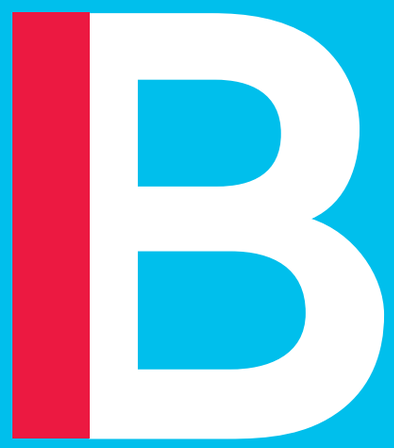
Imperial College Business School

Imperial means Intelligent Business

Machine Learning in Finance

Lecture 4

Markov Models & Hidden Markov Models



Arnaud de Servigny & Hachem Madmoun

Outline:

- Introduction : Position of to the problem
- Markov Models
- Hidden Markov Models
- Programming Session

Part 1 : Introducing the problem

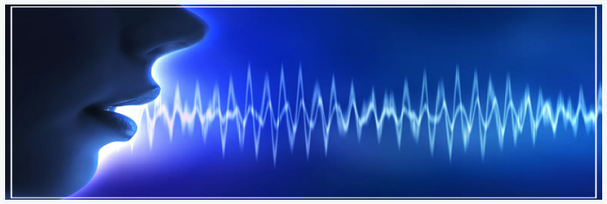
Markov models in finance:

- When the i.i.d. assumption is relaxed, sequentiality starts to matter! Forget the Brownian motion.
- In this family of models, the main assumption is that we can find a stable direct or indirect link between what happened yesterday and what happens today.
- The above assumption is quite simple. Maybe too simple, but such models are parsimonious on data. This is good when the training set is limited in size.
- Markov models represented a significant improvement over traditional ARMA / ARIMA models, as they can treat different pieces of information at once (e.g. return & volatility)
- This approach accounts particularly well for Regimes. In finance growth and recession regimes tend to be quite well separated.
- The technique was first developed by Hamilton in 1989, but a lot of innovation / refinements have been brought to the table since then. It is therefore not old science!

Markov models apply to many different Sequential Problems

- Sequences are everywhere:
  - Financial data

  - Speech data



  - Texts

- Ignoring sequentiality in the data results in sub-optimal models since a huge amount of information is contained in the order.

- For instance, let's consider this example:
  - « Never Quit. Do your best »
  - « Never do your best. Quit. »

- A **bag of words*** model won't make any difference between these two sentences.

Bag of words*: model that treats a sentence as a bag of its words disregarding word order (but keeping multiplicity).

What is our objective today ?

- We will not use Markov models on quantitative data although this would be possible, but rather on qualitative data. Applications are numerous:
    - Identifying the mood reflected in a sentence.
    - Relating a sentence or a text to a type of individuals.
    - Predicting the word or the sentence to come.
    - Identifying textual anomalies.

- Ultimately, the objective of this lecture is to introduce a graphical model, called **Hidden Markov Model** (HMM) in the context of **Language Modelling.**

- A **Language Model** is a model used to predict the probability of a sequence of words.

- We want our language model to assign a probability to every sentence according to its level of coherence (with respect to the training data). For instance:

    - « The quick brown fox jumps over the lazy dog » : **high probability**

    - « Despite the constant negative press covfefe » : **average probability**

    - « The the the the the the » : **low probability**

What do we need to calculate these probabilities ?

- First, we need a **model**, which consists in making some assumptions on the dependencies between words.

- We also need a **training corpus:** a large number of sentences (called **documents**).

- For instance, after training a **model** on a corpus of **documents** from John Stuart Mill, the model should assign high probabilities to the sentences that correspond to JSM way of thinking and much lower probabilities to the ones that contradict his ideas:

  - p(« the greatest good for the greatest number ») >> p(« An action, to have moral worth, must be done from duty »)

Common day-to-day applications:

- Language Models can be used for several applications, such as the following:

  - **Sentence Completion**: A language model can predict the next word given the previous words in a sentence.

  - **Classification**: Using Bayes rule, we can turn a language model into a ***generative classifier***. For instance, it can be used for sentiment analysis, which consists in predicting the sentiment conveyed by a sentence.

