Réseaux Bayésiens en Python avec pyAgrum

LTO@AGRUM.ORG

2ème Journée Aquitaine IA, RO et Data Science

7 Février 2019 - Bordeaux - Kedge Business School

aGrUM
Une API C++ GPL 3 depuis 2008

Auteurs	• Pierre-Henri Wuillemin • Christophe Gonzales
Mainteneurs	• Lionel Torti
Utilisateurs	• LIP6, INRA, IRSN • EDF, Airbus, IBM, Engie • Akheros, EdgeMind • Open Turns

Objectif: de la recherche à l'application

Recherche scientifique

Projets européens

MIDAS

SCISSOR

Projets ANR

SKOOB

INCALIN

LARDONS

DESCRIBE

SUNSTONE

Une API dédiée

Hyper paramétrable

Structure de données optimisées

Algorithmes optimisés

Diffuser et promouvoir

Wrapper Python

Documentations

Notebooks

Interventions

pyAgrum: un wrapper Python d'aGrUM

- aGrUM c'est quoi ?
  - Une API C++ d'environ 300 000 lignes de code.
  - Des templates, beaucoup de templates…
  - Du code portable: GCC, MVSC, Clang, MinGW…
  - Beaucoup de tests et l'envie de faire de la qualité.
- Pourquoi pyAgrum ?
  - Un accès simplifié aux algorithmes d'aGrUM.
  - Un accès à l'écosystème Python.
  - Une intégration avec les Jupyter Notebooks.
- Une combinaison gagnante!
  - L'implémentation C++ rend pyAgrum très performant.
  - L'accès en Python rend aGrUM *user friendly*.
  - Bientôt des wrappers Java et R !

Réseaux Bayésiens

EN THÉORIE

Réseaux Bayésiens: c’est quoi ?

Les nœuds :

- Des variables aléatoires.

Les arcs:

- Des corrélations.

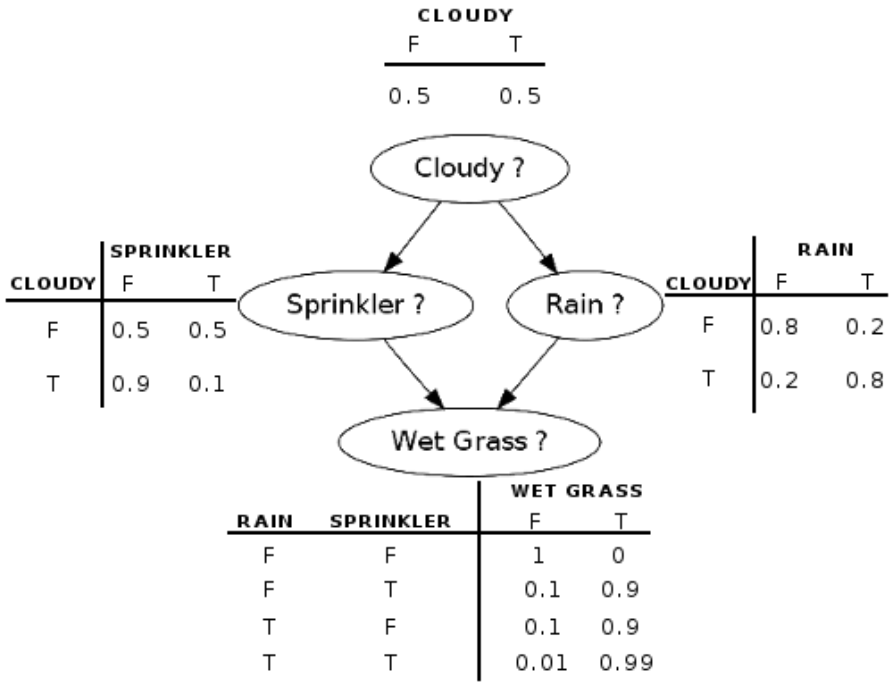
Les paramètres:

- Des Table de Probabilités Conditionnelles (TPC).

