & 75 & 1 \end{array}\right) \qquad \left(\begin{array}{cc|c} H(cm) & W(kgr) & NBA \\ 193 & 102 & ? \\ 180 & 87 & ? \\ 200 & 150 & ? \\ \dots & & \\ 168 & 65 & ? \end{array}\right)$$

Definition

A feature matrix is a 2D matrix where each row (instance or sample) is a feature vector consististing of numbers (attribute or feature) characterizing a particular object and an associated class label (the ground truth). A column corresponds to attribute or feature.

Train Set and Test Set

$$\left(\begin{array}{cc|c} H(cm) & W(kgr) & NBA \\ 190 & 100 & 0 \\ 195 & 110 & 0 \\ 180 & 85 & 0 \\ \dots & & \\ 178 & 75 & 1 \end{array}\right) \qquad \left(\begin{array}{cc|c} H(cm) & W(kgr) & NBA \\ 193 & 102 & ? \\ 180 & 87 & ? \\ 200 & 150 & ? \\ \dots & & \\ 168 & 65 & ? \end{array}\right)$$

Definition

A feature matrix is a 2D matrix where each row (instance or sample) is a feature vector consististing of numbers (attribute or feature) characterizing a particular object and an associated class label (the ground truth). A column corresponds to attribute or feature.

Train Set and Test Set

University of Victoria Computer Science

$$\left(\begin{array}{cc|c} H(cm) & W(kgr) & NBA \\ 190 & 100 & 0 \\ 195 & 110 & 0 \\ 180 & 85 & 0 \\ \dots & & \\ 178 & 75 & 1 \end{array}\right) \qquad \left(\begin{array}{cc|c} H(cm) & W(kgr) & NBA \\ 193 & 102 & ? \\ 180 & 87 & ? \\ 200 & 150 & ? \\ \dots & & \\ 168 & 65 & ? \end{array}\right)$$

Definition

A feature matrix is a 2D matrix where each row (instance or sample) is a feature vector consististing of numbers (attribute or feature) characterizing a particular object and an associated class label (the ground truth). A column corresponds to attribute or feature.

Train Set and Test Set

$$\left(\begin{array}{cc|c} H(cm) & W(kgr) & NBA \\ 190 & 100 & 0 \\ 195 & 110 & 0 \\ 180 & 85 & 0 \\ \dots & & \\ 178 & 75 & 1 \end{array}\right) \qquad \left(\begin{array}{cc|c} H(cm) & W(kgr) & NBA \\ 193 & 102 & ? \\ 180 & 87 & ? \\ 200 & 150 & ? \\ \dots & & \\ 168 & 65 & ? \end{array}\right)$$

Definition

A feature matrix is a 2D matrix where each row (instance or sample) is a feature vector consististing of numbers (attribute or feature) characterizing a particular object and an associated class label (the ground truth). A column corresponds to attribute or feature.

Classification Formulation

University of Victoria Computer Science

Input

Training vectors $\mathbf{x_i} \in R^d, i = 1, \ldots, n$ where i is the instance index, n is the number of instances, and d is the dimensionality of the feature vector. Associated ground truth classification labels can be written as integers y_i.

Classification Operations

University of Victoria Computer Science

Classifier

A classification algorithm typically supports 2 operations:

- **Train** takes as input a labeled training set (a 2D matrix of features and 1D vector of associated labels) and outputs a model (a representation of the classifier for that particular problem).
- **Predict** takes as input a trained model and an unlabeled testing set (2D feature matrix) and produces a predicted labeled test set (1D vector of labels).

Classification Operations

University of Victoria Computer Science

Classifier

A classification algorithm typically supports 2 operations:

- **Train** takes as input a labeled training set (a 2D matrix of features and 1D vector of associated labels) and outputs a model (a representation of the classifier for that particular problem).
- **Predict** takes as input a trained model and an unlabeled testing set (2D feature matrix) and produces a predicted labeled test set (1D vector of labels).

Evaluation

University of Victoria Computer Science

Evaluator

- **Evaluate** takes as input a vector of ground truth labels and a corresponding vector of predicted labels and calculates various classification performance metrics. Classifier is treated as a black box

Example

Classification accurancy can be computed as the percentage of labels that were correctly predicted i.e the ground truth label and the prediction label match.

Evaluation

University of Victoria Computer Science

Evaluator

- **Evaluate** takes as input a vector of ground truth labels and a corresponding vector of predicted labels and calculates various classification performance metrics. Classifier is treated as a black box

Example

Classification accurancy can be computed as the percentage of labels that were correctly predicted i.e the ground truth label and the prediction label match.

Confusion Matrix

This is an C by C matrix M in which each element $M_{i,j}$ shows the percentage of instances with ground truth class label i that were predicted as class label j. The diagonal element correspond to the correctly classified instances for each class. The confusion matrix reveals information about how classification and misclassification are distributed among different combinations of classes.

Example confusion matrix

Linear SVM classifier for automatic music genre classificaiton (80%) classification accuracy.

a	b	c	d	e	f	g	h	i	j	$\leftarrow$ *classified as*
79	0	4	4	0	2	6	0	3	2	$a = bl$
0	94	1	0	0	3	0	0	0	2	$b = cl$
5	0	74	1	0	1	3	0	3	13	$c = co$
1	0	3	74	2	0	1	4	6	9	$d = di$
2	0	0	5	78	0	4	3	8	0	$e = hi$
2	4	3	0	0	89	1	0	0	1	$f = ja$
3	0	0	5	1	0	83	1	0	7	$g = me$
0	0	10	5	4	0	0	72	2	7	$h = po$
3	0	5	8	11	1	0	2	68	2	$i = re$
3	0	17	9	0	0	8	2	6	55	$j = ro$

Generative Approaches to Classification

University of Victoria Computer Science

Main Idea

Transform the problem of classification to the multiple subproblems. Each subproblem consists of estimating the parameters of a model capable of generating samples similar to the ones associated with a particular class. In order to transform the problem and perform the model estimation we first have to review some concepts from probability and statistics.

Joint Probability Distribution

University of Victoria Computer Science

Joint Probability Distribution

Complete set of RVs used to describe the problem can be represented as the joint probability distribution. For example the joint distribution $P(Weather, Raincoat, Season)$ can be represented as a $2x2x4$ table.

Conditional Probability

University of Victoria Computer Science

Definition

$P(a/b)$ where a and b are propositions. The probability of a given that all that we know is b. The conditional probability can be defined in terms of unconditional probabilities as follows:

$$P(a/b) = \frac{P(a,b)}{P(b)} \tag{4}$$

Conditional Probability Notation

University of Victoria Computer Science

Notation

$$P(X = x_1, Y = y_1) = P(X = x_1/Y = y_1)P(Y = y1) \quad (5)$$
$$P(X = x_1, Y = y_2) = P(X = x_1/Y = y_2)P(Y = y2) \quad (6)$$
$$\ldots \quad (7)$$

can be combined with the notation denoting a set of equations:

$$\mathbf{P}(X, Y) = \mathbf{P}(X/Y)\mathbf{P}(Y) \quad (8)$$

Marginal, joint and conditional

	Male	Female
Engineering	150	50
Humanities	20	80

$$P(\text{X is Male}) = \frac{150 + 20}{150 + 20 + 50 + 80} = \frac{170}{300} \approx 0.57$$

$$P(\text{X is Female and X in Engineering}) =$$

$$P(\textit{Female}, \textit{Engineering}) = \frac{50}{300} \approx 0.17$$

$$P(\text{X is Female if we know that X is in Engineering}) =$$

$$P(\textit{Female}\ \textit{Engineering}) = \frac{50}{200} = 0.25$$

Bayes Rule

University of Victoria Computer Science

Definition

$$\mathbf{P}(Y|X) = \frac{\mathbf{P}(X/Y)\mathbf{P}(Y)}{\mathbf{P}(X)}$$

Example

Suppose L is a rv corresponding to people with lung cancer in a population and S is a rv corresponding to the smokers. We have the following data: $P(L) = 0.001$, $P(S/L) = 0.9$, $P(S/\hat{L}) = 0.21$. P(L/S) corresponds to the percent of smokers who have lung cancer and can be calculated using the Bayes theorem:

$$P(L/S) = \frac{P(S/L)P(L)}{P(S)} = \frac{0.0009}{0.9 * 0.001 + 0.21 * 0.999} = 0.0043$$

What's the big deal ?

Bayes theorem allows us to "choose" in a particular problem the conditional probabilities that are easier to calculate. For example it is easier to obtain the probability that someone who has lung cancer is a smoker than the probability that a smoker has lung cancer.

Bayes Classification

$$P(Y/\mathbf{X}) = \frac{P(\mathbf{X}/Y)P(Y)}{P(\mathbf{X})} \tag{9}$$

where Y is the class label and $\mathbf{X}$ is the feature vector. Notice that this is a set of equations, one for each class label in Y. Therefore there will be L posterior probabilities one for each class. To classify a test instance a *Bayesian* classifier computes these posterior probabilities and selects the class label corresponding to the maximum posterior. Main challenge becomes how to estimate $P(\mathbf{X}/Y)$ from the labeled training samples. For each class the corresponding training samples are used to estimate the parameters of the corresponding pdfs.

Bayes Classification

$$P(Y/\mathbf{X}) = \frac{P(\mathbf{X}/Y)P(Y)}{P(\mathbf{X})} \tag{9}$$

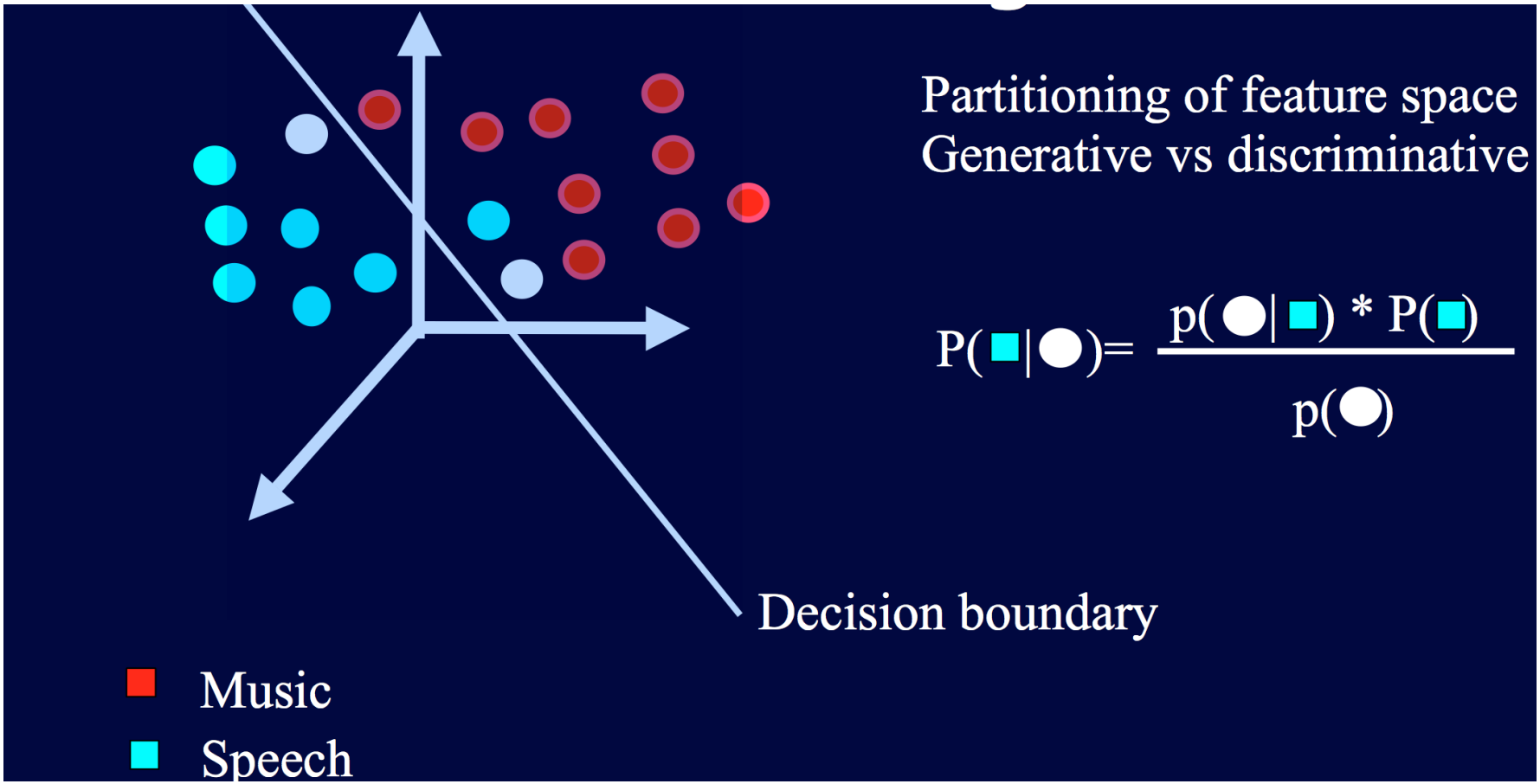
where Y is the class label and $\mathbf{X}$ is the feature vector. Notice that this is a set of equations, one for each class label in Y. Therefore there will be L posterior probabilities one for each class. To classify a test instance a *Bayesian* classifier computes these posterior probabilities and selects the class label corresponding to the maximum posterior. Main challenge becomes how to estimate $P(\mathbf{X}/Y)$ from the labeled training samples. For each class the corresponding training samples are used to estimate the parameters of the corresponding pdfs.