  - **Automatic essay writing**: A language model learns the probability distribution of sentences according to a specific corpus. We can then sample from this distribution to generate artificial sentences.

What is a Generative Classifier in the context of a Sequential Approach? (I)

- Let's consider a training dataset of N samples (*typically sentences*) used for a classification task.

- Let's denote the samples: $(X_i)_{1 \leq i \leq N}$, and the targets $(Y_i)_{1 \leq i \leq N}$ (*typically mood levels*)
  (Suppose $\forall i \quad Y_i \in \{0, 1\}$)
- We assume that each sample is made of sequential constituents (*typically words*)
  $$\forall 1 \leq i \leq N \quad X_i = X_i^1, \ldots, X_i^T$$
- Supervised Classification consists in estimating $p_\theta(Y_i|X_i)$ where θ represents the parameters of the supervised learning algorithm we are looking to estimate.

- A generative model like HMM aims to learn from the distribution of the training set of sequences:
  $$p_\theta(X_i) = p_\theta(X_i^1, \ldots, X_i^T)$$
- We can train 2 different models. The first model, parametrized by θ_0 is associated to the samples of target 0 and the second model parameterized by θ_1 is associated to the samples of target 1.

- As a consequence, we will learn the two following distributions:

$$p_{\theta_0}(X_i^1, \ldots, X_i^T|Y_i = 0) \qquad \text{and} \qquad p_{\theta_1}(X_i^1, \ldots, X_i^T|Y_i = 1)$$

So far, we have built a ***class conditional density estimator***

What is a Generative Classifier in the context of a Sequential Approach? (II)

- Using the Bayes rule described as : $\forall A, B \in \Omega \quad \mathbb{P}(B|A) = \frac{\mathbb{P}(A,B)}{\mathbb{P}(A)} = \frac{\mathbb{P}(A|B)p(B)}{\mathbb{P}(A)}$

we can turn the previous class conditional density estimator into a ***classifier***.

$$\forall 1 \leq i \leq N :$$

$$p(Y_i = 0|X_i^1, \ldots, X_i^T) = \frac{p(X_i^1, \ldots, X_i^T, Y_i = 0)}{p(X_i^1, \ldots, X_i^T)}$$

$$= \frac{p_{\theta_0}(X_i^1, \ldots, X_i^T|Y_i = 0)p(Y_i = 0)}{p_{\theta_0}(X_i^1, \ldots, X_i^T|Y_i = 0)p(Y_i = 0) + p_{\theta_1}(X_i^1, \ldots, X_i^T|Y_i = 1)p(Y_i = 1)}$$

- Similarly:

$$\forall 1 \leq i \leq N :$$

$$p(Y_i = 1|X_i^1, \ldots, X_i^T) = \frac{p_{\theta_1}(X_i^1, \ldots, X_i^T|Y_i = 1)p(Y_i = 1)}{p_{\theta_0}(X_i^1, \ldots, X_i^T|Y_i = 0)p(Y_i = 0) + p_{\theta_1}(X_i^1, \ldots, X_i^T|Y_i = 1)p(Y_i = 1)}$$

- Where: $p(Y_i = 0)$ and $p(Y_i = 1)$ are estimated by their frequencies in the dataset.

Constructing a Generative Classifier. An example (I)

- Let's suppose we have a N sized training sample composed of documents from John Stuart Mill (**J**) and from Immanuel Kant (**K**).

Training Sample	**Target**
• Document 1 : « The sole evidence it is possible to produce that anything is desirable is that people actually to desire it. »	**J**
• Document 2 : « In law a man is guilty when he violates the rights of others. In ethics he is guilty if he only thinks of doing so. »	**K**
• Document 3 : « Always recognize that human individuals are ends, and do not use them as means to your end. »	**K**
…	
• Document N : « Justice is a name for certain moral requirements, which, regarded collectively, stand higher in the scale of social utility and are therefore of more paramount obligation than any others. »	**J**

- We want our model, to classify new sentences like « The greatest good for the greatest number ».