La loi jointe:

- Le produit des TPC.

La loi jointe est compressée sans perte d’information.



CLOUDY	F	T
	0.5	0.5

CLOUDY	SPRINKLER: F	SPRINKLER: T
F	0.5	0.5
T	0.9	0.1

CLOUDY	RAIN: F	RAIN: T
F	0.8	0.2
T	0.2	0.8

RAIN	SPRINKLER	WET GRASS: F	WET GRASS: T
F	F	1	0
F	T	0.1	0.9
T	F	0.1	0.9
T	T	0.01	0.99

$$P(C,S,R,W) = P(C).P(S|C).P(R|C).P(W|S,R)$$

Réseaux Bayésiens: comment ?

L'inférence probabiliste

- Calculer $\mathrm{P}(\mathrm{X}|\epsilon)$ où ϵ est une observation.
  - Quelle est la probabilité d'avoir un cancer si on est fumeur ?
  - Quelle est la probabilité d'être fumeur si on a un cancer ?
- Calculer $\mathrm{P}(\epsilon)$ où ϵ est un ensemble d'observations.
  - Quelle est la probabilité d'être fumeur et d'avoir un cancer ?
- Calculer l'explication la plus probable $arg\max_{x} P(x, \epsilon)$.
  - Si je suis fumeur, quelle est mon état de santé le plus probable ?

L'apprentissage automatique

- Apprendre la structure.
  - C'est-à-dire le graphe.
- Apprendre les paramètres.
  - C'est-à-dire les Tables de Probabilités Conditionnelles.

Réseaux Bayésiens: pourquoi ?

- Diagnostic automatique (HP, diagnostiques d'imprimantes).
- Gestion des risques (EDF, sûreté nucléaire).
- Marketing (score probabiliste).
- Analyse de modèle probabiliste (simulations).
- Classification (spam).
- Apprentissage de réseaux de gènes (bio-informatique)
- Apprentissage et modélisation d'utilisateur (agents conversationnels)

Réseaux Bayésiens

EN PRATIQUE

Titanic: Machine Learning from Disaster

Challenge Kaggle basé sur le naufrage du Titanic.

L'objectif est de prédire si un passager va survivre au naufrage.

Pour cela nous allons avoir trois approches:

- Une approche expert.
- Une approche de machine learning.
- Une approche combinant les deux approches précédentes.

Réseaux Bayésiens

L'APPROCHE EXPERT

L’approche expert

Aucun apprentissage automatique !

- Donnée inexistante ou inutilisable.
- Des experts du domaines disposés à nous aider.

La première étape consiste à définir les variables de notre réseau:

- Le genre: féminin ou masculin.
- L’âge: nourrisson, petit enfant, grand enfant, adolescent, adulte et personne âgée.
- Fratries: présence ou non de proches (enfants, frères et sœurs, mari et femme).
- Parents: présence ou non de parents (parents, grands-parents).
- Survivant: vrai si le passager a survécu, faux sinon.

L’approche expert

Entrée [2]:

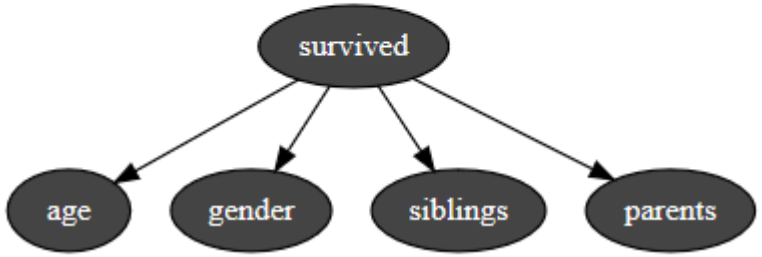
```
bn = gum.BayesNet("Surviving Titanic")

bn.add(gum.LabelizedVariable('survived','Did the passenger survived ?',
                             ["False","True"]))
bn.add(gum.LabelizedVariable('age', "The passenger's age category",
                            ["baby",
                             "toddler",
                             "kid",
                             "teen",
                             "adult",
                             "old"]))
bn.add(gum.LabelizedVariable('gender', "The passenger's gender",
                             ["Female","Male"]))
bn.add(gum.LabelizedVariable('siblings', "Did the passenger had siblings aboard ?",
                             ["False","True"]))
bn.add(gum.LabelizedVariable('parents', "The passenger had parents aboard ?",
                             ["False","True"]))

for i in bn.nodes():
    print(bn.variable(i))
bn
```

```
survived<False,True>
age<baby,toddler,kid,teen,adult,old>
gender<Female,Male>
siblings<False,True>
parents<False,True>
```

