Réseaux bayésiens avec OpenBayes

1 Introduction

Le but de ce TP va être de vous familiariser avec les réseaux bayésiens. Pour ce faire, nous nous baserons sur l'utilisation d'OpenBayes (`http://www.openbayes.org/`).

OpenBayes est un module python permettant de définir des réseaux bayésiens et incluant des algorithmes de propagation de croyances. Malgré le fait qu'il soit actuellement en version 0.1 et que son développement soit pour l'instant suspendu, il est assez facile d'accès et ses fonctionnalités nous permettront de mettre en pratique une partie de la théorie vue au cours.

2 Installation

1. Installez Numarray

   Windows : Récupérez l'installeur `numarray-1.5.2.win32-py2.5.exe` sur le serveur et lancez-le.

   Linux (basé Debian) : `sudo apt-get install python-numarray`.

   Autres plate-formes : récupérez `numarray-1.5.2.tar.gz` sur le serveur, décompressez-le et, dans le répertoire ainsi obtenu, tapez `python setup.py install`.

2. Installez OpenBayes

   Windows : Récupérez l'installeur `OpenBayes-0.1.0.win32.exe` sur le serveur et lancez-le.

   Autres plate-formes : récupérez `OpenBayes-0.1.0.tar.gz` sur le serveur, décompressez-le et, dans le répertoire ainsi obtenu, tapez `python setup.py install`.

Pour voir si tout s'est bien passé, lancez un interpréteur python et tapez :

```
>>> import OpenBayes
```

Si aucune erreur ne survient, vous pouvez passer à la suite.

3 Construire un réseau

Pour prendre en main OpenBayes, nous allons étudier la situation suivante :

> Un arbre peut perdre ses feuilles pour deux raisons : soit il est malade, soit le sol est trop sec. Il y a *a priori* 10% de chances qu'il soit malade ; de même, la probabilité que le sol soit sec est de 10%.
>
> Les probabilités conditionnelles de perte de feuilles en fonction de la sécheresse et de la maladie sont les suivantes :

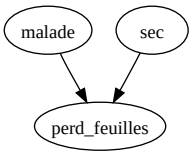

FIG. 1: Un réseau simple

	sec		¬sec	
	malade	¬malade	malade	¬malade
perd_feuilles	0.95	0.85	0.90	0.02
¬perd_feuilles	0.05	0.15	0.10	0.98

Nous allons donc construire dans OpenBayes le réseau de la figure 1.

Pour ce faire, nous allons commencer par importer du module OpenBayes les éléments dont nous aurons besoin :

```
from OpenBayes import BNet, BVertex, DirEdge, JoinTree
```

Ensuite, nous créons le réseau lui-même :

```
G = BNet( 'Botanical Problem' )
```

L'argument est simplement le nom du réseau et n'a au fait pas grande importance.

Créons maintenant les variables (noeuds, angl. *vertex*) de notre réseau ; pour créer un noeud, nous avons besoin de trois paramètres :

- une chaîne pour le nom de la variable
- un booléen indiquant si la variable est discrète (`True`) ou continue (`False`)
- le nombre d'états de la variable

Dans notre cas, cela donne donc :

```
s = G.add_v( BVertex( 's', True, 2 ) )
d = G.add_v( BVertex( 'd', True, 2 ) )
l = G.add_v( BVertex( 'l', True, 2 ) )
```

Il ne nous reste plus qu'à créer les flèches (angl. *edges*) ; pour chacune, nous devrons donner trois arguments :

- un ID unique (0, 1, 2, ...)
- le point de départ et le point d'arrivée

Ainsi, nous pourrions créer nos flèches avec le code :

```
G.add_e(DirEdge(0, s, l))
G.add_e(DirEdge(1, d, l))
```

Avec deux flèches, ça va, mais quand nous en aurons 15 cela fera beaucoup de copier-coller (ce qui est inacceptable dans un cours de programmation avancée...). Nous remplacerons donc le code ci-dessus par :

```
for e in [(s,l),(d,l)]:
    G.add_e( DirEdge( len( G.e ), *e ) )
```

Notre réseau est prêt. Ajoutez encore la ligne suivante :

```
print G
```

Sauvez puis exécutez votre code. Vous devriez voir une description du réseau s'afficher sur la sortie standard.