- Let's review the different steps involved in computing the following probabilities:
  - p(Y = K | « The greatest good for the greatest number »)
  - p(Y = J | « The greatest good for the greatest number »)

Constructing a Generative Classifier. An example (II)

STEP 1

- Document 1 : « The sole evidence it is possible to produce that anything is desirable is that people actually to desire it. »

…

- Document N : « Justice is a name for certain moral requirements, which, regarded collectively, stand higher in the scale of social utility and are therefore of more paramount obligation than any others. »

J

- Document 2 : « In law a man is guilty when he violates the rights of others. In ethics he is guilty if he only thinks of doing so. »

…

- Document 3 : « Always recognize that human individuals are ends, and do not use them as means to your end. »

K

STEP 2 Train the density estimator 1 on documents with target J

Train density estimator 2 on documents with target K

STEP 3 For each new sentence, calculate the probability of the target being J or K, conditional on this provided sequence, using the Bayes Rule and the two density estimators trained in Step 2.

STEP 4 Make a decision based on a decision rule: New sentence has been written by J, as

$$p(Y = J \mid \text{« The greatest good for the greatest number)} > p(Y=K \mid \text{« The greatest good for the greatest number)}$$

Part 2 : Markov Models

A Markov model relates to a process
=> which is capable of being in more than one state,
=> which can make transitions among those states, and
=> in which the available states and transition probabilities only depend upon what state the system is currently in.

Introduction

- In this section, we will discuss a simple Model called **Markov Model**.

- Notations:

  - We are dealing with sequences $(X_i)_{1 \leq i \leq N}$ of discrete ***observations*** (*e.g. sentences*).

  - Each observation X_i is a sequence of T discrete ***states*** **:** $X_i^1, \ldots, X_i^T$ (*e.g. words*).

  - Example :

Observation : « The greatest good for the greatest number »

X_1 : X_1^1 X_1^2 X_1^3 X_1^4 X_1^5 X_1^6 X_1^7

  - Each X_1^i is our example represents a word in the English dictionary.

Defining the ***Probability*** of a sequential Observation:

- Using simple probability rules, for any observation *i,* we can write the ***probability*** $p(X_i)$ as the product of conditional probabilities :

$$p(X_i^1, \ldots, X_i^T) = p(X_i^T | X_i^j, j < T) p(X_i^{T-1} | X_j, j < T-1) \ldots p(X_i^2 | X_i^1) p(X_i^1)$$

- The basic idea behind the ***Markov Property*** is to assume that X_i^t captures all the relevant information to account for the next step. The previous equation therefore simplifies to:

$$p(X_i^1, \ldots, X_i^T) = p(X_i^T | X_i^{T-1}) p(X_i^{T-1} | X_i^{T-2}) \ldots p(X_i^2 | X_i^1) p(X_i^1)$$

- **Additional assumptions**:
  - We assume that the transition function $p(X_i^t | X_i^{t-1})$ is independent of time t. I.e. will always hold.
  - We assume that the observed variables X_i^t are discrete : $X_i^t \in \{1, \ldots, V\}$

    In our previous example, each X_1^i represents a unique word among the different words of an English dictionary of size V *(the number of words listed in it)*.

To be clear, the Markov property means :

*p(The greatest good for the greatest number) = p(The) *p(greatest | The)* p(good | greatest) *p(for | good) *p(the | for) *p(greatest | the)* p(number | greatest)*

Summarizing the sequential rules learnt in a ***Transition Matrix***

- The conditional distribution $p(X^t|X^{t-1})$ inferred from all the observations can be written as a $V \times V$ matrix, known as the **transition matrix** Q **,** where