Out[2]:

L’approche expert

```
bn.addArc('survived', 'age')
bn.addArc('survived', 'gender')
bn.addArc('survived', 'siblings')
bn.addArc('survived', 'parents')
bn
```



L’approche expert

```
bn.cpt('survived').fillWith([100, 1]).normalizeAsCPT()
```

survived	
False	True
0.9901	0.0099

```
bn.cpt('age')[0:] = [ 1, 1, 1, 10, 10, 1]
bn.cpt('age')[1:] = [ 10, 10, 10, 1, 1, 10]
bn.cpt('age').normalizeAsCPT()
```

	age					
survived	baby	toddler	kid	teen	adult	old
False	0.0417	0.0417	0.0417	0.4167	0.4167	0.0417
True	0.2381	0.2381	0.2381	0.0238	0.0238	0.2381

```
bn.cpt('gender').fillWith([ 1, 1, 10, 1]).normalizeAsCPT()
```

	gender	
survived	Female	Male
False	0.5000	0.5000
True	0.9091	0.0909

```
bn.cpt('siblings').fillWith([ 1, 10,10, 1]).normalizeAsCPT()
```

	siblings	
survived	False	True
False	0.0909	0.9091
True	0.9091	0.0909

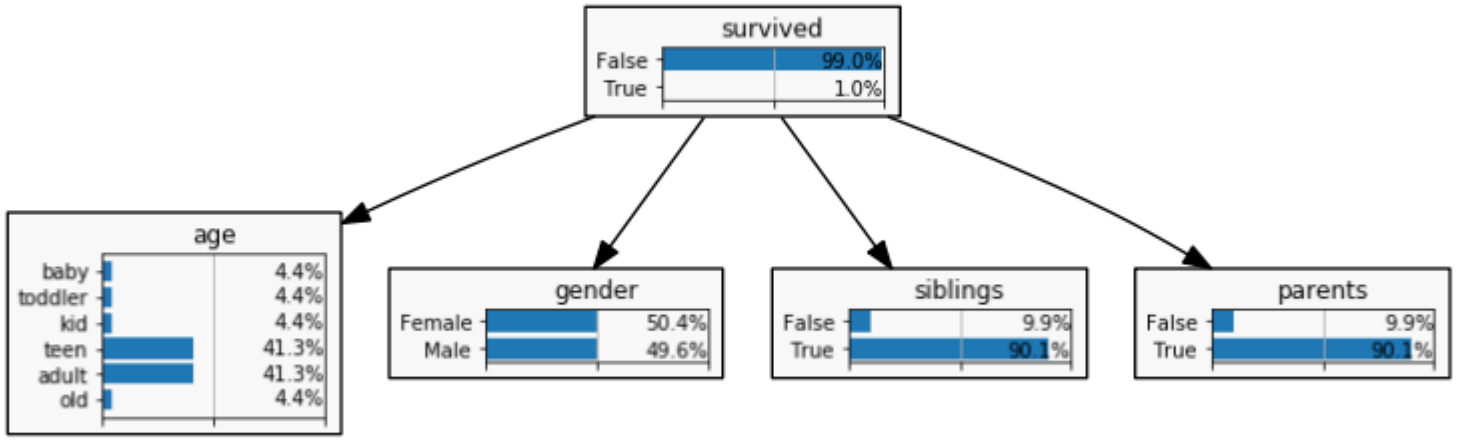
```
bn.cpt('parents').fillWith([ 1, 10,10, 1]).normalizeAsCPT()
```

	parents	
survived	False	True
False	0.0909	0.9091
True	0.9091	0.0909

```
gnb.showInference(bn,size="10")
```