4 Tables de probabilités

Notre réseau n'est pour l'instant pas complet : sa structure est déterminée, mais il y manque les tables de probabilité !

Tout d'abord, il faut préparer le terrain :

```
G.InitDistributions()
```

Ensuite, on va entrer les probabilités *a priori* des noeuds s et d :

```
s.setDistributionParameters([0.9, 0.1])
d.setDistributionParameters([0.9, 0.1])
```

Enfin, les probabilités conditionnelles du noeud l, selon une syntaxe assez transparente :

```
l.distribution[{'s':0,'d':0}]=[.98, .02]
l.distribution[{'s':0,'d':1}]=[.15, .85]
l.distribution[{'s':1,'d':0}]=[.1, .9]
l.distribution[{'s':1,'d':1}]=[.05,.95]
```

Notre réseau est maintenant complet ; nous allons tenter de l'utiliser pour du raisonnement.

5 Inférence bayésienne

Pour propager des probabilités à travers le réseau, on commence par créer un "moteur" de calcul :

```
ie = JoinTree(G)
```

On peut maintenant directement calculer les probabilités qui nous intéressent :

```
print ie.Marginalise('l')
```

Exécutez votre programme ; vous devriez voir qu'en l'absence d'informations complémentaires, nos arbres ont presque 82% de chances de garder leur feuilles.

Maintenant, en regardant dans le jardin, vous constatez que les arbres perdent quand même leurs feuilles :

```
ie.SetObs({'l':1})
```

Si vous vous demandez quelles sont les nouvelles probabilités dans le réseau suite à cette information, vous pouvez tout calculer d'un coup :

```
results = ie.MarginaliseAll()
for n,r in results.items(): print r, "\n"
```

Il y a donc 47% de chances que votre sol soit sec et 49% pour que vos arbres soient malades.

Vous arrosez régulièrement vos plantation et pourtant elles perdent toujours leurs feuilles[1] ?

```
ie.SetObs({'l':1, 'd':0})
print  ie.Marginalise('s')
```

Vous avez donc 83% de chances que vos arbres soient malades...

[1] Notez que le nouveau `SetObs` *écrase* les informations précédentes. Donc si on veut ajouter une nouvelle observation, il faut répéter la/les précédente(s).

6 À vous de jouer !

6.1 Licencié ou augmenté ?

Voici une situation hautement réaliste : ;–)

> Dans l'entreprise Bayes&Co, il y a un patron despotique qui décide périodiquement de licencier des employés et d'augmenter le salaire d'autres employés. Sa politique n'est pas très populaire car il y a autant de licenciements que d'augmentations.
>
> Chaque fois que le patron prend une de ces décisions, elle doit être traitée par le département des ressources humaines et par la comptabilité, qui envoient ensemble une notification à l'employé (mal)heureux.
>
> Le département des ressources humaines compte trois employés : Janet, Jim et Julia. Lors d'un licenciement, Janet s'occupe du dossier dans 30% des cas, Jim dans 35% et Julia 35%. Si par contre il s'agit d'une augmentation de salaire, les proportions passent à 20%, 20% et 60% respectivement.
>
> À la comptabilité travaillent James et Jules. Lors d'un licenciement, James s'occupe de 37% des cas (et Jules de 63%), alors que lors d'une augmentation la proportion passe à 42% (et donc 58% pour Jules).
>
> Les employés concernés peuvent être avertis au choix par mail ou par courrier. Il suffit que les deux responsables du dossier se mettent d'accord. On constate donc que Janet et James écrivent une lettre dans 40% des cas (et donc un mail dans 60% des cas) ; mais lorsque Janet collabore avec Jules, ces proportions sont de 70% et 30% ; pour Jim et James 30% et 70% ; pour Jim et Jules 65% et 35% ; pour Julia et James 25% et 75% ; enfin pour Julia et Jules, 55% et 45%.

