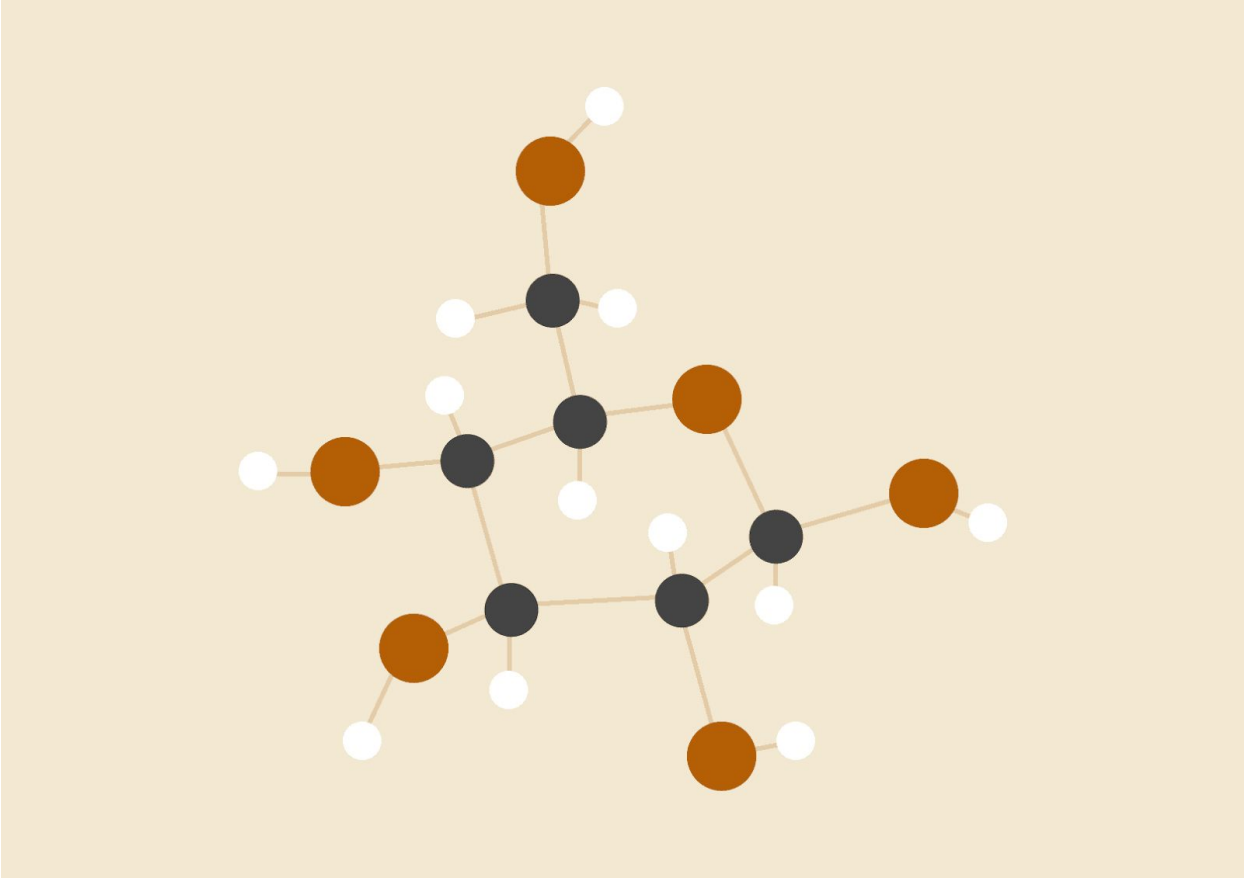
Data mining

TP 4



Michael Quinto, Jeremy Malera

625-2

Semestre d'automne 2020

INTRODUCTION

La banque pour laquelle nous travaillons, nous a demandé de créer un modèle qui nous servira pour prédire le produit d'investissement qu'un client est le plus susceptible de choisir selon son comportement. Ceci est utile pour ne proposer que le meilleur produit au client, au lieu de toutes les options possibles.

Dans le dernier TP, nous avons utilisé plusieurs outils qui nous ont permis de déterminer les variables prédictives les plus importantes pour prédire le résultat de la variable cible. Comme outils, on avait l'information gain, l'information gain ratio et le gini index et les variables les plus utiles que nous avons trouvé étaient : BA3, BA7, et BA5.

Dans ce TP, nous n'allons plus étudier les variables prédictives et leurs impacts sur la variable cible, mais plutôt considérer chaque variable comme étant indépendante et essayer de prédire le résultat. Pour cela, on va utiliser un algorithme populaire qui se nomme Naive Bayes. Cet algorithme est basé sur le théorème de Bayes, qui est fondé sur les probabilités conditionnelles, et la prédiction se fait en prenant en compte les valeurs des variables prédictives d'une instance et les probabilités conditionnelles de celles-ci.