L’approche expert

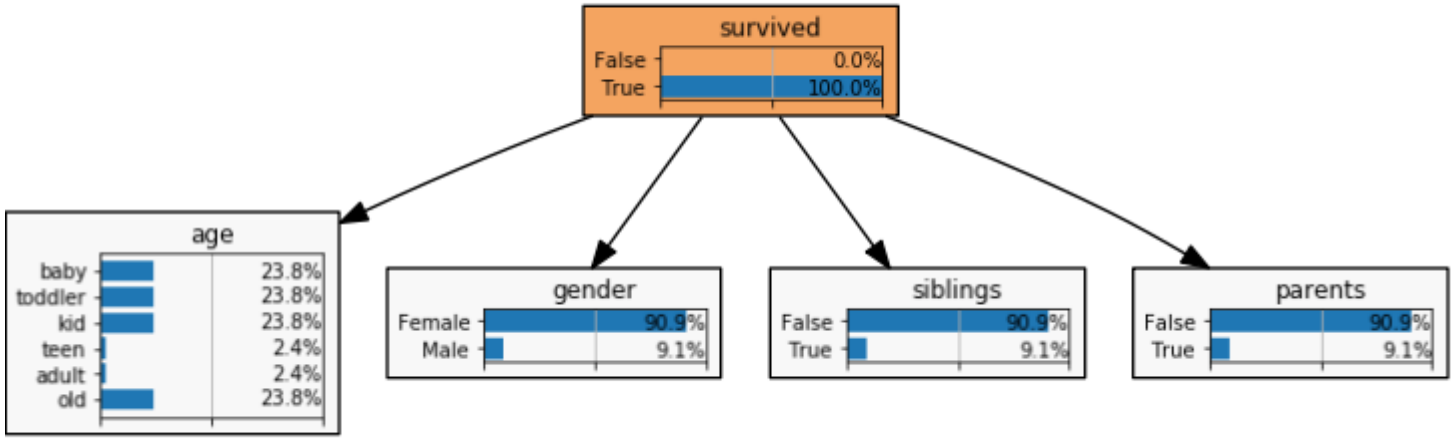
```
gnb.showInference(bn,size="10")
```



Inference in 1.03ms

L’approche expert

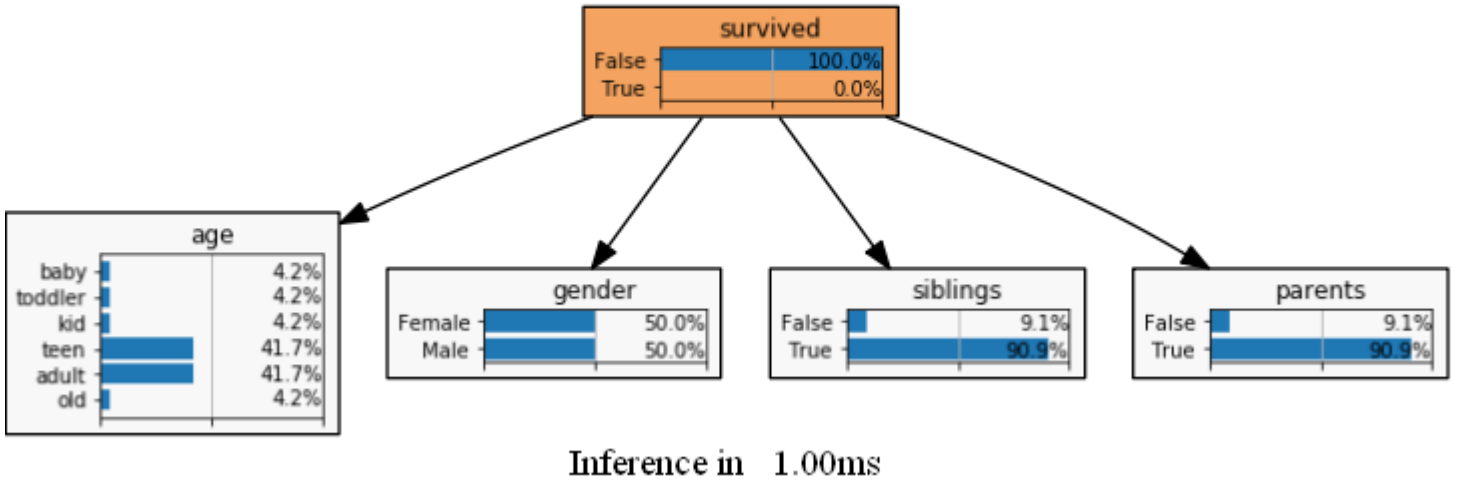
```
gnb.showInference(bn,size="10", evs={'survived':'True'})
```



Inference in 1.01ms

L’approche expert

```
gnb.showInference(bn,size="10", evs={'survived':'False'})
```



L’approche expert

55% de bonnes prédictions lorsque nous la confrontons à la base de test

Réseaux Bayésiens

L'APPROCHE MACHINE LEARNING

L’approche Machine Learning

99% d’apprentissage automatique

- Il vaut mieux définir les variables aléatoires.
  - Par contre nous allons presque toutes les utiliser !
- On ignore les lignes incomplètes.
  - Mais nous pourrions utiliser EM !

Les variables supplémentaires:

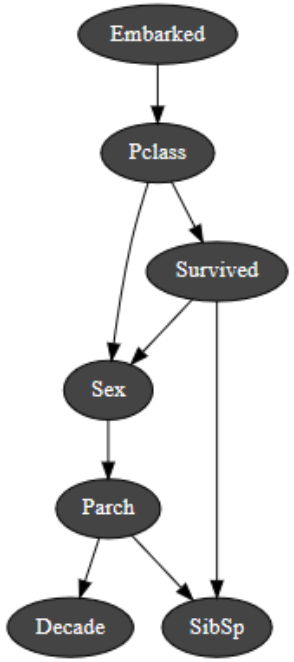
- Embarquement: le port d’embarquement.