Sur la base de cette situation :

1. Dessinez (sur papier !) un réseau bayésien représentant cette situation. Au besoin, aidez-vous de la feuille "Construire un réseau bayésien : Aide mémoire" distribuée au cours.
2. En utilisant OpenBayes, écrivez un programme implémentant ce réseau.
3. Utilisez ce programme pour répondre aux questions suivantes :

   a) Jeremy a reçu ce matin une lettre de Janet et Jules ; quelle est la probabilité qu'il s'agisse d'une lettre de licenciement ? (réponse : 62%)

   b) Et s'il a reçu une lettre de Jim et James ? (réponse : 61%)

   c) Si tout ce que Jeremy sait, c'est que Julia est en charge d'un dossier le concernant, quels sont ses risques d'être licencié ? (réponse : 37%)

   d) Jeremy a reçu un mail ce matin, mais il ne sait pas qui a traité son dossier. Quelle probabilité a-t-il d'y lire une annonce d'augmentation de salaire ? (réponse : 52%)

6.2 Enquête

"Un cambriolage a été constaté au siège de l'entreprise BN Tech. Une enquête préliminaire a permis de montrer que le coupable a opéré seul et qu'il s'agit forcément de l'un des 100 employés que compte l'entreprise.

Quelques jours plus tard, la police a reçu une lettre anonyme accusant Pierre Kiroul, un nouvel employé de BN Tech. La police a immédiatement relevé ses empreintes digitales et les a comparé à celles retrouvées sur le lieu du forfait. Aucune trace correspondant à M. Kiroul n'a pu être mise en évidence.

Mais selon l'inspecteur de la police scientifique chargé de l'affaire, il ne faudrait pas en tirer des conclusions hâtives. En effet, le vol a pu être effectué avec des gants. Environ 45% des cambrioleurs recourent à cette technique (alors que la probabilité de mettre des gants sans faire de cambriolage peut être considérée comme nulle, dans ce contexte).

– Si le voleur n'a pas mis des gants, il y a 99% de chances que nous détections ses empreintes ; mais avec des gants, cette chance tombe à 2%. La probabilité d'observer les empreintes de quelqu'un de non-compable sur les lieux du crime est de 5%, estime l'expert de la police.

À ce point-là de l'histoire (1), Pierre semblait presque hors de cause. Mais la police avait également trouvé sur le lieux du forfait quelques cheveux du coupable, ce qui lui a permis de faire une analyse ADN. Cette analyse a provoqué un retournement de situation : l'ADN du coupable correspondait à celui de Pierre !

– Ce test est d'une grande fiabilité ; les taux de faux positifs et de faux négatifs sont tous deux à 0.1%. Autant dire que nous avons maintenant une quasi-certitude que M. Kiroul est le coupable, triomphe l'inspecteur. (2)

Pierre Kiroul a donc été accusé ; à son procès, son avocat a brillamment montré que Pierre était dans l'impossibilité de se fournir le type de gants nécessaire à la réalisation propre du crime dont on l'accusait. L'hypothèse que Pierre portait des gants doit donc être écartée. (3) M. Kiroul va-t-il être condamné ou acquitté ?"

Sur la base du palpitant thriller ci-dessus :

1. Représenter la situation sous forme de réseau bayésien (sur papier, avec les tables de probabilité).
2. En utilisant OpenBayes, écrivez un programme implémentant ce réseau.
3. Utiliser cette implémentation pour calculer la probabilité de culpabilité de Pierre Kiroul aux points (1), (2) et (3) de l'histoire ci-dessus. (Réponses : 0.5%, 82.6%, 5.5%).