Bayes Classification

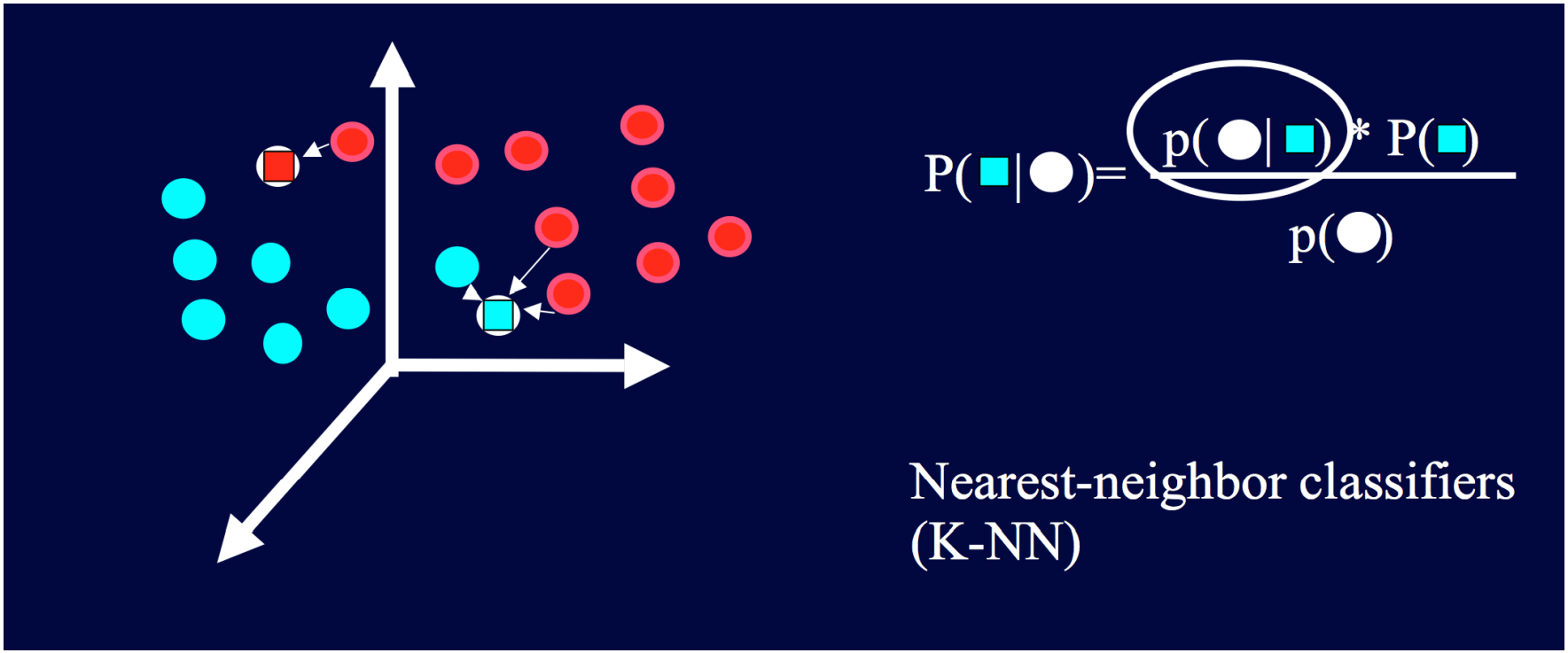
$$P(Y/\mathbf{X}) = \frac{P(\mathbf{X}/Y)P(Y)}{P(\mathbf{X})} \tag{9}$$

where Y is the class label and $\mathbf{X}$ is the feature vector. Notice that this is a set of equations, one for each class label in Y. Therefore there will be L posterior probabilities one for each class. To classify a test instance a *Bayesian* classifier computes these posterior probabilities and selects the class label corresponding to the maximum posterior. Main challenge becomes how to estimate $P(\mathbf{X}/Y)$ from the labeled training samples. For each class the corresponding training samples are used to estimate the parameters of the corresponding pdfs.

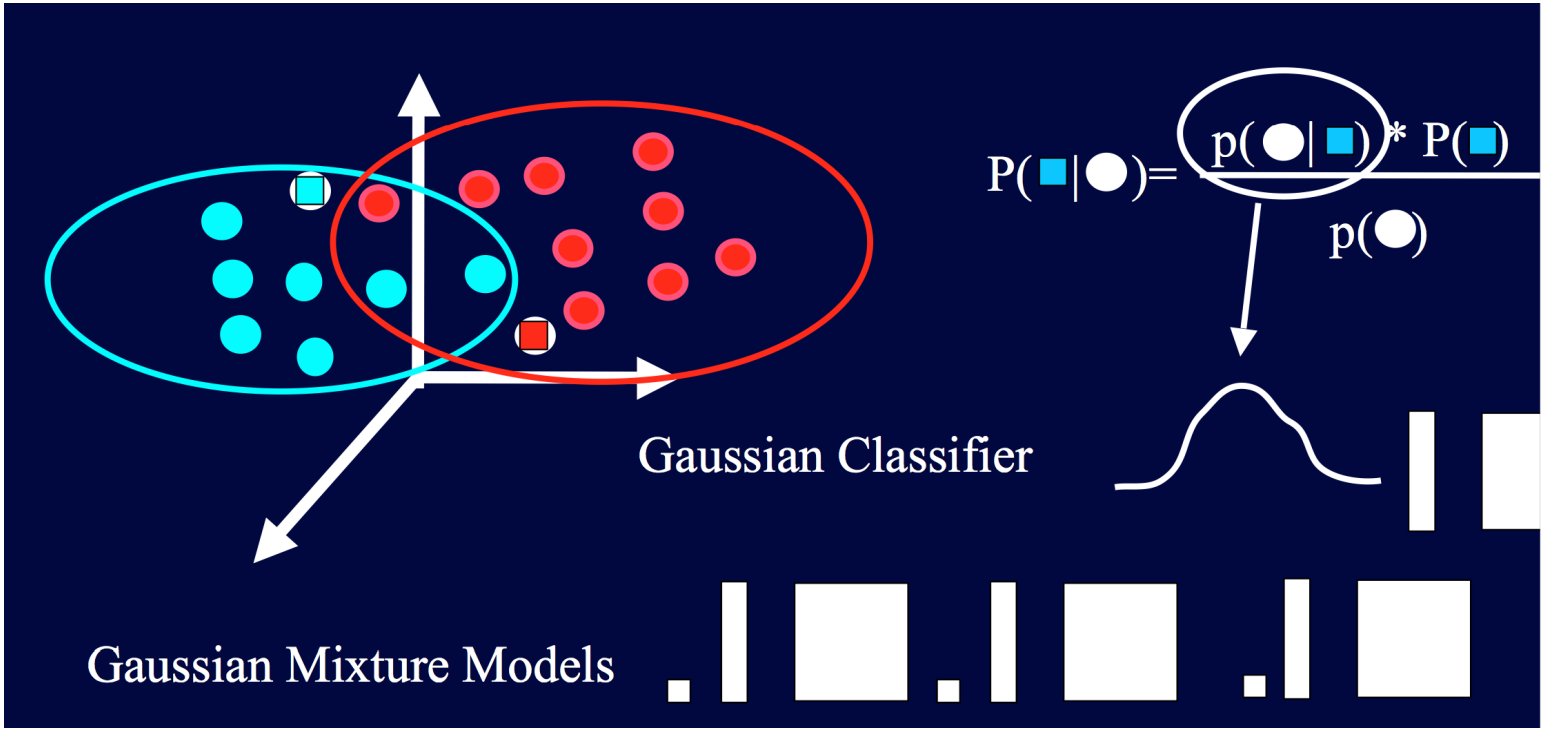
Bayes Classification



Bayes Classification - KNN



Bayes Classification - GNB GMM



Bayes Error Rate

University of Victoria Computer Science

Definition

Assume that we know the true probability distribution that governs $P(\mathbf{X}|Y)$. The Bayes error rate is the area in which the class distributions overlap. For a given feature space, the Bayes Error Rate is a lower bound on the error rate that can be achieved by any pattern classifier acting on that space. In general it can only be known directly if all the class priors and class-conditional likelihoods are known. The reason is that some of the error is inherent due to overlapping class densities but some additional error can creep in because of deficiencies in the classifier and limitations of the training data.

Conditional independence

University of Victoria Computer Science

Definition

Two variables A and B are said to be **conditionally independent** given another variables C if $P(A|C) = P(A|B, C)$.

For example, consider the probability that people are carrying ubmrellas and that there is heavy traffic. In general those are not independent as both traffic and umbrella usage will go up if it is raining. However if we know that it is raining then the probability of heavy traffic does NOT depend on whether there is increased umbrella usage (conditional independence).

Naive Bayes Classifier

University of Victoria Computer Science

Definition

A Naive Bayes classifier estimates the class-conditional probability bu assuming that the attributes are conditionally independent, given the class label y. More formally:

$$P(\mathbf{X}|Y=y) = \prod_{i=1}^{d} P(X_i|Y=y) \tag{10}$$

Example

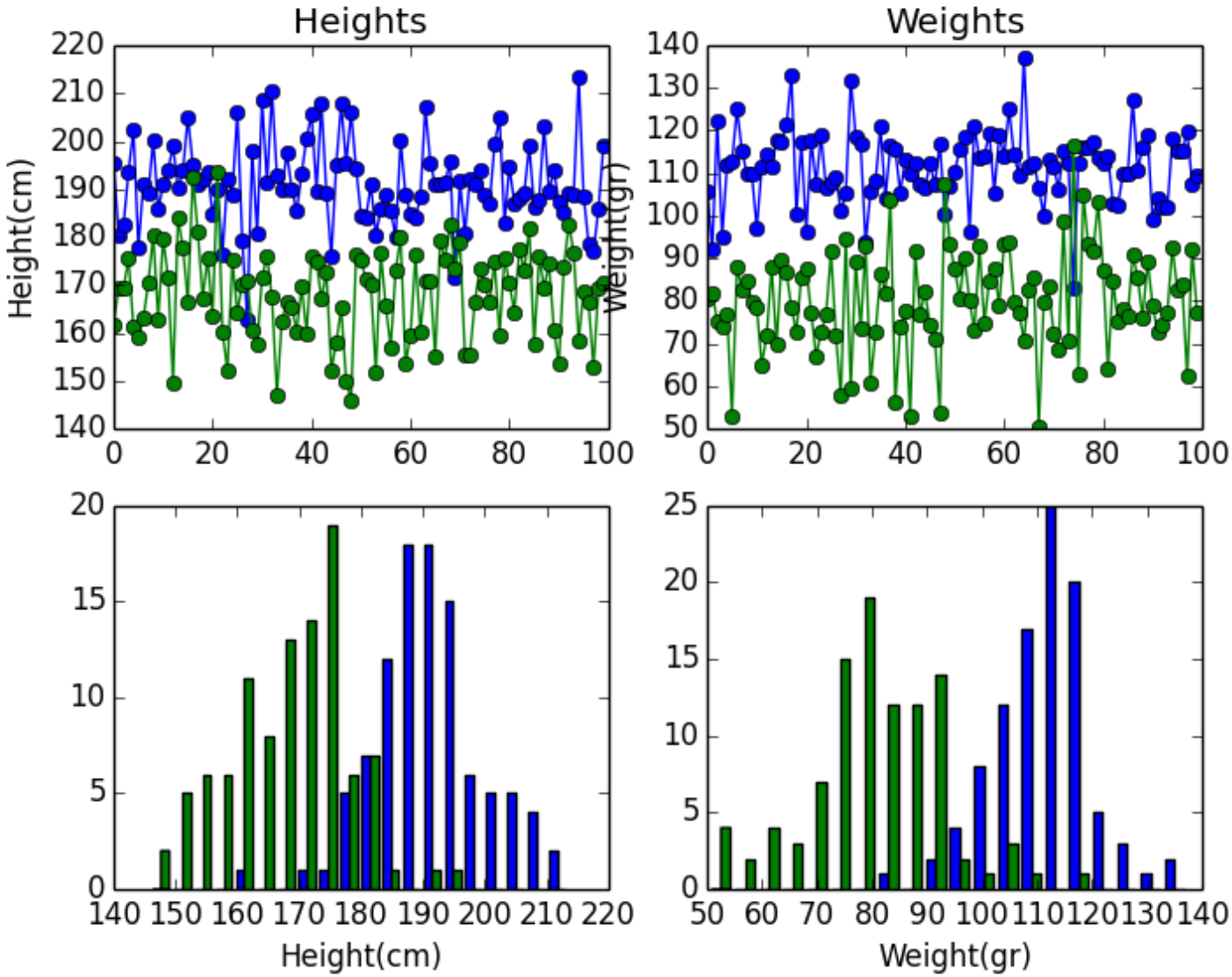
If we know that someone is a professional basketball player then the probability they have a certain height does not depend on their weight (this is the naive assumption). However the height of person does depend on their weight if we don't know whether they are basketball players or not.

Naive Bayes Classifier

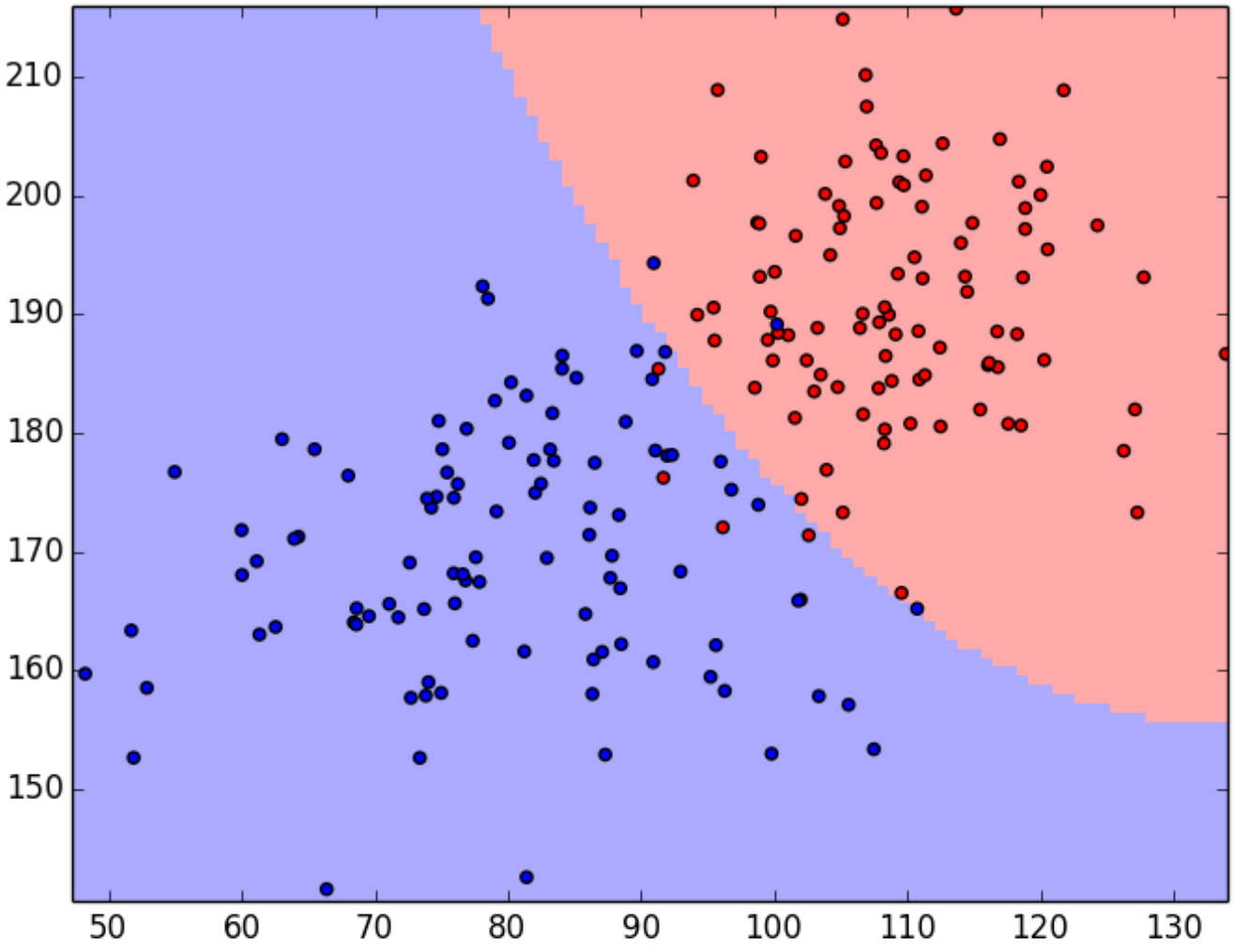
The naive assumption about conditional independence of the attributes allows us to model each attribute separately. For categorical attributes estimating the conditional probabilities is straightforward counting based on the training data. For the continuous attributes it is typically assumed that the probability density function follows a certain parametric form and the task is to estimate the corresponding parameters. A common choice is the Gaussian distribution characterized by two parameters, its mean μ, and variance σ^2. For each class y_il, the class-conditional probability for attribute X_i is:

$$P(X_i = x_i|Y = y_j) = \frac{1}{\sqrt{2\pi\sigma_{ij}}} \exp^{-\frac{(x_i-\mu_{ij})^2}{2\sigma_{ij}^2}} \quad (11)$$