- La classe : la classe de la cabine.

L’approche Machine Learning

```
template=gum.BayesNet()
template.add(gum.RangeVariable('Survived', 'Survived',0,1))
template.add(gum.RangeVariable('Pclass', 'Pclass',1,3))
template.add(gum.LabelizedVariable('Sex', 'Sex',, ['female', 'male']))
template.add(gum.RangeVariable('SibSp', 'SibSp',0,8))
template.add(gum.RangeVariable('Parch', 'Parch',0,9))
template.add(gum.LabelizedVariable('Embarked', 'Embarked', ['', 'C', 'Q', 'S']))
template.add(gum.LabelizedVariable('Decade', 'Calculated decade', ['baby', 'toddler', 'kid', 'teen', 'adult', 'old']))

gnb.showBN(template)
```

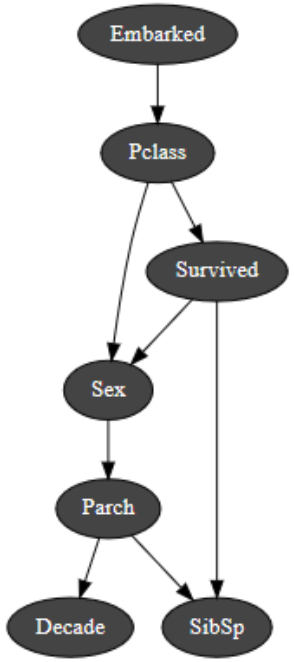
L’approche Machine Learning

```
file = os.path.join('res', 'titanic', 'post_train.csv')
learner = gum.BNLearner(file, template)
bn = learner.learnBN()
bn
```