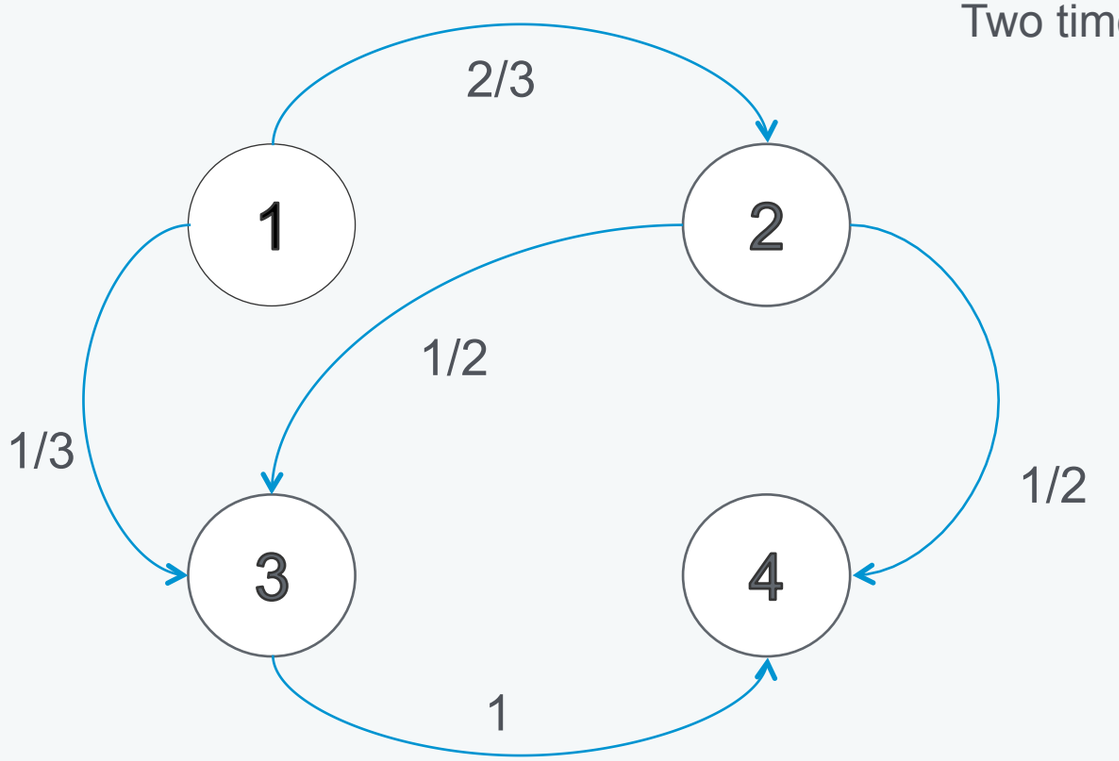
$$\forall v, w \in \{1, \dots V\} \quad Q_{vw} = p(X_i^t = w | X_i^{t-1} = v)$$

- Example : Let us consider the following 3 observations, containing 4 states in a dictionnary of size 4

  - The greatest good
  - The greatest number
  - The good number

V = 4

the	1	good	3
greatest	2	number	4



Two times out of three, ‘the’ is followed by ‘greatest’, hence $p(1,2)=2/3$

$$Q = \begin{pmatrix} 0 & 2/3 & 1/3 & 0 \\ 0 & 0 & 1/2 & 1/2 \\ 0 & 0 & 0 & 1 \\ 1/4 & 1/4 & 1/4 & 1/4 \end{pmatrix}$$

Note that the rows of the transition matrix must add up to one. This Transition matrix is also called a **stochastic matrix**.

Estimating the Transition Matrix from the Training Observations

Training Observations

$$\begin{aligned} X_1 &: X_1^1, \ldots, X_1^T \\ X_2 &: X_2^1, \ldots, X_2^T \\ &\vdots \\ X_N &: X_N^1, \ldots, X_N^T \end{aligned}$$

- The Data is of the form $X = [X_i^t]_{\{1 \leq i \leq N; 1 \leq t \leq T\}}$: we call it a ***tensor*** of shape (N, T).
- Each X_i^t represents a word in the English dictionary of dimension V. So, $X_i^t \in \{1, \ldots, V\}$
- Let's introduce the parameters π and Q , where :

$$\forall v \in \{1, \ldots, V\} \quad \pi_v = p(X_i^1 = v)$$

$$\forall v, w \in \{1, \ldots, V\} \quad Q_{vw} = p(X_i^{t+1} = w | X_i^t = v)$$

- Our objective is to find optimal π^* and $Q*$ (i.e the ones that maximize the likelihood of the data)

Maximum Likelihood Estimation for the Markov Model:

Interactive Session

Estimating the Transition Matrix from Training Data

- Let's recall the parameters we want to optimize:

$$\forall v \in \{1, \dots, V\} \quad \pi_v = p(X_i^1 = v)$$

$$\forall v, w \in \{1, \dots, V\} \quad Q_{vw} = p(X_i^{t+1} = w | X_i^t = v)$$

- We introduce for all $v, w \in \{1, \dots, V\}$ the following counts:

$C_v^1 = \sum_{i=1}^{N} \delta_{\{X_i^1 = v\}}$: The number of times the word v appears at the beginning of a sentence.