The hoops dataset



The hoops dataset (Naive Bayes Gaussian)



Plot Generation

University of Victoria Computer Science

Steps for generating plot

- Plot training set points with colors indicating class labels
- Train classifier
- Predict the classes of a dense grid of points (pixels)
- Color based on the predicted class to visualize decision boundary

Plot Generation

University of Victoria Computer Science

Steps for generating plot

- Plot training set points with colors indicating class labels
- Train classifier
- Predict the classes of a dense grid of points (pixels)
- Color based on the predicted class to visualize decision boundary

Plot Generation

University of Victoria Computer Science

Steps for generating plot

- Plot training set points with colors indicating class labels
- Train classifier
- Predict the classes of a dense grid of points (pixels)
- Color based on the predicted class to visualize decision boundary

Plot Generation

University of Victoria Computer Science

Steps for generating plot

- Plot training set points with colors indicating class labels
- Train classifier
- Predict the classes of a dense grid of points (pixels)
- Color based on the predicted class to visualize decision boundary

Plot Generation

University of Victoria Computer Science

Steps for generating plot

- Plot training set points with colors indicating class labels
- Train classifier
- Predict the classes of a dense grid of points (pixels)
- Color based on the predicted class to visualize decision boundary

Characteristics of Naive Bayes Classifiers

University of Victoria Computer Science

- Robust to isolated noise points especially when there is a lot of data
- Robust to irrelevant attributes as $P(X_i/Y)$ becomes almost uniformly distributed and therefore has little impact to the posterior probability.
- Correlated attributes can degrade the performance
- Very fast to train and predict

Hands-on Bernoulli Naive Bayes

Jupyter notebook for exploring how to estimate the probability parameters of a simple Bernoulli Naive Bayes classifer for classifying genre based on lyrics.

Table of Contents I

Table of Contents II

EM-Algorithm

University of Victoria Computer Science

Definition

The Expectation Maximization (EM) algorithm is a technique that finds maximum likelihood estimates in parametric models with incomplete data. It has many applications in machine learning such as unsupervised clustering, gaussian mixture modeling, handling missing values, learning hidden (latent) variables in Bayesian Networks, semi-supervised learning and others.

EM-algorithm

University of Victoria Computer Science

Definition

The EM-algorithm uses a set of interlocking equations in which the solution to the parameters requires the values of the latent variables and vice versa. It is an iterative (and not guaranteed to be optimal) procedure that finds the maximum likelihood estimates of the parameter vector by repeating two steps:

- **E-step** Assuming a known model characterized by parameters θ estimate the expected values of the missing data.
- **M-step** Treating the estimated values of the missing data as actual values re-estimate using ML the parameter vector θ.
- Repeat the **E-step** and **M-step** until there is little change in the parameter vector θ.

Digression - Sufficient Statistics

University of Victoria Computer Science

Definition

A statistic is a single measure calculated on a set of samples. number

Definition

A sufficient statistic for a particular model and associated unknown parameter is a measure that provide all the information needed for computing the value of that parameter.

Example

For a normal distribution $N(\mu, \sigma^2)$ with $\theta = (\mu, \sigma^2)$ the sum of samples and sum of squares $(\sum x_i, \sum x_i^2)$ are sufficient statistics.

Semi-supervised

Basic idea:

- Build model with the labeled samples
- Classify unlabeled samples using the model
- Retrain model using the newly labeled samples using the classifier as ground truth
- Repeat the process until the labels don't change signficantly.

K-means

The *K-means* algorithms and it's many variants are classic point-based clustering algorithms that are based on a simple iterative scheme. Typically the number of desired clusters is provided but there are techniques for reasonable guesses.

- Randomly (or using a simple strategy) pick K initial points to be used as cluster centers.
- Assign each data point to the closest cluster center
- Calculate the mean values of the data points belonging to each cluster
- Set the cluster centers to be these mean values.
- Repeat the previous steps until there is little change in the cluster centers.

Gaussian Mixture Model

Linear super-position of Gaussians:

$$p(\boldsymbol{x}) = \sum_{k=1}^{K} p(k)p(\boldsymbol{x}/k)$$

where

$$\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

Sampling from a GMM

- Generate a uniform random number to determine whcih of K components to draw from a sample (based on the probabilities π_k.
- Generate a sample from the Gausssian $\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$.

Note: Equivalent procedure generate random samples $N * \pi_k$ from each component.

Fitting the GMM to data

Given the data points, we want to estimate:

- The mixing coefficients
- The means
- The covariances

If we knew which component generated each data point, we could use ML solution for each Gaussian separately. The problem is that the data is unlabelled i.e there is a *latent* or hidden variable.

Labeled vs unlabeled

University of Victoria Computer Science

Note: K-means gives a good set of initial clusters for the EM-algorithm

EM-algorithm

If we knew which point contributes to which Gaussian component, the problem can be solved more easily but we don't. So the idea is to guess these labels and proceed with the estimation. Then using the estimated model, relabel the points, and repeat the process until convergence.
Latent variable $z_{n,k}$ which is 1 if point n comes from the kth component Gaussian, and otherwise.

ML estimation

$$\ln p(\boldsymbol{X}|\pi,\boldsymbol{\mu},\boldsymbol{\Sigma}) = \sum_{n=1}^{N} ln\left(\sum_{k=1}^{K}\pi_k\mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_k\boldsymbol{\Sigma}_k)\right)$$

If oracle gaves us latent variables complete likelihood would be:

$$p(\boldsymbol{X},\boldsymbol{Z}|\boldsymbol{\mu},\boldsymbol{\Sigma},\pi) = \prod_{n=1}^{N}\prod_{k=1}^{K}\pi_k^{z_{n,k}}\mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)^{z_{n,k}}$$

and log-likehood looks better:

$$\ln p(\boldsymbol{X},\boldsymbol{Z}|\boldsymbol{\mu},\boldsymbol{\Sigma},\pi) = \sum_{n=1}^{N}\sum_{k=1}^{K} z_{n,k}\left(\ln\pi_k + ln\mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)\right)$$

EM algorithm

- Compute $p(Z|X,\theta)$, the posterior distribution over z given our current best guess at the values of theta
- Compute the expected value of the log likelihood $\ln p(X,Z|\theta)$ with respect to the distribution $P(Z,X,\theta)$
- Find new θ that maximizes the function
- Iterate

  Note: Softening the binary latent variables to continuous ones i.e the expected values of the latent variables

Hands-on example of GMM clustering

Jupyter notebook for showing a variety of GMM covariance types on a data set consists of three genres and two audio features (mean and standard deviation of spectral centroid)
Note: K-means is a special case of Gaussian Mixture Model clustering with equal covariance in each component.

Table of Contents I

Table of Contents II

Dimensionality Reduction

The goal of dimensionality reduction techniques is to transform a high dimensional set of features (or data points) to a space of lower dimensionality. There are several reasons dimensionality reduction is desired:

- Some machine learning algorithms have problems handling data of high dimensionality (curse of dimensionality)
- Can lower training (and predicition) time without significantly affecting the classification performance
- Useful for visualization purposes (2 and 3 dimensions)

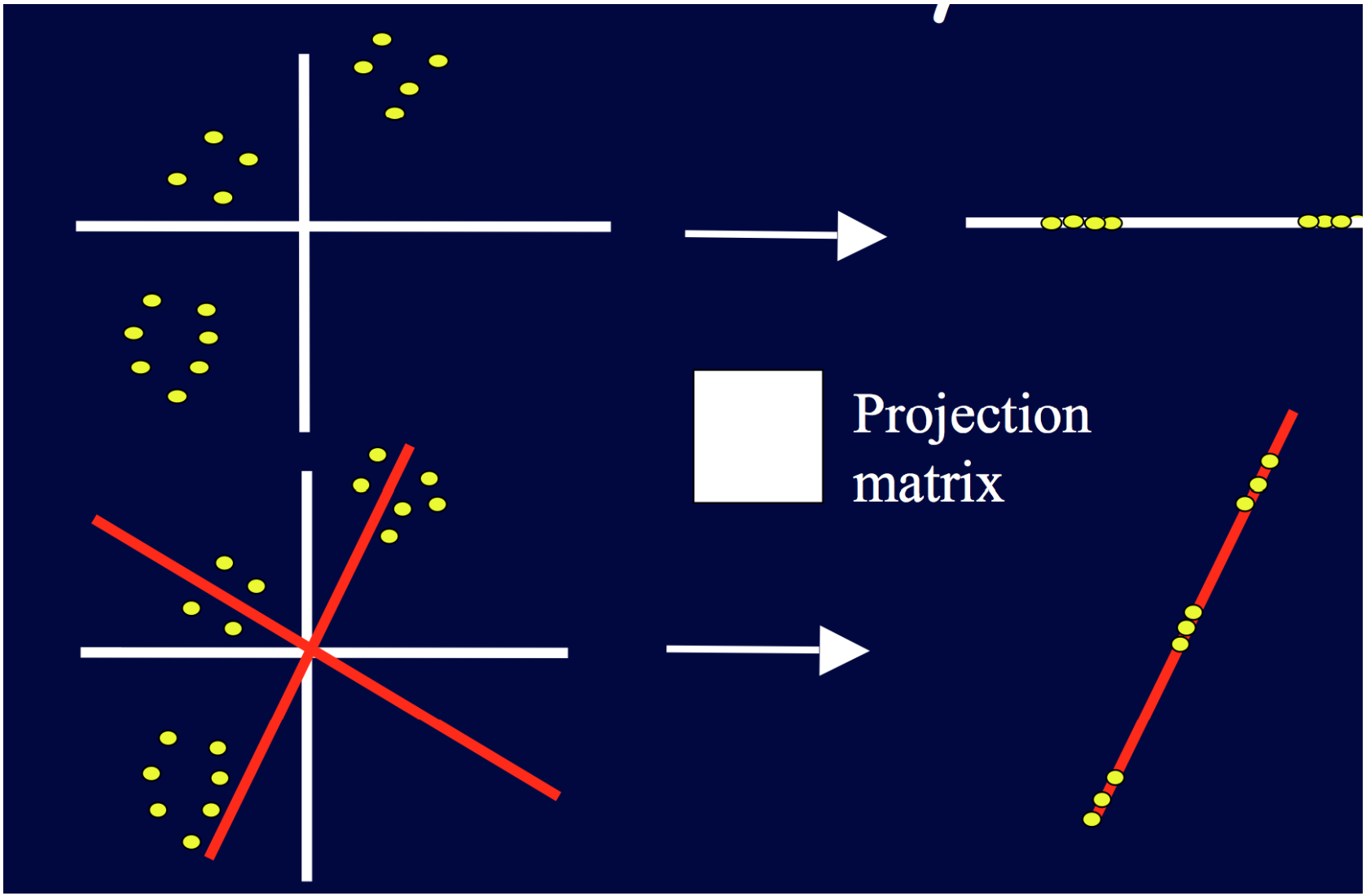
Principal Component Analysis

PCA is used to decompose a multivariate dataset in a set of successive orthogonal components that explain a maximum amount of the variance at each stage. It is an orthogonal linear transform that transforms the data to a new coordinate system. It is based on eigen-analysis of the sample covariance matrix.

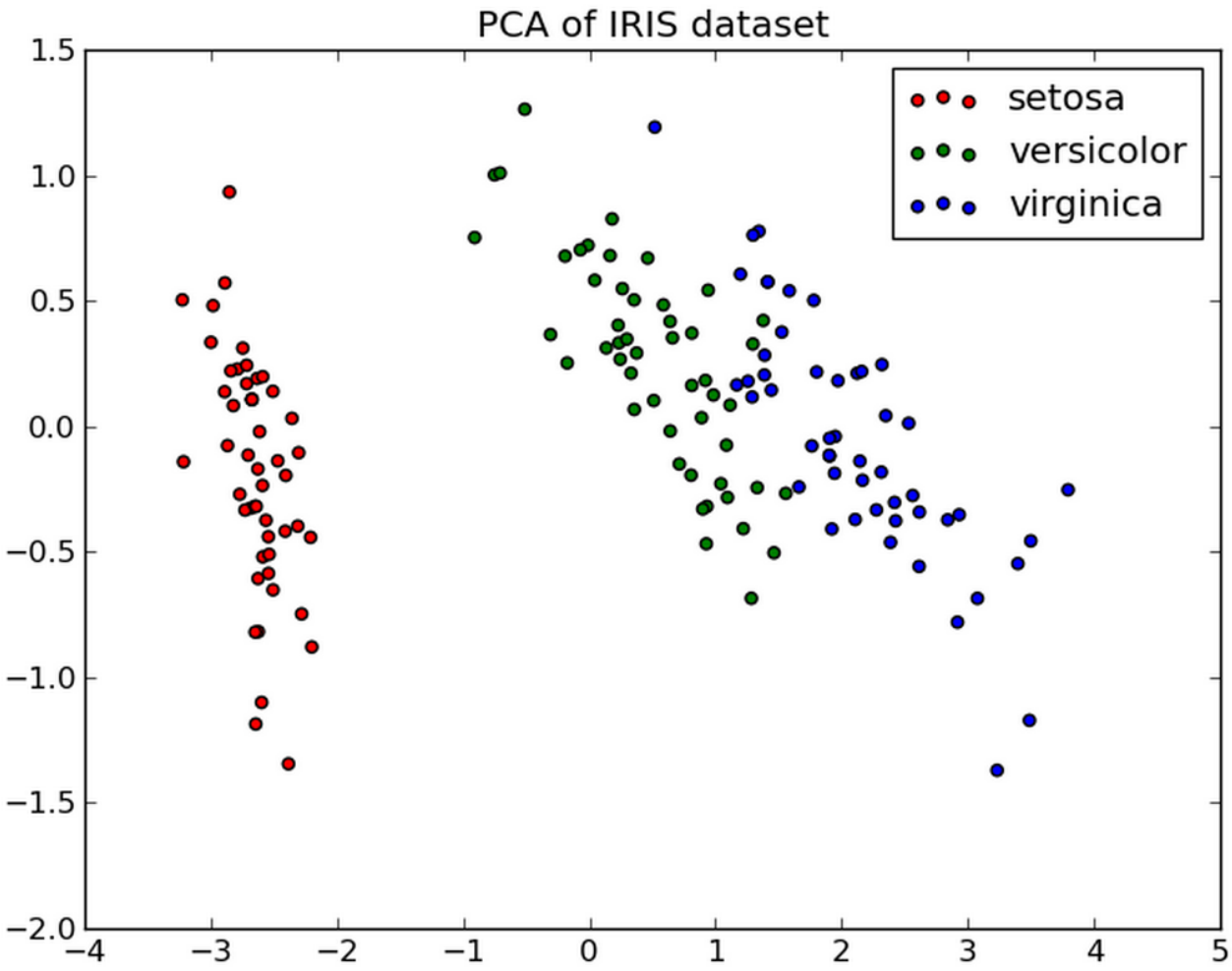
Probabilistic Principal Component Analysis

There is really nice formulation of PCA called **Probabilistic Principal Component Analysis** that was proposed in 1999 by M. Tipping and C. Bishop. The associated paper makes great reading for seeing how the topics covered in this tutorial such as the EM-algorithm for ML estimation can be applied to this particular problem.