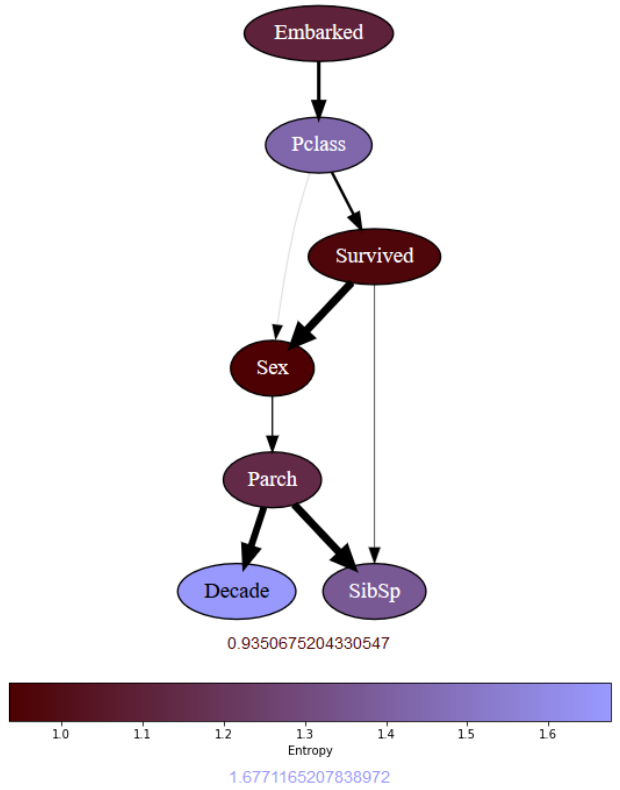


L’approche Machine Learning

```
file = os.path.join('res', 'titanic', 'post_train.csv')
learner = gum.BNLearner(file, template)
bn = learner.learnBN()
bn
```

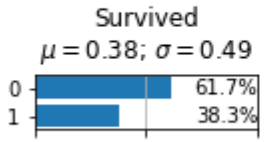



```
gnb.showInformation(bn,{},size="20")
```

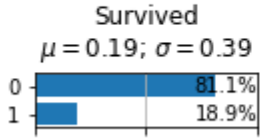


L’approche Machine Learning

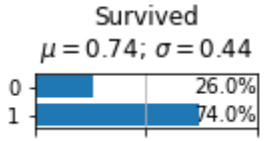
```
gnb.showPosterior(bn,evs={},target='Survived')
```



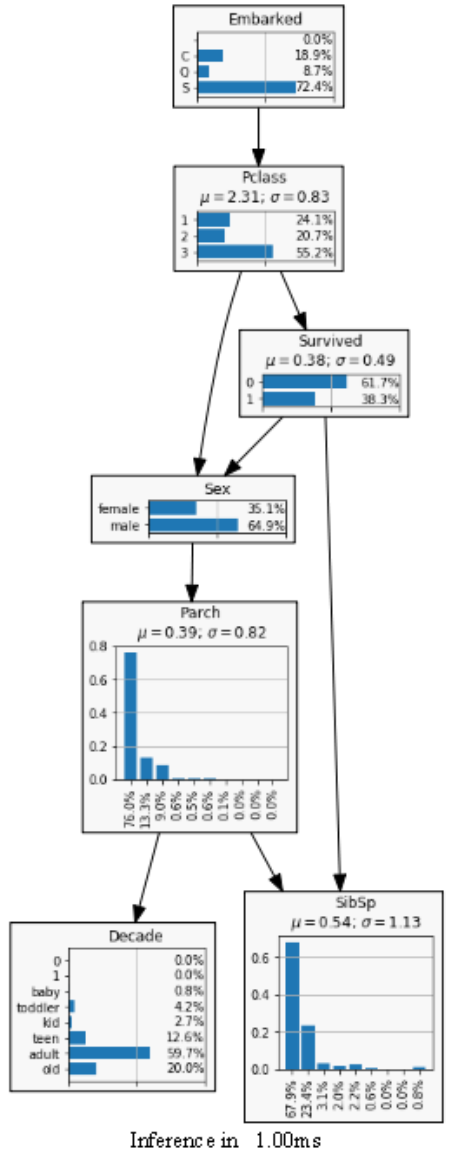

```
gnb.showPosterior(bn,evs={"Sex": "male", "Decade": 3},target='Survived')
```




```
gnb.showPosterior(bn,evs={"Sex": "female", "Decade": 'old'},target='Survived')
```




```
gnb.showInference(bn)
```



