Modélisation de l'incertitude — Master 2 MIAGE IA^2
Travaux dirigés N° 4 : Modèles de Markov cachés

Andrea G. B. Tettamanzi
Université côte d'Azur
andrea.tettamanzi@univ-cotedazur.fr

Année universitaire 2020/2021

Résumé

On va construire et utiliser un modèle de Markov caché discret pour l'analyse de séquences en se basant sur une bibliothèque Python, `Pomegranate` en l'occurrence.

1 Introduction

Le but de ce TP va être de vous familiariser avec les modèles de Markov cachés. Pour ce faire, nous nous baserons sur l'utilisation de Pomegranate (`https://pomegranate.readthedocs.io/en/docs/index.html`), un module Python, déjà utilisé pour le TD n° 3, qui réalise une grande variété de modèles probabilistes, parmi lesquels les modèles de Markov cachés.

Nous allons construire un modèle de Markov caché (dorenavant MMC) de séries historiques financières. Pour cela, vous allez récupérer les séries des rendements journaliers discrétisés en $\{-1, 0, +1\}$ obtenues lors du TD n° 1.

On se propose ici de reconnaître la tendance du marché à partir de l'observation des variations (discrétisées !) quotidiennes. Le modèle qu'on construira ne pourrait être plus simple : on fera l'hypothèse qu'il n'existent que deux tendances : à la hausse (*bullish*, du taureau, dans le jargon des traders) et à la baisse (*bearish*, de l'ours). Ces deux tendances sont les deux états "cachés" (non observables directement) de notre MMC ; les variations quotidiennes sont les observations "émises" par chaque tendance, suivant sa propre distribution de probabilité.

2 Consignes

1. Construisez avec Pomegranate le MMC décrit ci-dessus. Pour commencer, vous mettrez des probabilités arbitraires. Attention : pour quelque raison mistérieuse, Pomegranate ajoute toujours un état initial `start` et un état final `end` même si vous dites expressément au constructeur de la classe `HiddenMarkovModel` que vous ne les voulez pas. Il suffit alors d'ajouter une transition de probabilité 1/2 de `start` à *bullish* ainsi que de `start` à *bearish*.
2. Créez, avec un petit script Python, une série historique de test, où vous allez simuler un marché qui alterne, sur un total di 1000 jours, 50 jours de taureau et 50 jours d'ours ; lorsque le marché est *bullish*,

$$P(s_t = -1) = 0.2, \quad P(s_t = 0) = 0.3, \quad P(s_t = +1) = 0.5,$$

et lorsqu'il est *bearish*,

$$P(s_t = -1) = 0.5, \quad P(s_t = 0) = 0.2, \quad P(s_t = +1) = 0.3.$$

3. Entraînez le MMC (méthode `fit`) avec cette série de test. Une fois entraîné, le MMC devrait reconnaître (méthode `viterbi`) les deux tendances, au fur et à mesure qu'elles s'alternent. Vérifiez aussi que les probabilités d'émission des deux états reflètent bien les paramètres que vous avez utilisé pour produire la série de test.
4. Maintenant, appliquez le MMC aux séries réelles (celles que vous avez obtenues lors du TD n° 1). Cela veut dire entraîner le MMC avec une série, puis lui faire reconnaître les tendances sur la même série.
5. Évaluez les tendances obtenues sur la base des graphiques des prix des indices, que vous pouvez visionner sur Yahoo ! Finance. Y a-t-il correspondance systématique ? Y a-t-il un décalage ? La modélisation semble-t-elle marcher (ou ne pas marcher) de la même façon pour toutes les séries testées ?

Rendez votre code, vos réponses aux questions et vos observations dans un archive zippé par courriel.