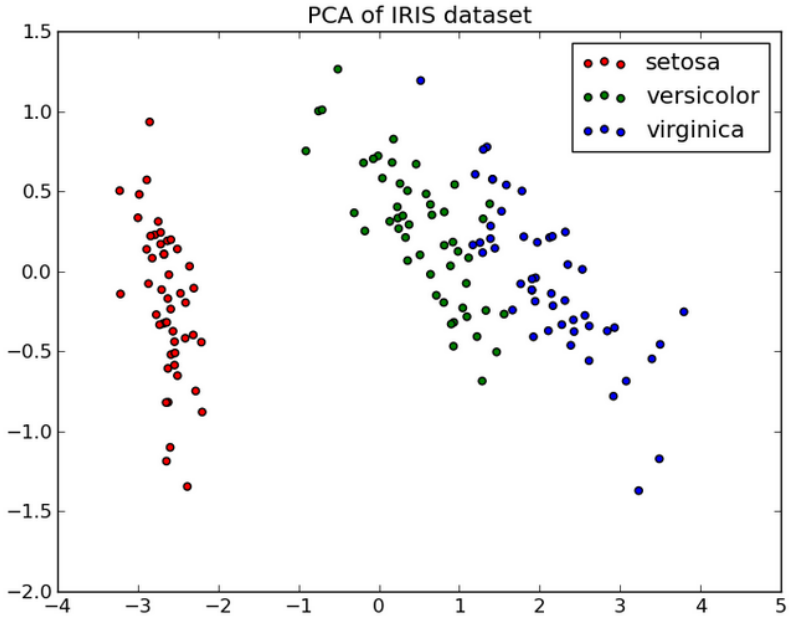
PCA geometry

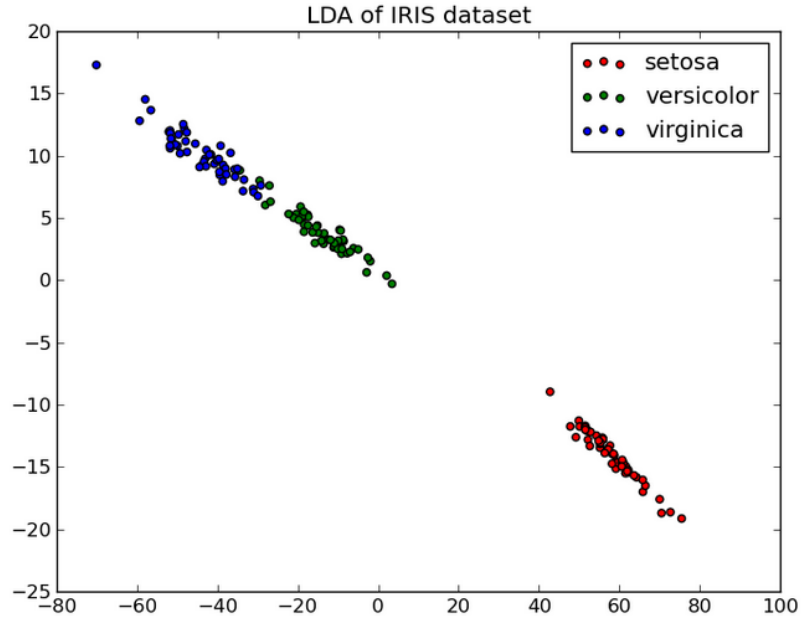


PCA on Iris dataset



PCA and LDA


PCA of IRIS dataset
setosa
versicolor
virginica
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
−4
−3
−2
−1
0
1
2
3
4
5


LDA of IRIS dataset
setosa
versicolor
virginica
20
15
10
5
0
−5
−10
−15
−20
−25
−80
−60
−40
−20
0
20
40
60
80
100

Linear Discriminant Analysis

Singular Value Decomposition

University of Victoria Computer Science

In PCA the principal components are the eigenvectors of the covariance matrix $\mathbf{X}\mathbf{X}^T$. Geometrically in two dimensions the sample covariance matrix expresses some rotation and scaling of the data. It is made made diagonal in the transformed space of the rotated axes. Eigevn values and eigen vectors are possible to compute because the covariance matrix is square and symmetric.
In SVD we directly decompose the data matrix into three matrices:

$$\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$$

In two dimenions it can be viewed as taking any arbitray matrix and converting it to an orthogonal matrix (rotation), a diagonal matrix (a stretch) and another orthogonal matrix (a second rotation).

Singular-Value Decomposition

Similarly to PCA, we can select a subset of the singular values and associated vector to create what is called a low rank approximation of the original matrix:

$$X : X \approx X_k = U_k \Sigma_k V_k^\top$$

After this operation, $U_k \Sigma_k^\top$ is the transformed training set with k features. To also transform a test set X, we multiply it with $V_k : X' = X V_k^\top$

Relationaship of PCA and SVD

It is not necessary to compute the full matrix SVD and then select the k singular values components and associated vectors as efficient algorithms can only compute the first k singular values.
This rank reduction is equivalent to PCA on the same data matrix with the means of every feature removed (centering around zero) - a standard requirement in PCA. Subtracting the mean feature values when the vectors are sparse can result in them loosing their sparsity therefore SVD is preferred in that case.

Latent Semantic Analysis (Indexing)

When truncated SVD is applied for text analysis (or indexing) it is termed latent semantic analysis. Roughly speaking it decomposes the word count (or tfidf) matrix into a term-concept vector matrix, a single value diagonal matrix, and a concept-document matrix.
In theory the vectors of the term-concept matrix summarize semantic information such as words (basketball, nba, Lebron) to hypothesized concepts (sports) and the vectors of the concept-document matrix summarize semantic information about which concepts (such as sports, or news) are contained in each document. To some extent this deals with the problem of synonyms and polynyms. Typically around 300 dimensions are used in text applications.

Probabilistic Latent Semantic Analysis

The motivation behind PLSA is similar to LSA but instead of using SVD to downsize the term-document matrix it is based on a probabilistic mixture decomposition based on a latent class model.
In this model the probability of a particular word w in a document d is modeled as a mixture of conditionally independent multinomial distribution:

$$P(w,d) = \sum_c P(c)P(d|c)P(w|c) = P(d)\sum_c P(c|d)P(w|c)$$

The parameters of the distributions are learned using the EM algorithm.

Some terminology: multinomial distribution

The multinomial distribution is a generalization of the binomial distribution.
An exampple: In a recent three-way election for a large country, candidate A received 20% of the votes, candidate B received 30% of the votes, and candidate C received 50% of the votes. If six voters are selected randomly, what is the probability that there will be exactly one supporter for candidate A, two supporters for candidate B and three supporters for candidate C in the sample?

Some terminology: Bayesian Estimation

Suppose that we want to estimate a parameter θ on the basis of a set of observations $\mathbf{x}$. Let $L(\mathbf{x}|\theta)$ be the likelihood of observing $\mathbf{x}$ when the parameter is θ. The maximum likelihood estimate of θ is:

$$\theta_{ML}(x) = \arg\max_{\theta} L(x|\theta)$$

Now assume that a prior distribution g over θ exists and instead of a fixed parameter we treat it as a random variable (Bayesian Statistics). Then the posterior distribution of θ is as follows:

$$f(\theta|x) = \frac{f(x|\theta)\, g(\theta)}{\int_{\vartheta \in \Theta} f(x|\vartheta)\, g(\vartheta)\, d\vartheta}$$

MAP estimation

The method of maximum a posterior estimation then estimates θ as the mode of the posterior distribution of this random variable:

$$\hat{\theta}_{\mathrm{MAP}}(x) = \arg\max_{\theta} \frac{f(x|\theta)\, g(\theta)}{\int_{\vartheta} f(x|\vartheta)\, g(\vartheta)\, d\vartheta} = \arg\max_{\theta} f(x|\theta)\, g(\theta).$$

When the prior distribution g is constant (uniform) then the MAP estimate coincides with the maximum likelihood estimate. For example both estimation methods the methods return the sample average as the estimate of the mean in the case of a Normal Distribution.

Latent Dirichlet Analysis

Latent Dirichlet Analysis is a topic modeling technique that is similar to pLSA. pLSA allows multiple topics in each document but the possible topic proportions are learned from the document collection. This means that as a generative model is has trouble modeling documents with arbitrary topic distributions that were not in the training collection. LDA introduced by David Blei in 2003 can be viewed as a Bayesian generalization of pLSA that allows for classifying new documents and controlling the number of topics.

Generative Model of LDA

University of Victoria Computer Science

- Choose number of words from Poisson distribution
- Choose vector of topic proportions according to a Dirichlet
- For each word in document
  - Choose topic from multinomial distribution over topics
  - Choose word from multinomial distribution over words conditioned on topic

Example of LDA

University of Victoria Computer Science

"POS tagging"	"Information Retrieval"	"Parsing"	"MT"	"Speech Recognition"	"Probabilistic Modeling"	"Experiments"	"Syntax"
pos	document	parsing	translation	speech	model	corpus	verb
tags	terms	parser	alignment	recognition	models	results	verbs
tagging	query	parse	word	spoken	probability	data	noun
sequence	term	treebank	english	language	training	number	case
tag	documents	accuracy	source	asr	data	table	syntactic
information	retrieval	parses	target	error	word	frequency	phrase
chunk	information	penn	translations	errors	language	test	clause
label	web	trees	machine	speaker	probabilities	average	structure
hmm	text	empty	phrase	utterances	set	found	phrases
learning	search	section	words	results	words	values	nouns
sequences	queries	wsj	language	turns	distribution	total	english
labels	system	proceedings	bilingual	rate	statistical	cases	subject
crf	collections	results	parallel	table	parameters	distribution	lexical

Inference

The LDA model needs to learn various distributions (topics, associated word probabilities, topic of each word, topic mixture in each document). This is a problem of Bayesian Inference for which methods such as variational Bayes or Gibbs sampling can be used.

Hands-on example of PCA, LDA

Example of PCA and LDA on 3 genres with high-dimensional audio features.

Table of Contents I

Table of Contents II

Markov Chains

A **Markov chain** is a sequence of random variables $X1, X2, X3, \ldots$ with the Markov property, namely that the probability of moving to the next state depends only on the present state and not on the previous states:

$$P(X_{n+1} = x | X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n) = P(X_{n+1} = x | X_n = x_n)$$

Transition Matrix

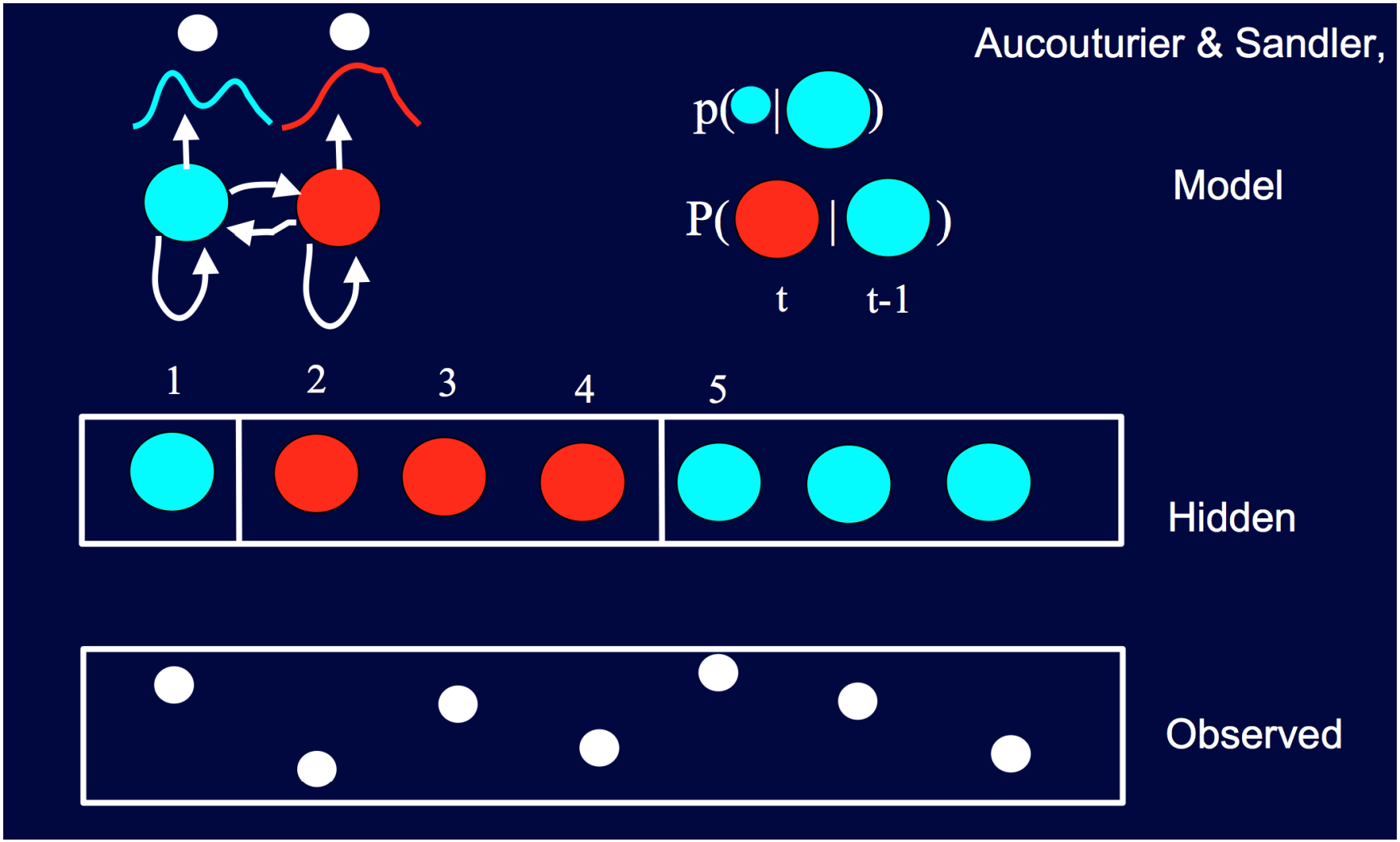
The possible values of Xi form a countable set S called the state space of the chain. A **Markov Chain** can be specified by a transition matrix with the probabilities of going from a particular state to another state at every time step.