L’approche Machine Learning

85% de bonnes prédictions lorsque nous la confrontons à la base de test

Réseaux Bayésiens

LE MEILLEUR DES DEUX MONDES

L’approche combinée

Parfois nous avons à la fois:

- Des données exploitables.
- Des experts disponibles.

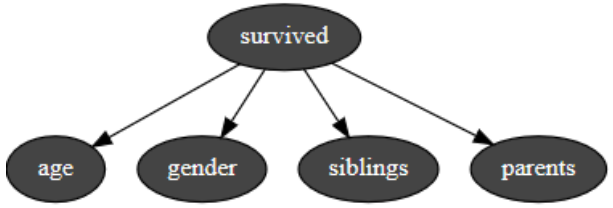
L’apprentissage automatique est très efficace pour calculer des probabilité.

L’expert est très efficace pour exprimer des relations de corrélation ou de causalité.

Et si nous utilisons le graphe de l’expert et l’apprentissage automatique sur les paramètres ?

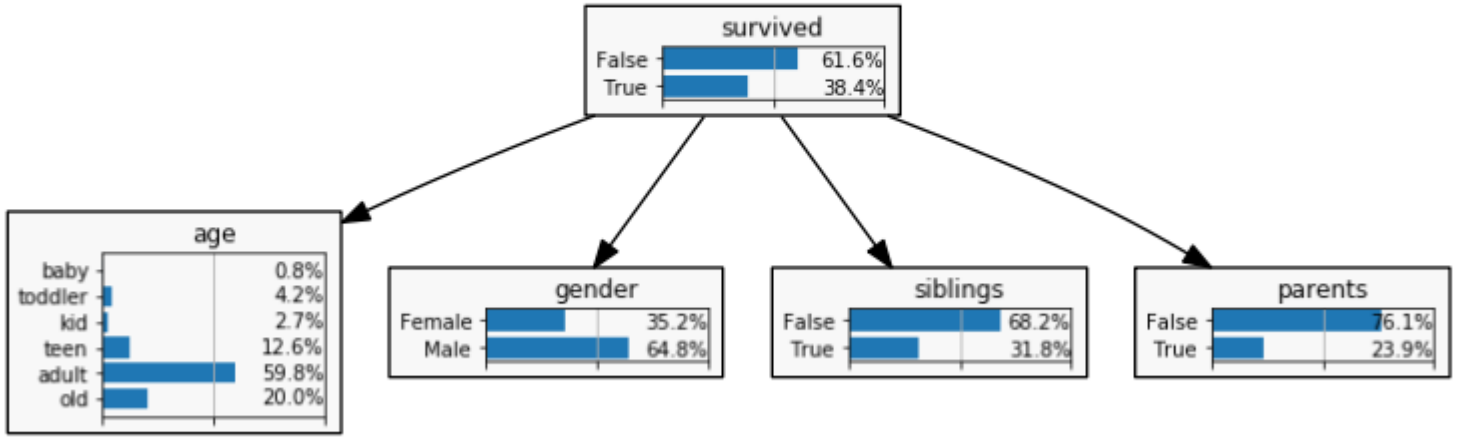
L’approche combinée

```
bn2 = gum.BayesNet("Surviving Titanic")
bn2.add(gum.LabelizedVariable('survived','Did the passenger survived ?',
                              ["False","True"]))
bn2.add(gum.LabelizedVariable('age', "The passenger's age category",
                             ["baby",
                              "toddler",
                              "kid",
                              "teen",
                              "adult",
                              "old"]))
bn2.add(gum.LabelizedVariable('gender', "The passenger's gender",
                              ["Female","Male"]))
bn2.add(gum.LabelizedVariable('siblings', "Did the passenger had siblings aboard ?",
                              ["False","True"]))
bn2.add(gum.LabelizedVariable('parents', "The passenger had parents aboard ?",
                              ["False","True"]))
bn2.addArc('survived', 'age')
bn2.addArc('survived', 'gender')
bn2.addArc('survived', 'siblings')
bn2.addArc('survived', 'parents')
bn2
```