$C_{vw} = \sum_{i=1}^{N} \sum_{t=1}^{T-1} \delta_{\{X_i^t = v\}} \delta_{\{X_i^{t+1} = w\}}$ The number of times we transition from the word to w

- After a Maximum Likelihood estimation, we find the following optimal (and very intuitive) parameters:

$$\pi_v^* = \frac{C_v^1}{\sum_{v'=1}^{V} C_{v'}^1}$$

$$Q_{vw}^* = \frac{C_{vw}}{\sum_{w'=1}^{V} C_{vw'}}$$

Limits of a Model driven by a Markov Process

- The Markov Model has a lot of limitations:
  - *A Memory Structure limit*: First, the Markov Property corresponds to a very strong assumption: Making each word only depend on the previous one is not realistic.
  - *A Dimensionality limit*: If we have $V \approx 100.000$ words in our Dictionnary, then a Markov Model will have about 10^{10} parameters to train, corresponding to all the possible pairs.
  - *A Missing Value limit*: It is also very unlikely to find all these pairs in the training dataset.

How could we more realistically use the Markov Approach?

- The next section will introduce a more subtle approach using the Markov scheme. In an other way rather than having the states (*the words*) in an observation which depend on their previous neighbour, we introduce a latent variable (*the grammar*) which will driver each state and which in turn will follow the Markov property. This is the reason why the word "**Hidden**" start to be used, as the latent variable (*the grammar*) has to be discovered by the model.

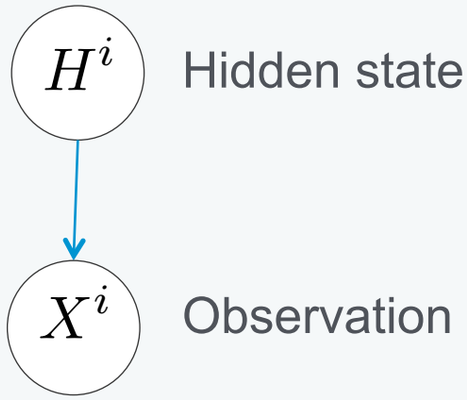
- A **Hidden Markov Model (HMM)** is a powerful discrete time / discrete state generative model which aims to solve the issues of the Markov Model via the concept of **Latent Variables.** A common metaphor is to think of the HMM as if the Markov Model were a mechanism hidden behind a curtain. The observer doesn't have any knowledge of how the mechanism is operating, and only knows what is going on behind the curtain from periodic reports that are emitted by the mechanism.

- In particular, it's worth noticing that the HMM model has the advantage of being able to represent long-range dependencies between observations. The Markov Property still holds, but not for the observations themselves.

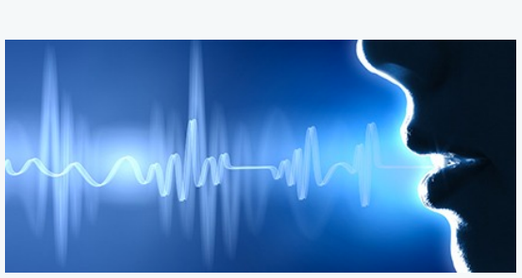
Part 3 : Hidden Markov Models

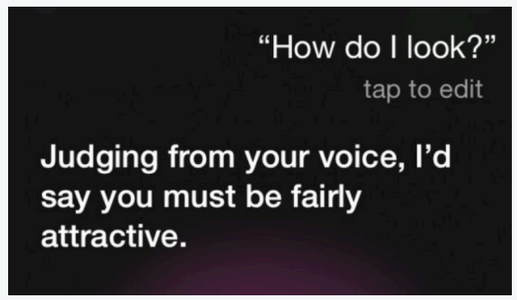
Introduction:

- The basic idea behind the HMM is the concept of a **hidden state** responsible for the observation.