Sequences of observations

There are many application areas and music is one of them in which we are interesting in modeling probability distributions over sequences of observations. We will denote the observation at time t by the variable Y_t. The variable can be a symbol from a discrete alphabet or a continuous variable and we assume that the observations are sampled at discrete equally-spaced time intervals so t can be an integer-valued time index.

Examples in music include the sequence of music intervals that form a melody or the sequence of chords in a song. In both of these cases the time can be relative to an underlying beat resulting into what is called a beat-synchronous representation.

Hidden Markov Models (HMMs) graphically



Hidden Markov Models (HMMs) assumptions

Properties:

- The observation at time t is generated by some random process whose state S_t is *hidden* from the observer.
- The hidden states form a Markov Chain i.e given the value of S_{t-1}, the current state S_t is independent of all states prior to $t-1$. The outputs also satisfy a Markov property which is that given state S_t, the observation Y_t is independent of all previous states and observations.
- The hidden state variable S_t is discrete

The joint distribution of an HMM

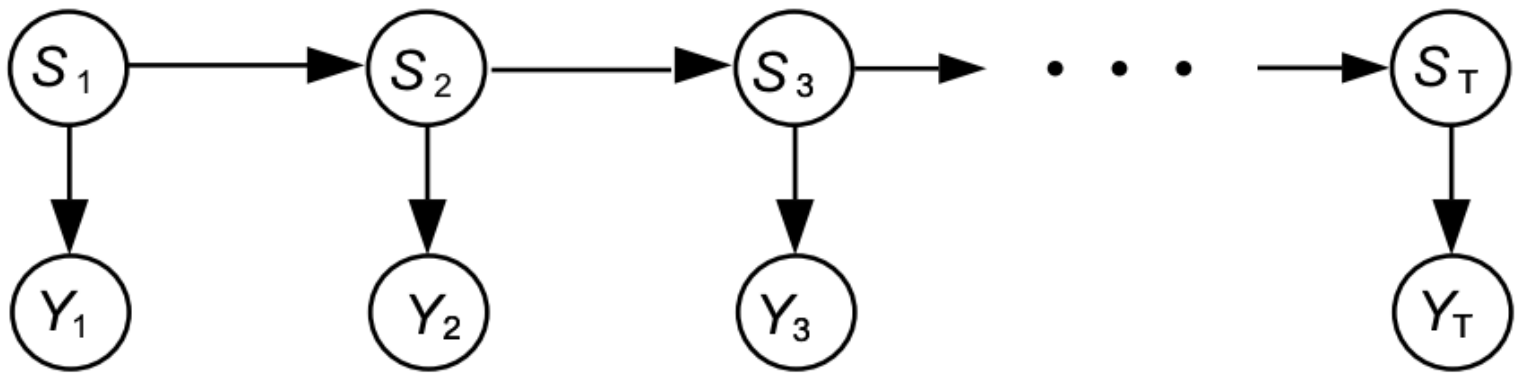
We can write the joint distribution of a sequence of states and observations by using the Markov assumptions to factorize:

$$P(S_{1:T}, Y_{1:T}) = P(S_1)P(Y_1|S_1)\prod_{t=2}^{T} P(S_t|S_{t-1})P(Y_t|S_t)$$

where the notation $X_{1:T}$ indicates the sequence $X_1, X_2, \ldots, X_T$.

Hidden Markov Models

We can view the Hiddean Markov Model graphically as a Bayesian network:



Specifying an HMM

So all we need to do to specify an HMM are the following components:

- A probability distribution over the intial state $P(S_1)$
- The K by K state transition matrix $P(S_t|S_{t-1})$, where K is the number of states
- The K by L emission matrix $P(Y_t|S_t)$ if Y_t is discrete and has L values, or the parameters θ_t of some form of continuous probability density function if Y_t is continuous.

Motivation for State Space Models

Imagine you are tracking a plane on a radar screen that appears as a blip that appears every few seconds. You are trying to estimate the state (positition and velocity for example) of a physical system from noisy observations over time. The problem can be formulated as inference in a temporal probability model, where the transition model describes the physics of motion and the sensor model describes the measurement process.

State Space Models

In state-space models, a sequence of D-dimensional real-valued observation vectors Y_t is modeled by assumign that at each time step Y_t was generated from a K-dimensional real-valued hidden state variable X_t, and the sequence of X's define a first-order Markov process:

$$P(X_{1:T}, Y_{1:T}) = P(X_1)P(Y_1|X_1)\prod_{t=2}^{T} P(X_t|X_{t-1})P(Y_t|X_t)$$

This factorization is identical to HMMs except that the discrete hidden S state variables are replaced by hidden continuous variables X.

Kalman Filter

The state transition probability $P(X_t|X_{t-1})$ can be decomposed into a deterministic and stochastic components:

$$X_t = f_t(X_{t-1}) + w_t$$

where f_t is the deterministic transition function, and w_t is zero mean random noise vector. Similarly the observation probability $P(Y_t|X_t)$ can be decomposed as

$$Y_t = g_t(X_t) + u_t$$

If these functions are linear and time invariant and the noise variables are Gaussian, then the model becomes a linear-Gaussian state-space model and it is called a **Kalman Filter** and is used extensively in signal processing.

Closed Property of HMMs and SSMs

What makes HMMs and SSMs special is that their hidden state spaces are closed under their respective transition probabilities and output models. By that I mean that the hidden state in HMMs is assumed to be a multinomial distribution and when the transition matrix is applied another multinomial distribution is obtained. In SSMs the hidden state is assumed to be Gaussian and after a linear transformation and Gaussian noise added it remains a Gaussian hidden state. This closed property makes learning and inference particularly simple and appealing in these models.

Hands-on example of Markov Chain and HMM

Jupyter notebook for exploring Markov Chains and Hidden Markov Models.

Table of Contents I

Table of Contents II

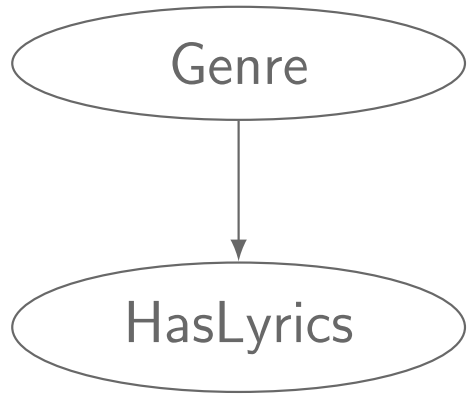
Bayesian Network

Definition

A Bayesian network (also known as belief network) is a probabilistic graphical model. It represents a set of random variables and their conditional dependencies in the form of a directed acyclic graph (DAG).

Bayesian Network Example

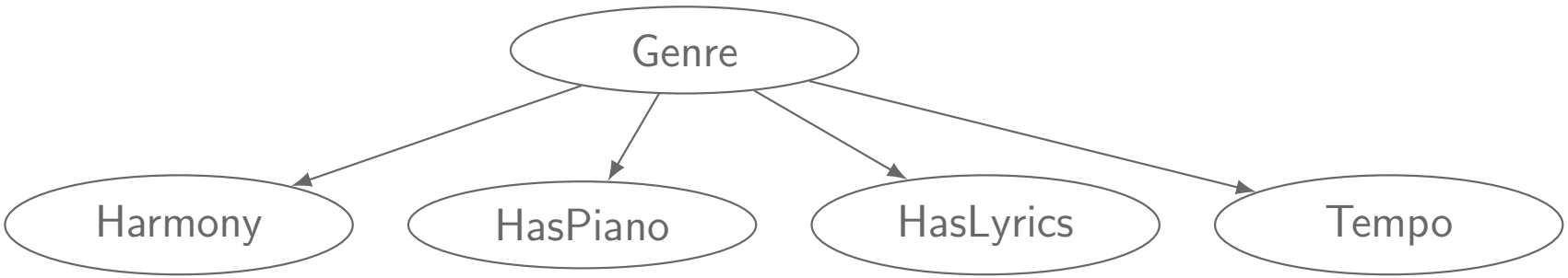
	HasLyrics	
HasLyrics	No	Yes
Country	0.2	0.8
Jazz	0.9	0.1



Genre	
Jazz	Country
0.3	0.7

Extending our BN example

This form with multiple features (causes) that are observed being conditionally independent of a class (in this case **genre**) corresponds to a Naive Bayes classifier. It also correspond to modeling a medical diagnosis from a set of symptoms.



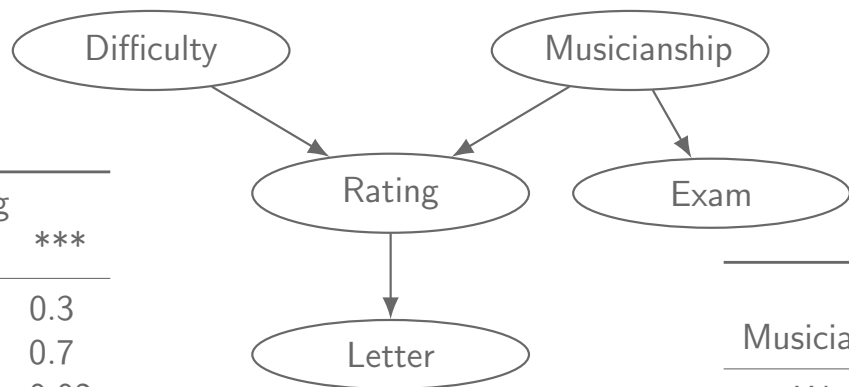
A more complex scenario

Consider the following scenario. An orchestra manager is trying to hire musicians. The candidate musicians audition by performing a set of pieces and are rated by judges. The rating of the judges is based on the musicianship (weak, strong) and the difficulty of the pieces they play (low, high). A consensus recommendation letter (weak, strong) is made stochastically by the judges based solely on their ratings. In addition, the candidate musicians provide their royal conservatory of music exam scores which only depend on their musicianship level. Let's try to express this problem using a Bayesian network.

A more complex BNN

Difficulty	
Low	High
0.6	0.4

Musicianship	
Weak	Strong
0.7	0.3



Difficulty	Musicianship	Rating *	**	***
Low	Weak	0.3	0.4	0.3
Low	Strong	0.05	0.25	0.7
High	Weak	0.9	0.08	0.02
High	Strong	0.5	0.3	0.2

	Exam	
Musicianship	Low	High
Weak	0.95	0.05
High	0.2	0.8

	Letter	
Rating	Weak	Strong
*	0.1	0.9
**	0.4	0.6
***	0.99	0.01

Exact Inference by enumeration

Consider how to compute the probability of a particular state for example the probability that the candidate has strong musicianship, the pieces played are not that difficult, the rating is 2 stars, the exam score is high, and the recommendation letter is weak. We can use the semantics of Bayesian networks to compute this joint probability as follows:

$$P(m = strong)P(d = low)P(r = **|m = strong, d = low)$$
$$P(e = high|m = strong)P(letter = weak|**)$$
$$= 0.3 * 0.6 * 0.08 * 0.8 * 0.4$$
$$= 0.004608$$

Any conditional probability distribution can be computed by summing terms from the full joint distribution.

Exact Inference

This is an example of the chain rule and can be more generally expressed:

$$P(D, M, R, E, L) = P(D)P(M)P(R|M, D)P(E|M)P(L|R)$$

Notice that we can compute the probability of any other event by summing over all possible values of the variables we are not interested in. For example how would we find P(M,G,E,L) ? How about P(M) ? Computing these multiple paths using the product and sum rules results in a lot of redundant computations. Variable elimination algorithms speed up this exact inference process by storing and re-using these common factors.

Probabilistic Reasoning

Consider a particular musician, George and we want to know how likely he will get a strong recommendation letter. Knowing nothing else this probability is about 50.2%. If we find out that George is not a strong musician then that probability goes down to around 38.9%.
In general we can use our Bayesian Network to answer any combination of observed, hidden, and query variables. This provides a flexible formalism that can be used to solve a variety of problems using the same underlying data and representation.

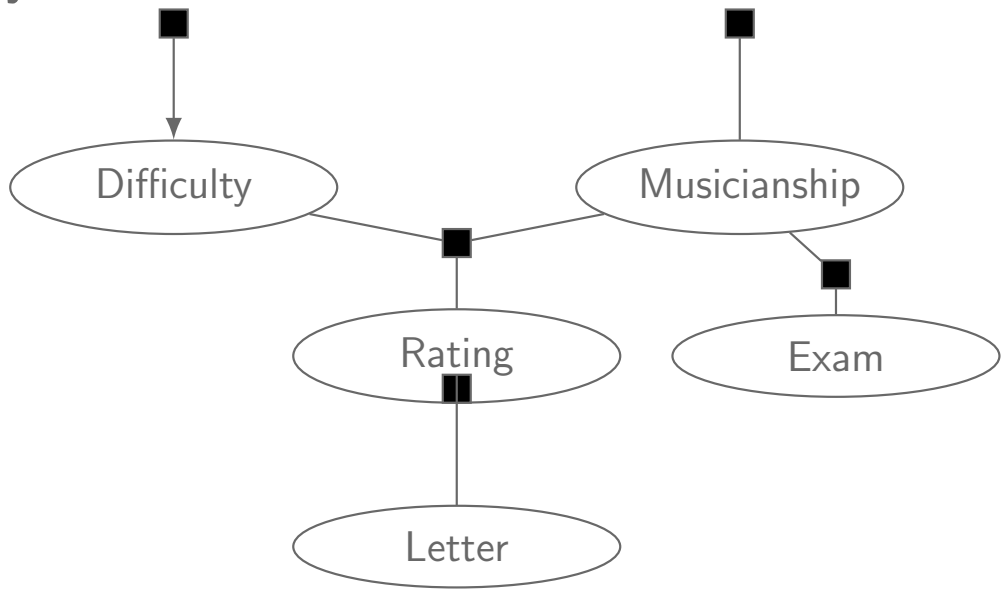
Answering a query

We are looking for the posterior probability distribution of a set of **query variables**, given values of some observed **evidence variables** and we don't care about the values of the remaining **hidden variables**.

$$P(X, e) = \alpha P(X, e) = \alpha \sum_{y} P(X, e, y)$$

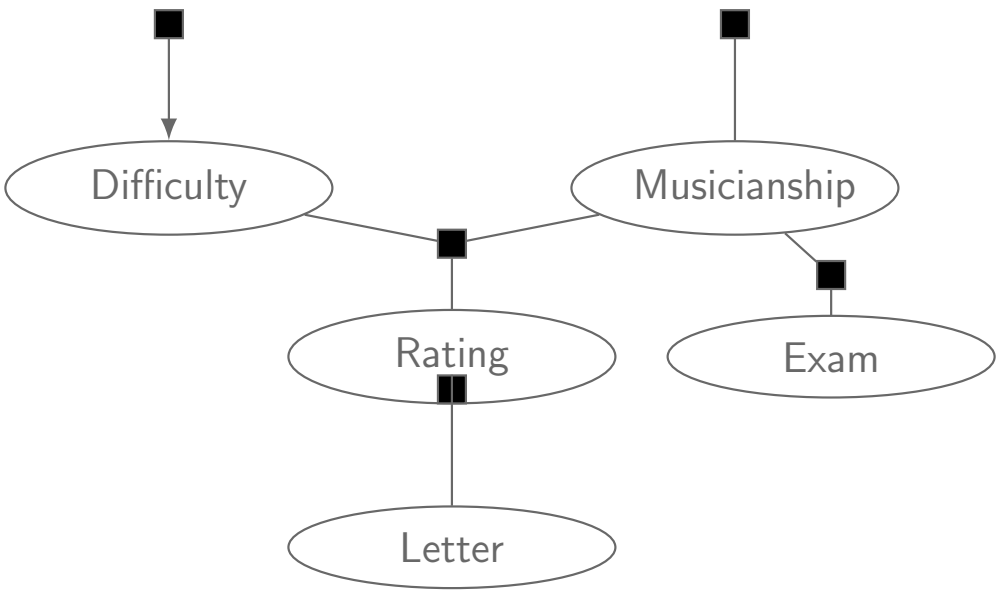
Factor Graphs

Algorithmically and visually it is possible to convert Bayesian Network graphs into factor graphs and run **factor graph propagation** making more clear the factorization of the joint probability distribution.
Each black square correpsonds to a CPT and connects the variables that are involved in each factor that is multiplied to form the joint probability distribution.