Dans un premier temps, nous allons utiliser Naive Bayes pour créer un modèle et qui va nous servir pour essayer de déterminer le type d'investissement des instances de notre jeu de données.

Dans un deuxième temps, nous regarderons de plus près comment l'algorithme de classification Naive Bayes procède pour classer les instances et on l'expliquera avec un exemple.

PREMIERE PARTIE

Dans cette première partie, nous allons créer deux types de modèles grâce à l'algorithme Naive Bayes. Le premier sera créé à partir de la totalité des instances dans notre jeu de données, et le deuxième à partir d'un sous-ensemble d'instances. Plus précisément, pour le deuxième type, nous allons diviser le jeu de données en deux parties :

- **trainData** : contient 2/3 des instances.
- **testData** : contient 1/3 des instances (le reste), et qui sera utilisé pour tester le modèle. En effet, on va classifier les instances, c'est-à-dire prédire le résultat de leur variable cible, et comparer ces prédictions avec les valeurs réelles.

Création et explication du modèle final

Le modèle final créé est constitué de la totalité des instances et nous allons maintenant regarder de plus près à quoi ressemble ce modèle.

Premièrement, le modèle nous indique la distribution de probabilité de la variable cible, qui est dans notre cas invType et prenant les valeurs "C0" ou "C1":

```
A-priori probabilities:
Y
       C0        C1
0.4896493 0.5103507
```

Deuxièmement, on peut voir les distributions conditionnelles pour les variables prédictives discrètes :

```
    IA1
Y              0            1            2           26           3           5
  C0 0.9956859362 0.0025884383 0.0012942192 0.0000000000 0.0004314064 0.0000000000
  C1 0.9958609272 0.0028973510 0.0004139073 0.0004139073 0.0000000000 0.0004139073
```

Pour les variables prédictives continues, on obtient les moyennes [,1] et les écarts types [,2] conditionnelles :

```
    NRI
Y        [,1]     [,2]
  C0 48.47929 18.84941
  C1 40.57368 11.79634
```

L'algorithme Naive Bayes utilise ensuite ce modèle afin de classer les instances en fonction de leurs informations (nous verrons comment dans la deuxième partie).

Test de la performance du modèle

Pour tester la performance du modèle créé avec les instances du trainData, ce dernier prend celles du testData et prédit pour chacune d'elles le résultat de la variable cible. Finalement, il nous suffit de comparer les prédictions avec les résultats réels, afin de déterminer la performance du modèle créé. Pour avoir des résultats corrects, nous avons répété cette expérience 5 fois.

Voici les résultats :

Le nombre d'instances dans le testData à classer est de **1578**.

Expérience n°1 :

- Nombre d'instances correctement classées : **920**
- Pourcentage d'instances bien classées : **58.3016476552598 %**

Expérience n°2:

- Nombre d'instances correctement classées : **941**
- Pourcentage d'instances bien classées : **59.6324461343473 %**

Expérience n°3:

- Nombre d'instances correctement classées : **904**
- Pourcentage d'instances bien classées : **57.2877059569075 %**

Expérience n°4:

- Nombre d'instances correctement classées : **926**
- Pourcentage d'instances bien classées : **58.6818757921419 %**

Expérience n°5:

- Nombre d'instances correctement classées : **892**
- Pourcentage d'instances bien classées : **56.5272496831432 %**

Performance moyenne:

- Moyenne d'instances correctement classées : **916.6**
- Moyenne du pourcentage d'instances bien classées : **58.0861850443599 %**

Nous pouvons donc voir que l'efficacité moyenne de l'algorithme Naive Bayes pour ce jeu de données dépasse les 50%, mais il n'atteint tout de même pas un niveau très élevé.

Dans ce TP, nous allons également comparer l'algorithme avec une autre méthode qui se nomme le "default classifier".

Comparaison avec le default classifier

Le default classifier est un classificateur utilisant une méthode simple pour prédire la variable cible des instances. En effet, après avoir calculé la distribution de la variable cible pour les instances du trainData, il prend la classe majoritaire et l'assigne aux instances du testData. Finalement, toutes les instances auront la même valeur pour la variable cible.