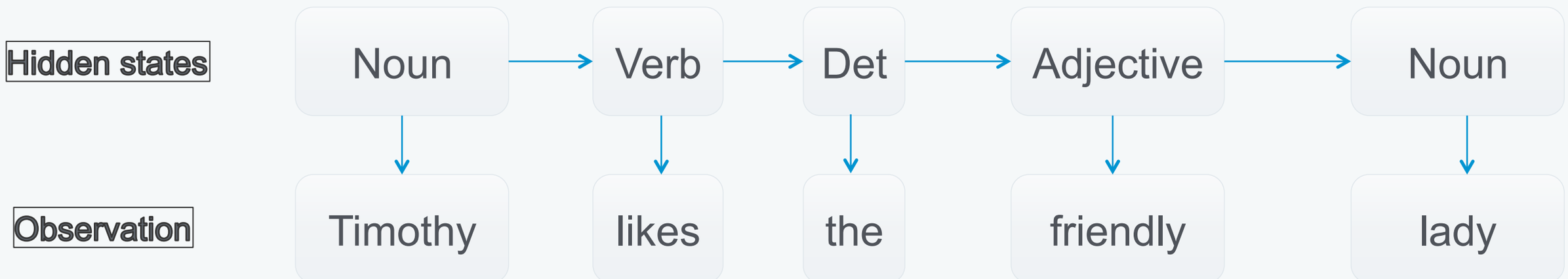


- Examples of HMMs where the hidden states have specific meanings:
  - **Automatic Speech Recognition**: The observation is the speech signal and the hidden state is the text.



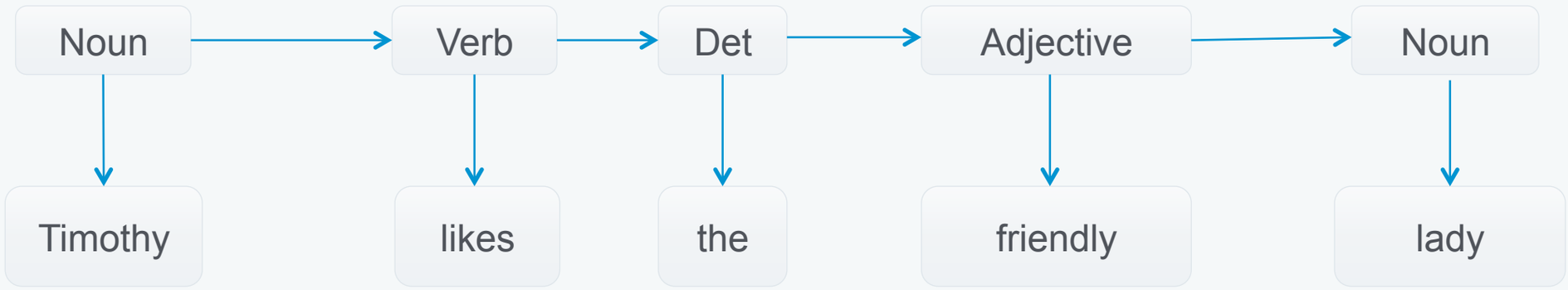


  - **POS (Part Of Speech) Tagging** : The observation is the sentence, the hidden state is the syntax.



Parameterization: The Hidden States form a Markov Chain

- Let us consider the example of POS tagging:



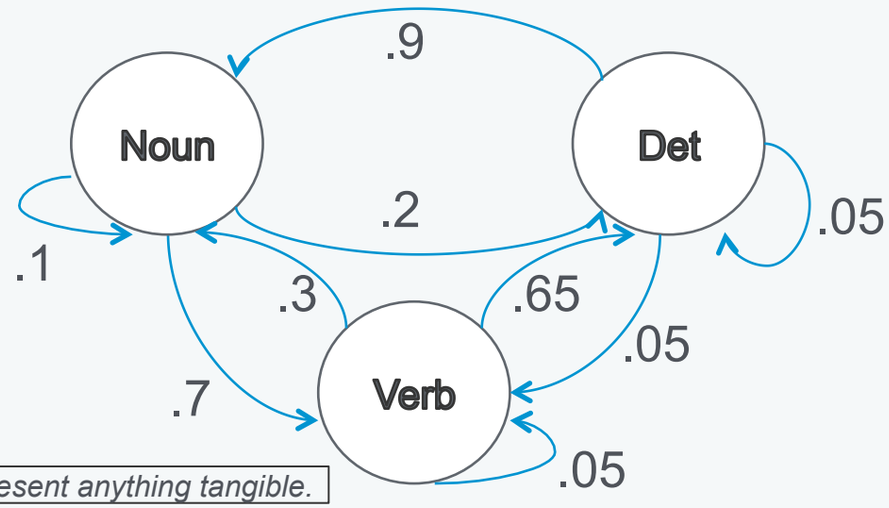
- We have M discrete syntax **tags** {Noun (N), Verb (V), Determinant (D), …} and V discrete words in the English Dictionnary.