Exact Inference with message passing

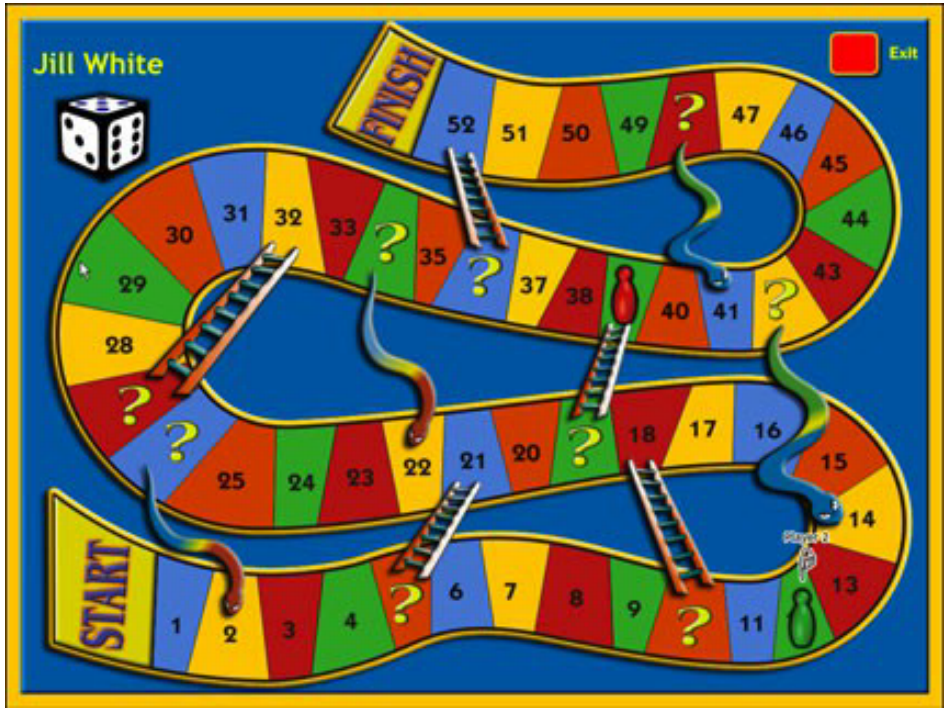
Exact inference in factor graphs can be performed by passing messages from the variable nodes to the factor nodes and vice versa. This is called *belief propagation*.



Approximate Inference

For certain types of networks (singly connected) exact inference is reasonable but for the general case it can be shown to be NP-hard so it is not practical in many cases. Approximate inference methods are based on randomized sampling algorithms, also called **Monte Carlo** algorithms that provide approximate answers whose accuracy depends on the number of samples generated and can deal with much larger networks.

A simple motivating example



- Draw N samples from sampling distribution
- Compute an approximate posterior
- Show that this approximate posterior converges to the true one

Direct Sampling

The basic idea is simple. Given a source of random numbers it is a simple matter to sample any discrete random variable. If the network has no evidence associated with it we can generate events by sampling each variable in turn in topological order. That way we always samples variables that are conditioned on values already assigned to the variable's parents.
The outcome of the the process is an event i.e an assignment of values to each random variable. The answers are computed by counting the actual samples generated. For example if we generate 1000 samples from the musician network, and 712 of them have $Exam = high$ then the estimated probability of $\hat{P}(Exam = high)$ is 0.711.

Direct Sampling Example

- Sample from $P(Difficulty) =< 0.6, 0.4 >$; suppose it returns *high*
- Sample from $P(Musicianship) =< 0.7, 0.3 >$; suppose it returns *weak*
- Sample from $P(Rating|Difficulty = high, Musicianship = weak) =< 0.9, 0.08, 0.02 >$; suppose it returns $**$
- Sample from $P(Exam|Musicianship = weak) =< 0.95, 0.05 >$; suppose it return *Low*
- Sample from $P(Letter|Rating = **) =< 0.4, 0.6 >$; suppose it return *Strong*

The randomly generated event will be:
$[Difficulty = high, Musicianship = weak, Rating = **, Exam = low, Letter = strong]$. Repeat to generate more samples.

Incorporating evidence - Rejection sampling

If we have some evidence we can still perform direct sampling and count the samples that are consistent with the evidence. Alternatively we can simply reject all samples that do not match the evidence and obtain the conditional estimate $\hat{P}(X = x|e)$ by counting how ofter $X = x$ occurs in the remaining samples. This is called **rejection sampling**. However if the events for which the evidence is true are rare then we will have to generate a lot of samples to get enough to compute this estimate.

Likelihood weighting

Likelihood weighting avoids this inefficiency of **rejection sampling** by generating only events that are consistent with the evidence. When this is done not all generated events are equal. Before tallying the counts in the distribution for the query variable, each event is weighted by the *likelihood* that the events accords to the evidence.

Markov Chain Monte Carlo (MCMC)

Unlike sampling algorithms that generate each event from scratch, MCMC generates each event by making a random change to the preceding event. It is useful to think of the network as being in a particular *state* specifying the value for each variable. The next state is generated by randomly sampling a value for one of the non-evidence variables X_i, conditioned on the current values of the variables in the Markov Blanket of X_i. MCMC wanders randomly around the state space, flipping one variable at a time *Gibbs sampling*, but keeping the evidence variables fixed. Many more variants exists but the basic principle of wandering in the state space is the same.

Note: The Markov Blanket of a variable X_i consists of its parents, children, and its children's other parents.

Hybrid Bayesian Networks

University of Victoria
Computer Science

Many real-world problems involve continuous quantities. A network with both discrete and continuous variables is called a **Hybrid Bayesian Network**. To specify a hybrid network, we need to specify two kinds of distributions: conditional distributions for continuous variables given discrete or continuous parents, and conditional distributions for discrete variables given continous parents. Various techniques that are beyond the scope of this tutorial can be used to integrate continuous variables. In some cases that we will examine the structure is such that it is obvious how to combine the continuous and discrete variables for example in Hidden Markov Models.

Dynamic Bayesian Networks

When we model phenomena over time we can view them as a sequence of snapshots. Each snapshot describes the state of the world at a particular time described by random variables some of which are observable $E_t = e_t$ and some of which are not X_t.
The **Markov assumption** is that for all t:

$$P(X_t|X_{0:t-1}) = (X_t|X_{t-1})$$

and

$$P(E_t|X_{0:t}, E_{0:t-1}) = P(E_t|X_t)$$

Inference tasks in temporal models I

- **Filtering:** we want to compute the posterior distribution over the current state, given all evidence to date. $P(X_t|e_{1:t})$. An almost identical calculation provides the likelihood of the evidence sequence $P(e_{1:T})$.
- **Prediction:** we want to computer the posterior distribution over the future state, given all evidence to date. $P(X_{t+k}|e_{1:t})$ for some $k > 0$.
- **Smoothing or hindsight:** computing the posterior distribution over a past state, given all evidence up to the present: $P(X_K|e_{1:t})$ for some $k < t$. It provides a better estimate of the state than what was available at the time, because it incorporates more evidence.

Inference tasks in temporal models II

- **Most likely explanation:**
  Given a sequence of observations, we might wish to find the sequence of states that is most likely to have generated these observations. That is we wish to compute: $\arg\max_{x_{1:t}} P(x_{1:t}|e_{1:t})$. This is the typical inference task in Speech Recognition using Hidden Markov Models.

Learning the transition and sensor models

In addition to these tasks, we need methods for learning the transition and sensor models from observations. The basic idea is that inference provides an estimate of what transitions actually occurred and what states generated the observations. These estimates can then be used to update the models and the process can be repeated. This is an instance of the **expectation-maximization (EM)** algorithm.

Sketch of filtering and prediction (Forward)

We perform **recursive estimation**. First the current state distribution is projected forward from t to $t+1$. Then it is updated using the new evidence e_{t+1}. We will not cover the details but it can be done by recursive application of Bayes rule and the Markove property of evidence and the sum/product rules.
We can think of the filtered estimate $P(X_t|e_{1:t})$ as a "message" that is propagated forward alogn the sequence, modified by each transition, and updated by each new observation.

Sketch of smoothing (Backward)

There are two parts to computing the distribution over past states given evidence up to the present. The first is the evidence up to k, and then the evidence from $k+1$ to t. The forward message can be computed as by filtering from 1 to k. Using conditional independence and the sum and product rules we can form a backward message that runs backwards from t. It is possible to combine both steps in one pass to smooth the entire sequence. This is, not surprisingly, called the Foward-Backward algorithm.

Finding the most likely sequence

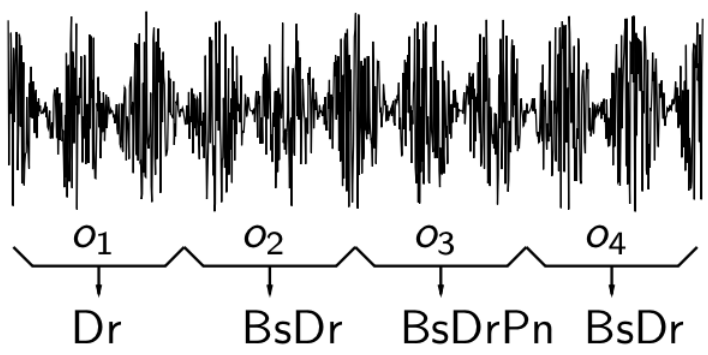
View each sequence of states as a path through a graph whose nodes are the possible states at each time step. The task is to find the most likely path through this graph, where the likelihood of any path is the product of the transition probabilities along the path and the probabilities of the given observations at each state. Because of the Markov property there is a recursive relationshtip between the most likely paths to each state x_{t+1} and most likely paths to each state x_t. By running forward along the sequence, and computing m messages at each time step we will have the probaiblity for the most likely sequence reaching each of the final states. Then we simply select the most likely one. This is called the **Vitterbi algorithm**.

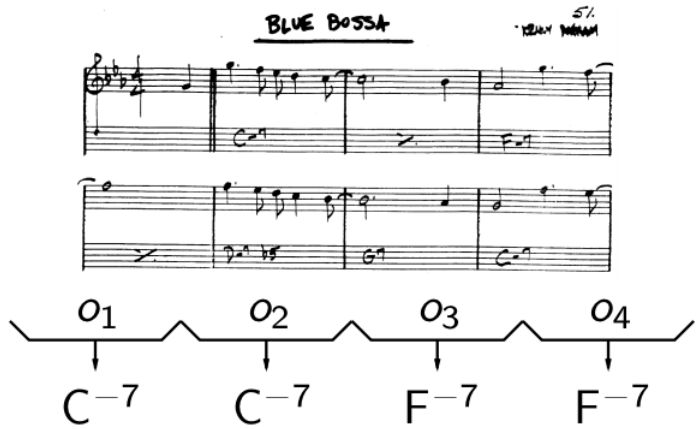
Connection to belief propagation

The forwardbackward algorithm in hidden Markov models (HMMs), and the Kalman smoothing algorithm in SSMs are both instances of belief propagation / factor graph propagation

Linear-chain data

In music it is common to have structured data that form a linear chain over time. Examples include music instrumentation prediction and chord recognition.





Notation for Linear Chain Data

- $\mathbf{x}$ is a sequence of observations
- $\mathbf{y}$ is the corresponding sequence of labels
- $\mathbf{D}$ is a training set of such N (i.i.d) such sequences $\mathbf{D} = \mathbf{x}^{(1)}, \mathbf{y}^{(1)}, \ldots, \mathbf{x}^{(N)}, \mathbf{y}^{(N)}$.

Feature Functions

University of Victoria Computer Science

Definition

A **feature function** is a real-valued function of both the input sequence (observations) and the output space (target labels). It can be used to compute characteristics of the observations and associated constrains.

Example:

$$f_j(o_i, y_i) = \begin{cases} 1 & \textit{if} \quad o_i = C, o_{i+1} = E \quad \textit{and} \quad y_i = Cmaj \\ 0 & \textit{otherwise} \end{cases}$$

Conditional Random Fields

University of Victoria Computer Science

Definition

CRF model definition

$$p(\mathbf{y}|\boldsymbol{x};\boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{x},\boldsymbol{\theta})} \exp \sum_{j=1}^{D} \theta_j F_j(\boldsymbol{x},\boldsymbol{y})$$
$$= \frac{1}{Z(\boldsymbol{x},\boldsymbol{\theta})} \Psi(\boldsymbol{x},\boldsymbol{y},\boldsymbol{\theta})$$

$Z(\boldsymbol{x},\boldsymbol{\theta})$ is called a **partition function**, and $\Psi(\boldsymbol{x},\boldsymbol{y},\boldsymbol{\theta})$ is called a **ponential function**. Feature functions can depend on the whole sequence of observation and labels.

Feature functions as constraints

University of Victoria Computer Science

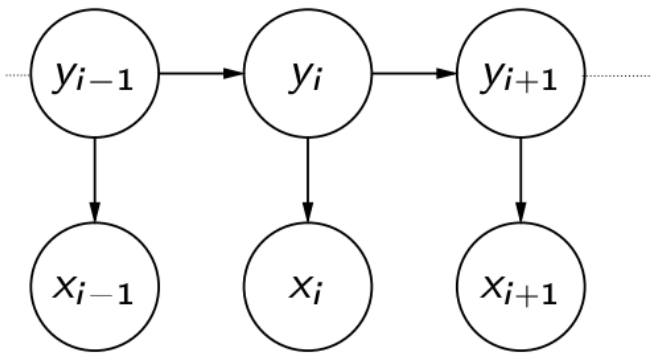
We can express varioust types of constraints through feature functions. For example for a linear-chain CRF each feature function depends on the whole sequence of observations but only on the **current** and **previous** labels. One can define **observation feature functions** and **transition feature functions**.
It can be shown that HMMs are a special case of linear chair CRFs with appropriately defined feature functions.