L’approche combinée

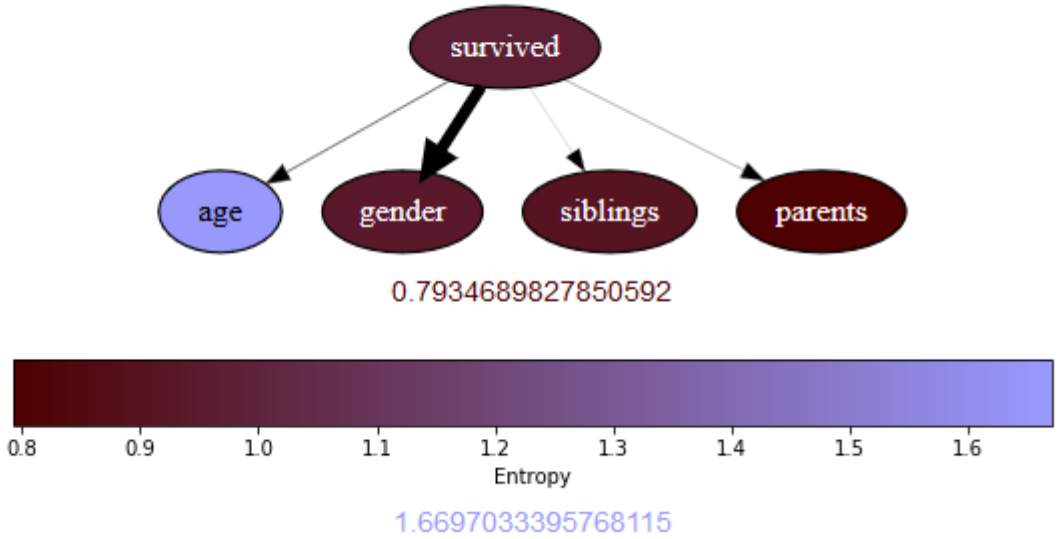
```
learner=gum.BNLearner(os.path.join("res", 'titanic', 'naive_bayes_post_train.csv'),bn2)
learner.setInitialDAG(bn2.dag())
bn2=learner.learnParameters()
gnb.showInference(bn2,size="10")
```



Inference in 1.00ms

L’approche combinée

```
gnb.showInformation(bn2)
```



L’approche combinée

95% de bonnes prédictions lorsque nous la confrontons à la base de test

Conclusion

Pour tout savoir sur aGrUM et pyAgrum : http://agrum.gitlab.io/

N’hésitez pas à nous contacter !

Sur GitLab : https://gitlab.com/agrumery/aGrUM

Sur le gitter: https://gitter.im/aGrUMers/Lobby

Sur la mailing list: https://mailia.lip6.fr/wws/info/agrum-users

Par email: info@agrum.org

MERCI !