Après avoir répété l'expérience avec les mêmes données, mais cette fois-ci avec le default classifier, on a obtenu les résultats suivants :

Expérience n°1 :

- Nombre d'instances correctement classées : **801**
- Pourcentage d'instances bien classées : **50.7604562737643 %**

Expérience n°2:

- Nombre d'instances correctement classées : **814**
- Pourcentage d'instances bien classées : **515842839036755 %**

Expérience n°3:

- Nombre d'instances correctement classées : **803**
- Pourcentage d'instances bien classées : **50.8871989860583 %**

Expérience n°4:

- Nombre d'instances correctement classées : **797**
- Pourcentage d'instances bien classées : **50.5069708491762 %**

Expérience n°5:

- Nombre d'instances correctement classées : **799**
- Pourcentage d'instances bien classées : **50.6337135614702 %**

Performance moyenne:

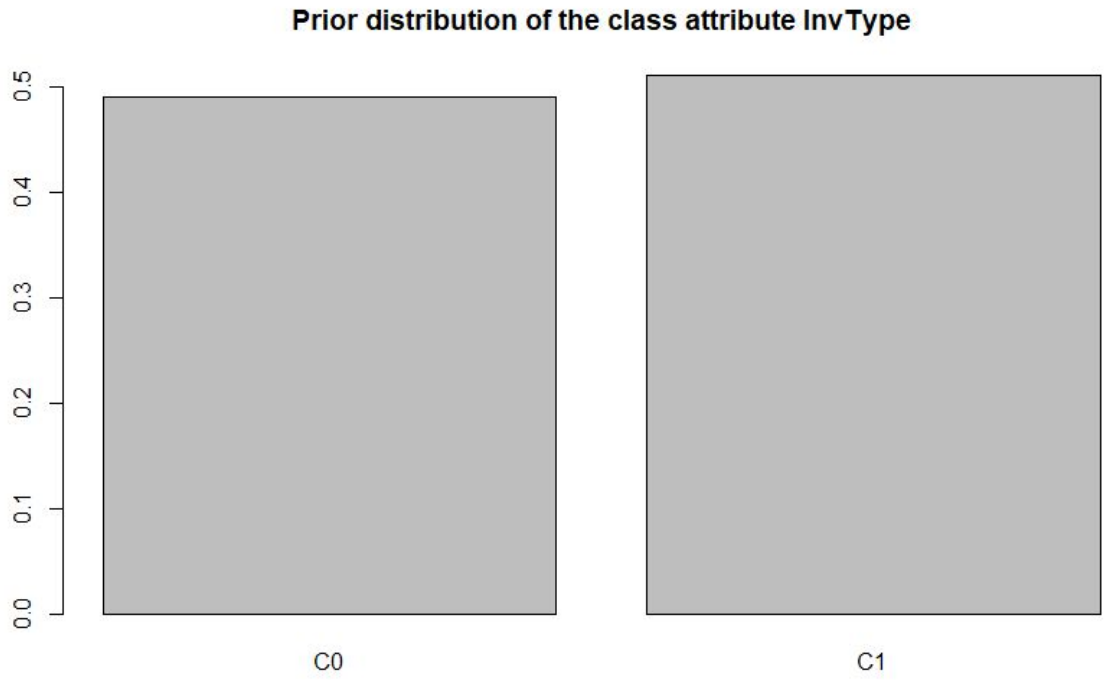
- Moyenne d'instances correctement classées : **802.8**
- Moyenne du pourcentage d'instances bien classées : **50.8745247148289 %**

Comme nous pouvons le constater, le default classifier a été moins performant que Naive Bayes pour toutes les expériences et tournait autour des 50%.

Ce dernier est considéré comme le baseline (ligne de base) et nous permet de savoir si on peut garder ou jeter notre modèle créé avec Naive Bayes. Dans notre cas, notre modèle est plus performant que le default classifier et nous pouvons donc le garder.