Differences of CRFs with HMMs

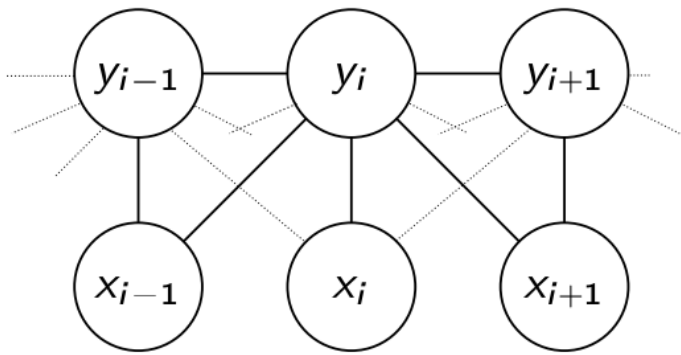
University of Victoria
Computer Science

CRFs have a number of advantages over HMMs based on the following differences :

- CRFs are discriminative models
- CRFs are undirected models



HMM



CRF

As is the case with any probabilistic model we can perform inference given a model and we need to learn the parameters of the model given a training set.

Hands-on Bayesian Network

Jupyter notebook for exploring Bayesian Networks and inference.

Table of Contents I

Table of Contents II

Estimation of Content Information from Audio Signals of Music: State-of-the-art

University of Victoria Computer Science

What are the limitation of existing approaches for estimating content information from music signals?

- Most existing computational models tend to focus on a single music attribute (chord, beat, structure, etc.).

  $\rightarrow$ It is contrary to the human understanding and perception of music that processes holistically the global musical context.

What is needed

University of Victoria Computer Science

What are the limitation of existing approaches for estimating content information from music signals?

- Dealing with real audio recordings requires the ability to handle both:
  - Rich probabilistic and
  - Complex relational structure at multiple levels of representation.

$\rightarrow$ Existing approaches for musical retrieval tasks fail to capture both of these aspects.

Probabilistic Graphical Models

University of Victoria Computer Science

Probabilistic graphical models are popular in MIR

- Hidden Markov models (HMM) are successful in modeling tasks where objects can be represented as sequential phenomena:
  - Chord estimation [Sheh & Ellis 2003], beat tracking [Klapuri *et al.* 2006] etc.
  - Limitation: hard to express dependencies in the data.
  - Other formalisms allow more complex dependencies:
  - Conditional random fields [Burgoyne *et al.* 2007], N-grams [Scholz *et al.* 2008] or tree structures [Paiement *et al.* 2005].

Limitations of probabilistic graphical models

University of Victoria Computer Science

- Although probabilistic models can handle the inherent uncertainty of audio, most of them fail to capture important aspects of higher-level musical relational structure and context.
- This aspect has been more specifically explored within the framework of Logic.

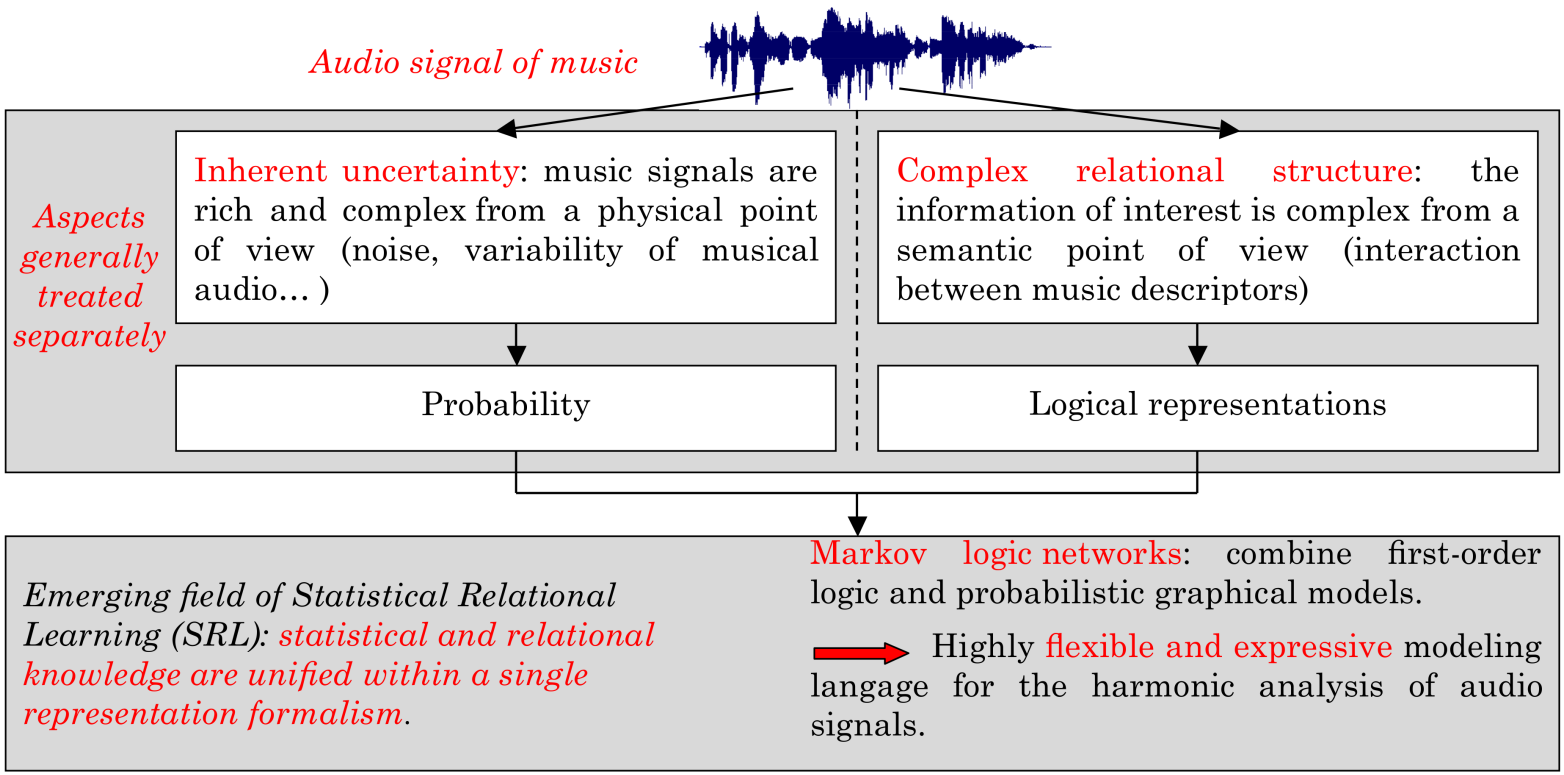
Logic Frameworks and expressiveness

University of Victoria Computer Science

- Modelling music rules is compact and human-readable:
  - Intuitive description of music.
  - Knowledge (e.g. music theory) can reflected in rules: *"CM chord* $\wedge$ *first degree* $\Rightarrow$ *CM key"*
- Inductive Logic Programming (ILP) [Muggleton 1991] that combines logic programming with machine learning has been used to model and learn music rules:
  - Harmony characterization [Anglade & Dixon 2008].
  - Expressive music performance [Ramirez *et al.* 2007].
- Limitations:
  - Logic-based approaches have focused on symbolic representations (e.g. MIDI).
  - Applying these approaches to audio generally requires a transcription step.

Markov Logic for the Analysis of Music Signals

Markov Logic Networks (MLNs) [Richardson & Domingos 2006] as an expressive formalism to estimate music content information from audio signals.



Markov Logic for Modeling Chord Progressions

University of Victoria Computer Science

Problem: Estimate the harmonic progression of an audio music signal at the analysis-frame level, taking into account a more global semantic level. Audio signals contain music notes that correspond to a chord sequence over time.

- What are these notes? (physical complexity)
- Which chords do they represent? (semantic complexity)
- Segmentation: chord labels related to the global semantic structure (complex relational structure)

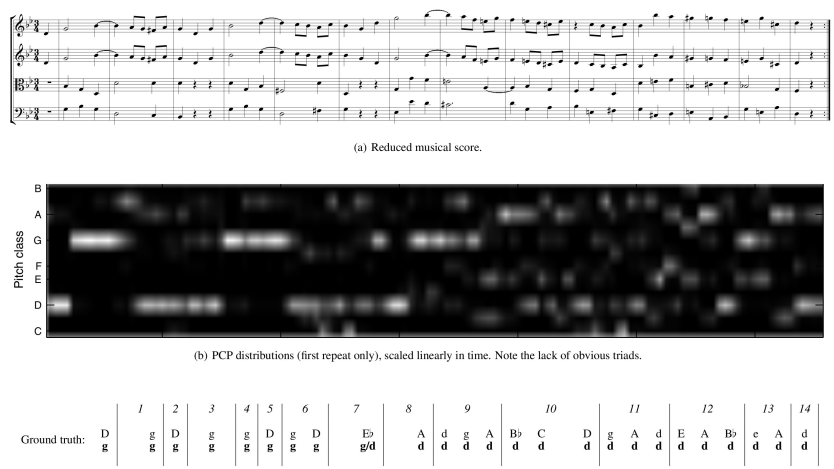


Figure: Adapted from [Burgoyne and Saul 2005].

Structurally consistent mid-level representation of music

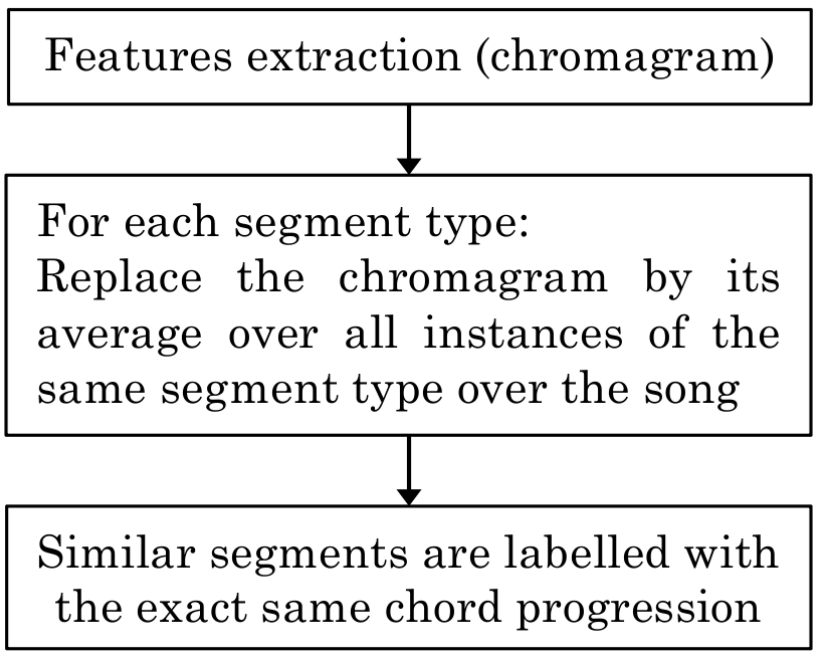
Considered problem: Estimating the harmonic progression of an audio music signal considering several time-scales of representation.

- Music pieces are structured at several time scales, from musical phrases to longer sections that generally have multiple occurrences.
- Here, popular music: pieces can be segmented into specific repetitive segments with labels such as *chorus*, *verse*, or *refrain*.
- Segments are considered as similar here if they represent the same musical content.
- In particular, two same segment types are likely to have similar harmonic structures.

Structurally consistent mid-level representation I

Previous work: music structure as a constraint:

- [Dannenberg 2005]: constraint to a beat tracking program.
- [Mauch *et al.* 2009]: context information for chord estimation.



Limitations:

- Repeated segments often present variations between several occurrences.
- An instance of blurred segment features (e.g. noise or percussive sounds) will automatically affect all same segments.

Proposed Approach

Using prior structural information to enhance chord estimation in a more elegant and flexible way using Markov Logic Networks.

- The model is not constrained to have the exact same chord progression in all sections of the same type (no hard constraint) .
- But we only favor same chord progressions for all instances of the same segment type.
- Potential of incorporating more context information and discovering new predicates.

Hypothesis

- Song segmentation in beats and structure is given
- Similar segment instances have the same length in beats.

Markov Logic Networks (MLN): short overview

A Markov Logic Network is a set of weighted first-order logic formulas that can be seen as a template for the construction of a probabilistic graphical model.

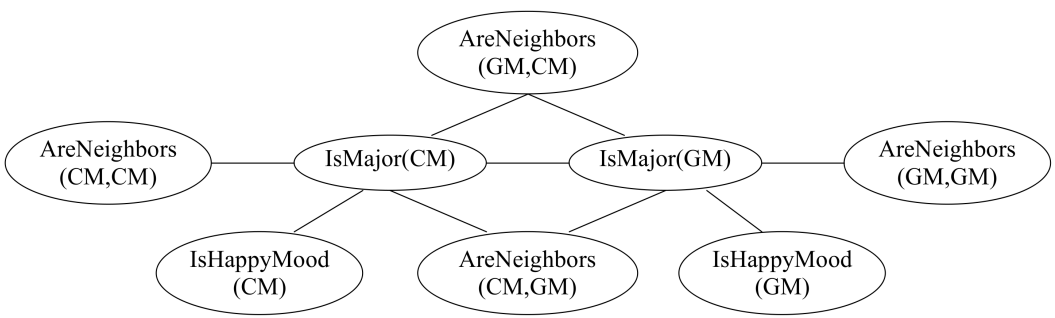
A first-order knowledge base (KB) is a set of formulas in first-order logic that can be seen as a set of hard constraints on the set of possible worlds.

- First-order KB: if a world violates even one formula, it has zero probability.
- In *real world schemes*, logic formulas are *generally* true, but *not always*.

MLN overview II

Basic idea in Markov logic: to soften these constraints to handle uncertainty. The weights reflect how strong a constraint is.

Knowledge	*Logic formula*	*Weight*
A major chord implies an happy mood.	$\forall$ x IsMajor(x) $\Rightarrow$ IsHappy-Mood(x)	$w_1 = 0.5$
If two chords are neighbors, either the two are major chords or neither are.	$\forall$ x $\forall$ y AreNeighbors(x, y) $\Rightarrow$ (IsMajor(x) $\Leftrightarrow$ IsMa-jor(y))	$w_2 = 1.1$



Example of a first-order KB and corresponding weights in the MLN.

Ground Markov network obtained by applying the formulas to the constants CM and GM chord.