- The dynamics of the tags will be parameterized by a Markov Model : $\pi \in \mathbb{R}^M$ and $Q \in \mathbb{R}^{M \times M}$

- The Markov Property makes much more sense in terms of grammar rules. For instance, we expect p(Noun| Determinant) to be high and p(Det|Det) to be very low.

- Illustration:

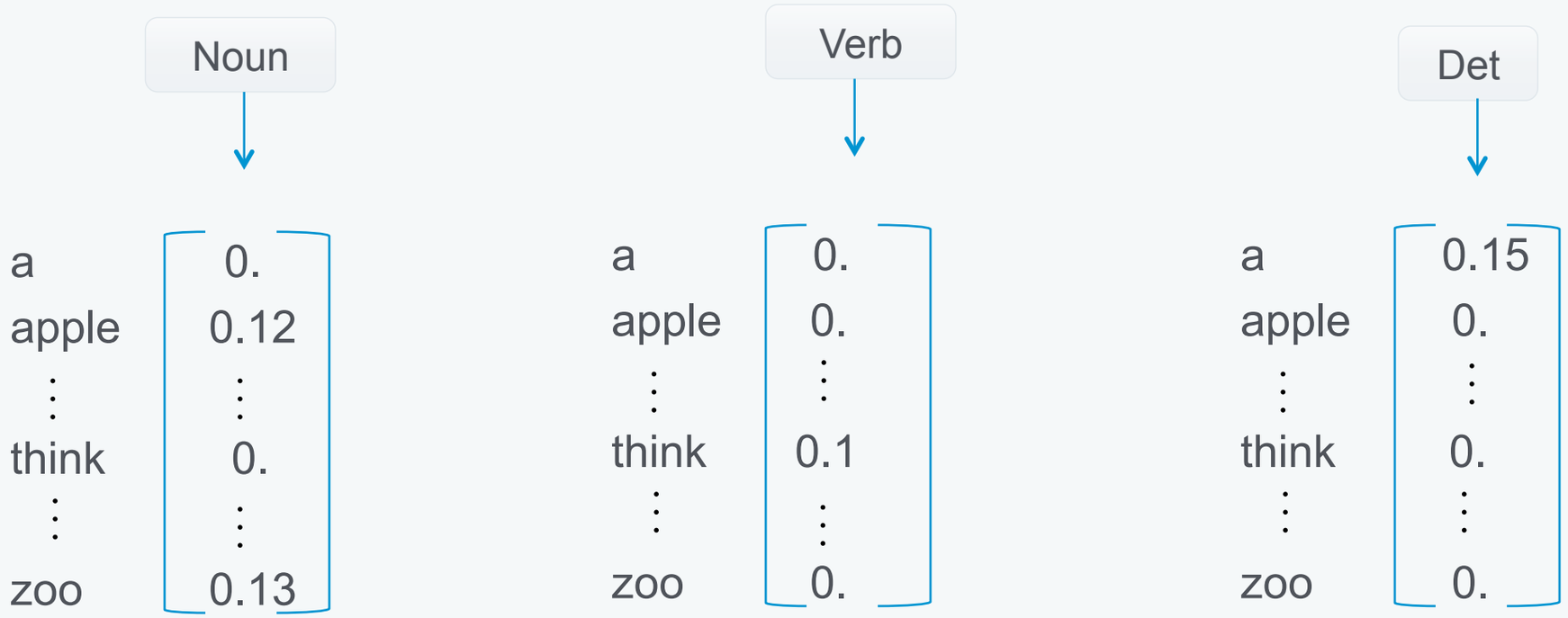
$$Q = \begin{matrix} \\ N \\ V \\ D \end{matrix} \begin{matrix} \begin{matrix} N & V & D \end{matrix} \\ \begin{pmatrix} .1 & .7 & .2 \\ .3 & .05 & .65 \\ .9 & .05 & .05 \end{pmatrix} \end{matrix}$$



Although we often imbue the hidden states with some specific meaning, they don't necessarily have to represent anything tangible.

Parameterization : Obtaining the Observation Matrix

- Each Hidden state is responsible for a discrete probability distribution over all the possible words in the English Vocabulary:



- We group all these probability distributions into a matrix called the **Observation Matrix.**

$$O = \begin{pmatrix} 0. & \dots & 0. & \dots & 0.15 \\ 0.12 & \dots & 0. & \dots & 0. \\ \vdots & \dots & \vdots & \dots & \vdots \\ 0. & \dots & 0.1 & \dots & 0. \\ \vdots & \dots & \vdots & \dots & \vdots \\ 0.13 & \dots & 0. & \dots & 0. \end{pmatrix} \in \mathbb{R}^{V \times M}$$

What needs to be estimated to have a HMM work :

- To sum up, when we are dealing with discrete observations, the Hidden Markov Model is parametrized by the three following elements:

$$\pi \in \mathbb{R}^M \qquad Q \in \mathbb{R}^{M \times M} \qquad O \in \mathbb{R}^{V \times M}$$

- Compared to the Markov Model, we have reduced the dimensionality from $V + V^2$ to $M + M^2 + M \times V$

$$\pi = \begin{pmatrix} 0.3 \\ \vdots \\ 0.1 \\ \vdots \\ 0.5 \\ \vdots \end{pmatrix} \begin{matrix} \text{Noun} \\ \\ \text{Verb} \\ \\ \text{Det} \\ \end{matrix}$$

$$Q = \begin{array}{c} \begin{matrix} \text{Noun} & & \text{Verb} & & \text{Det} \end{matrix} \\ \begin{pmatrix} 0.2 & \dots & 0.6 & \dots & 0.15 \\ \vdots & \dots & \vdots & \dots & \vdots \\ 0.3 & \dots & 0.05 & \dots & 0.4 \\ \vdots & \dots & \vdots & \dots & \vdots \\ 0.8 & \dots & 0.02 & \dots & 0.02 \end{pmatrix} \end{array} \begin{matrix} \text{Noun} \\ \\ \text{Verb} \\ \\ \text{Det} \end{matrix}$$

$$O = \begin{array}{c} \begin{matrix} \text{Noun} & & \text{Verb} & & \text{Det} \end{matrix} \\ \begin{pmatrix} 0. & \dots & 0. & \dots & 0.15 \\ 0.12 & \dots & 0. & \dots & 0. \\ \vdots & \dots & \vdots & \dots & \vdots \\ 0. & \dots & 0.1 & \dots & 0. \\ \vdots & \dots & \vdots & \dots & \vdots \\ 0.13 & \dots & 0. & \dots & 0. \end{pmatrix} \end{array} \begin{matrix} \text{a} \\ \text{apple} \\ \\ \text{think} \\ \\ \text{zoo} \end{matrix}$$

Maximum Likelihood Estimation for the HMM:

Interactive Session

The limit and interest of the HMM approach

- By looking at a more abstract and synthetic element, the hidden variable, we acknowledge two things:
  - Most of the time, it will be difficult to put a name and identify the meaning behind this variable.
  - We have to lose part of the information in order to reduce dimensionality and handle a tractable model.

- In the previous model related to a sentence, we have for instance renounced to identify the precise word to come. The only things we will be able to say is that its most likely syntax is identifiable and that among such a syntax the most likely word is also identifiable. But we go for two levels of guesses.

- The HMM is therefore certainly not the best class of model to obtain a precise prediction. It carries the very useful advantage to invent classes, whose definition is not obvious, but as long as we find these classes informative generically, we have learnt something. In addition, provided we have enough data, we can learn from the HMM what is the optimal number of classes to consider. As a result it may prove a very effective clustering tool.

- In total the HMM should be considered as a data-parsimonious semi-supervised model from which we can identify a priori unknown clusters.

Go to the following link and take Quiz 5 :

https://mlfbg.github.io/MachineLearningInFinance/

Programming Session