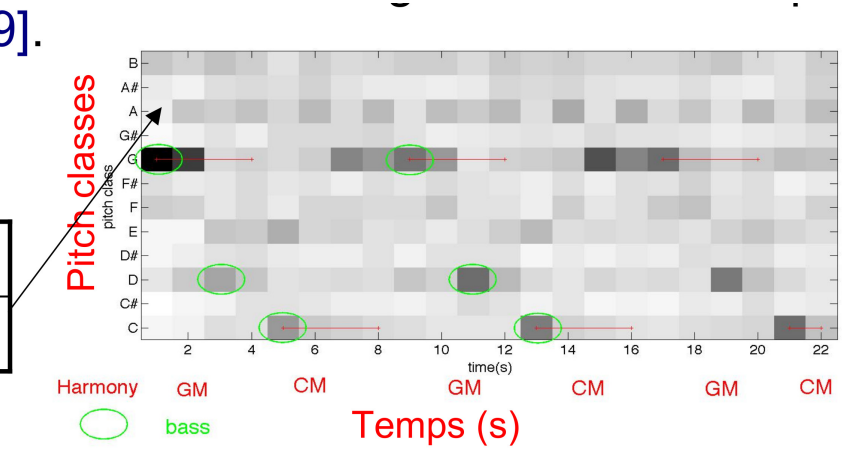
Modeling harmonic content of audio signals

University of Victoria Computer Science

Chroma representation: Mapping of spectrum to a 12-dimensional vector representing the intensity of the 12 semitone pitch classes.

[Fujishima 1999, Wakefield 1999].

Freq. (Hz)	55	110	220	440	880	1760
Note	A1	A2	A3	A4	A5	A6



Beat-synchronous chroma feature: observation features related to the meter (beat tracking algorithm [Peeters 2007]).

HMM Chord Detection

University of Victoria Computer Science

Classical approach: Hidden Markov Model (HMMs) have been widely used to model the chord progression of an audio file [Sheh & Ellis 2003], [Bello & Pickens 2005], [Lee 2006] etc.

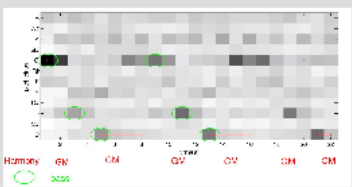
Baseline HMM for Chord Estimation [Papadopoulos & Peeters 2011]

FEATURE EXTRACTION

Audio mono signal 11025 Hz

↓

Beat-synchronous 12-dimensional chroma vectors

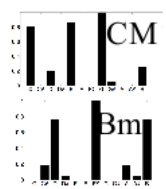


MODEL

24-state HMM where each state S_n represents one of the 24 major and minor triads $c_i, i \in [1,24]$
Observed : succession of N chroma vectors $o_n, n \in [0, N-1]$

Prior distribution of states $P(s_0)$: uniform.

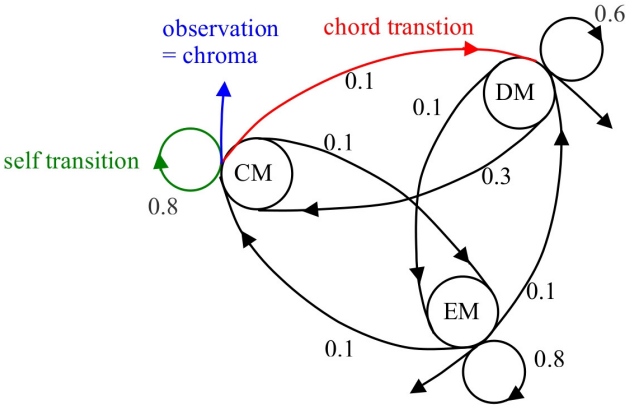
Conditional observation probabilities: $P(o_n \mid s_n)$
Obtained by correlation between chroma vectors and chord templates considering higher harmonics.



Transition probabilities $P(s_n \mid s_{n-1})$: based on music knowledge.

HMM Chord Detection

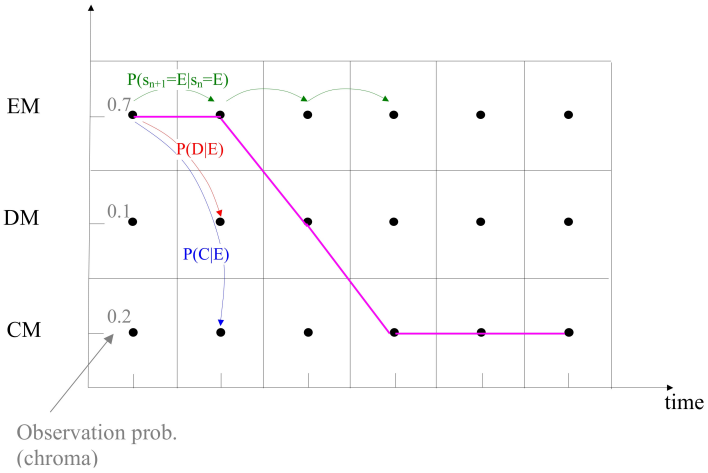
Classical approach: Hidden Markov Model (HMMs):



$p(s_{n+1}\|s_n)$	C	D
C	0.8	0.1
D	0.3	0.6
E	0.1	0.1

s_n / s_{n+1}

CCCCCDDCCCEEEEEDC(



- The observation (chroma) tells us something about the state.
- The transitions between states follows a transition probability distribution

The chord progression over time is estimated in a maximum likelihood sense using the Viterbi decoding algorithm.

From HMMs to MLNs

The chord progression can be equivalently modeled in the MLN framework using three generic weighted logical formulas that reflect the constraints defining the HMM.

Predicate declarations	
$Chord(chord!, time)$ $Observation(chroma!, time)$ $Succ(time, time)$	
Weight	*Formula*
Prior observation chord probabilities:	
$log(P(CM(t=0)))$ $\cdots$ $log(P(Bm(t=0)))$	$Chord(CM, 0)$ $\cdots$ $Chord(Bm, 0)$
Probability that the observation (chroma) has been emitted by a chord:	
$log(P(o_0\|CM))$ $log(P(o_0\|C\#M))$ $\cdots$ $log(P(o_{N-1}\|Bm))$	$Observation(o_0, t) \wedge Chord(CM, t)$ $Observation(o_0, t) \wedge Chord(C\#M, t)$ $\cdots$ $Observation(o_{N-1}, t) \wedge Chord(Bm, t)$
Transition probability between two successive chords:	
$log(P(CM\|CM))$ $log(P(C\#M\|CM))$ $\cdots$ $log(P(Bm\|Bm))$	$Chord(CM, t_1) \wedge Succ(t_2, t_1) \wedge Chord(CM, t_2)$ $Chord(CM, t_1) \wedge Succ(t_2, t_1) \wedge Chord(C\#M, t_2)$ $\cdots$ $Chord(Bm, t_1) \wedge Succ(t_2, t_1) \wedge Chord(Bm, t_2)$

Structural information is incorporated using the time temporal predicate $Succ(t_1, t_2)$ that indicates the position of successive frames.

A chroma feature is observed at each time frame:
$Observation(o_0, 0) \cdots$
$Observation(o_{N-1}, N-1)$
The temporal order of the frames is known:
$Succ(1, 0) \cdots$
$Succ(N-1, N-2)$

Chord Progression: Global Semantic Structure

University of Victoria Computer Science

Flexibility of MLN: *Prior global structural information*

Formulas added to express the constraint that *two same segment types are likely to have a similar chord progression*.

Predicate declarations	
$Observation(chroma!, time)$ $\quad$ $Succ(time, time)$ $Chord(chord!, time)$ $\quad$ $SuccStr(time, time)$	
Weight	*Formula*
Prior observation chord probabilities:	
$log(P(CM(t=0)))$ $\cdots$ $log(P(Bm(t=0)))$	$Chord(CM, 0)$ $\cdots$ $Chord(Bm, 0)$
Probability that the observation (chroma) has been emitted by a chord:	
$log(P(o_0\|CM))$ $\cdots$ $log(P(o_{N-1}\|Bm))$	$Observation(o_0, t) \wedge Chord(CM, t)$ $\cdots$ $Observation(o_{N-1}, t) \wedge Chord(Bm, t)$
Transition probability between two successive chords:	
$log(P(CM\|CM))$ $\cdots$ $log(P(Bm\|Bm))$	$Chord(CM, t_1) \wedge Succ(t_2, t_1) \wedge Chord(CM, t_2)$ $\cdots$ $Chord(Bm, t_1) \wedge Succ(t_2, t_1) \wedge Chord(Bm, t_2)$
Probability that similar segments have the same chord progression:	
w_{struct} w_{struct} $\cdots$ w_{struct}	$Chord(CM, t_1) \wedge SuccStr(t_2, t_1) \wedge Chord(CM, t_2)$ $Chord(C\#M, t_1) \wedge SuccStr(t_2, t_1) \wedge Chord(C\#M, t_2)$ $\cdots$ $Chord(Bm, t_1) \wedge SuccStr(t_2, t_1) \wedge Chord(Bm, t_2)$

The predicate *SuccStr* allows considering wider windows, as opposed to consecutive frames via the *Succ* predicate.

A chroma feature is observed at each time frame:

$Observation(o_0, 0) \cdots$

$Observation(o_{N-1}, N-1)$

The temporal order of the frames is known:

$Succ(1, 0) \cdots$

$Succ(N-1, N-2)$

Prior information about position of same segment type in the structure is given:

$SuccStr(1, 10)$

$SuccStr(2, 11) \cdots$

Inference

University of Victoria Computer Science

How to estimate the chord progression?

- Query: Find the chord progression Given:
- The structure of the domain: represented by a set of logical formulas (rules) F_i, with attached weight w_i.
- A set of evidence literals (chroma observations, position of consecutive frames and same segment types).

Maximum A Posteriori (MAP) inference, finds the most probable state given the evidence.

For inference, we used the exact solver toulbar2 branch & bound MPE inference [Allouche *et al.* 2010] with the ProbCog toolbox [Jain 2011].

Results

- Test-set: 143 hand-labeled Beatles songs $\rightarrow$ Removing songs for which the structure was ambiguous.
- Evaluation measure: chord label accuracy.

	Chord LA results	*Stat. Sig.*
MLN_chord	72.57 ± 13.51	}*yes*
MLN_struct	74.03 ± 13.90	
[36]	73.90 ± 13.79	}*no*

Figure: *MLN_struct: MLN incorporating prior structural information, MLN_chord: baseline HMM, [36]: chromagram averaged over same segment types as in [Mauch et al. 2009].*

Results

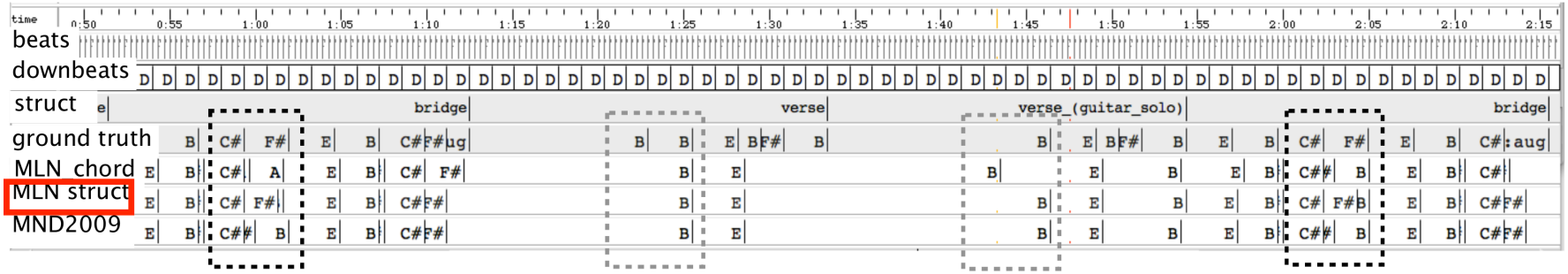


Figure: *Chord estimation results for an excerpt of the song One After 909*

Global semantic information can be concisely and elegantly combined with information at the beat-synchronous time scale:

→ Chord estimation results are significantly improved, and more consistent with the global structure (see the grey dashed rectangles).

→ Variations between segments are taken into account (see the black dashed rectangles).

Results of Key and Chord Estimation using Markov Logic Networks

University of Victoria Computer Science

Improving chord estimation using provided key information.
Joint estimation provides key estimation for free.

	Chord LA	*Stat. Sig.*
HMM	72.49 ± 14.68	**no**
Chord MLN	72.33 ± 14.78	
Prior key MLN, WMCR	$\mathbf{73.00 \pm 13.91}$	**yes**
Prior key MLN, CB	72.22 ± 14.48	no
Joint chord/key MLN	72.42 ± 14.46	no

	EE	*EE*	*E+N*	*Stat. Sig.*
Joint chord/key MLN	82.27	88.09	94.32	
DTBM-chord	48.59	67.39	89.44	yes
DTBM-chroma	75.35	85.14	95.77	yes

Music Content Estimation using Markov Logic

University of Victoria Computer Science

- Markov logic: a formalism that enables intuitive, effective, flexible and expressive reasoning about complex relational structure and uncertainty of music data.
- New step towards a unified multi-level description of audio at various time-scales, in which information specific to various semantic levels (analysis frame, phrase and global structure) interact.
- Future work: extend this approach to a fully automatic one (incorporating estimated beats and structure location).

Future work

University of Victoria Computer Science

Major objectives:

- Automatically discover and model new structural rules with learning algorithms.
- Take advantage of the flexibility of the MLN to combine training iformation with background music knowledge.

Open questions - future work

University of Victoria Computer Science

- Think in parallel about a "good signal representation" and the "models for information extraction".
- Fully integrate in the model chord and beat estimation in an interactive fashion (e.g. update the chromagram with the other parameters during MCMC inference in order to possibly improve the chord estimation).
- Incorporate more music knowledge / mid-level representations of audio (structure, instrumentation etc.) to inform signal representation.
- Build models that take into account complex relational structures and uncertainty of real audio.
- Towards a multi-scale signal analysis at multiple semantic levels in a complex acoustical environment.

More open questions

University of Victoria Computer Science

- Improve the priors (e.g. by using a time segmentation, a larger chord lexicon, using onsets combined with beat positions etc.).
- Extend the model so that dependencies *between* layers are taken into account
- Future work: extend this approach to a fully automatic one (incorporating estimated beats and structure location).
- Major objectives:
- Automatically discover and model new structural rules with learning algorithms.
- Take advantage of the flexibility of the MLN to combine training information with background music knowledge.

Markov Logic Networks in Music

University of Victoria
Computer Science

Thank you for your attention!

Related publications

H. Papadopoulos and G. Tzanetakis. Models for Music Analysis from a Markov Logic Perspective, *IEEE/ACM Transactions on Audio, Speech and Language Processing*, 25(1), 19-34, 2016.

H. Papadopoulos and G. Tzanetakis. Exploiting Structural Relationships in Audio Music Signals Using Markov Logic Networks, *Proceedings of ICASSP*, 2013.

H. Papadopoulos and G. Tzanetakis. Modeling Chord and Key Structure with Markov Logic, *Proceedings of ISMIR*, 2012.

Table of Contents I

Table of Contents II

Summary

Probabilistic modeling allows us to formulate and solve many interesting problems in music analysis and signal processing in general. It allows us to express our assumptions and associated uncertainties and learn from data. By using inference we can solve a variety of interesting problesm based on the same underlying model. It is easy to get lost in the details of individual algorithms and models but once you start understanding the basic concepts it becomes easier to make sense of the literature. I hope that this tutorial helped you formulate this big picture view of model-based probabilistic machine learning.