Visualisation avec des barplots

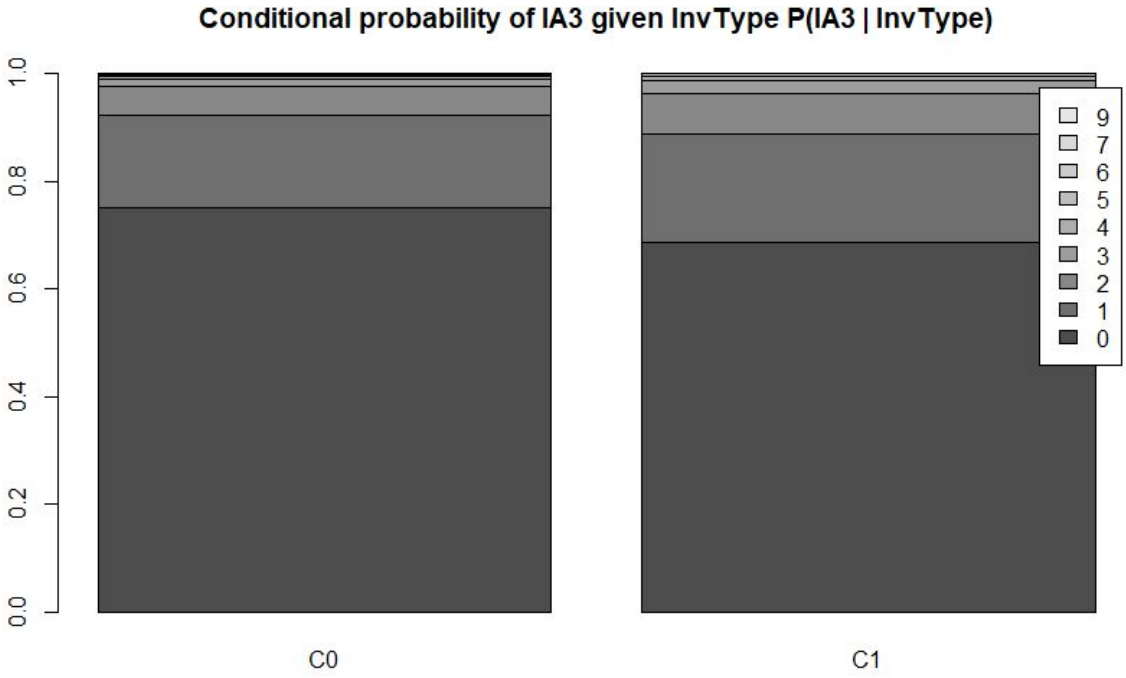
Distribution de la variable cible InvType



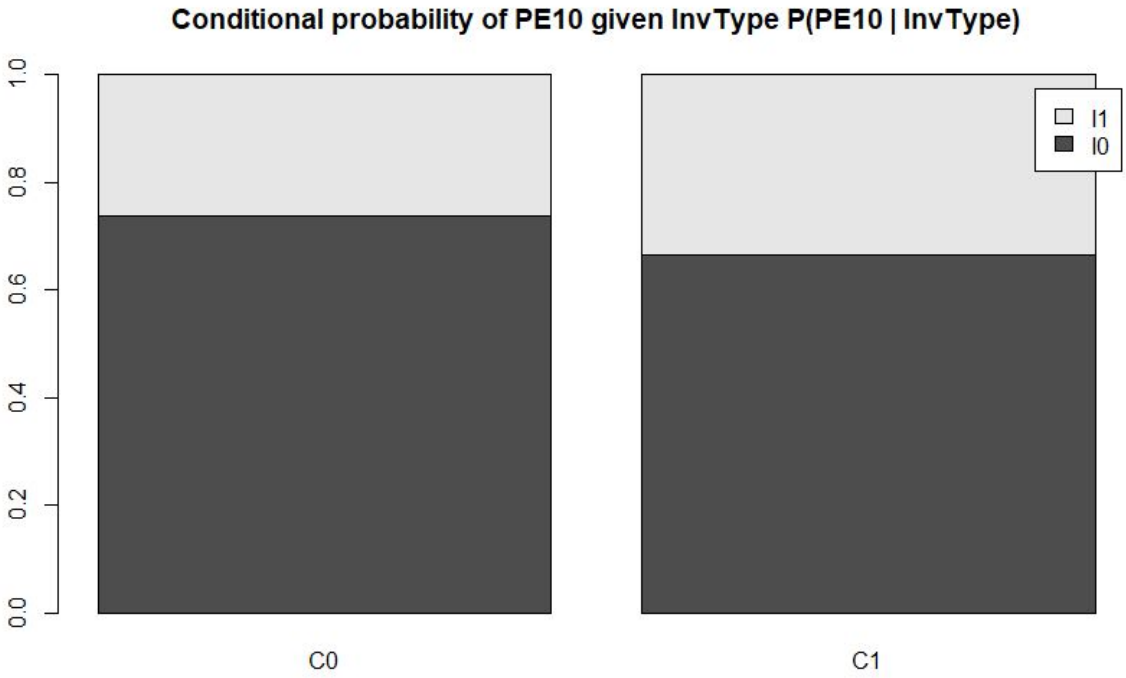
Comme on peut le voir, les probabilité de C0 et C1 sont presque identiques, avec la part de C1 légèrement plus élevée.

Les barplots suivants représentent la distribution de probabilité des variables prédictives discrètes selon la variable cible.

Variable prédictive discrète IA3



Variable prédictive discrète PE10



Ces probabilités conditionnelles sont celles qui apparaissent dans le modèle créé par Naive Bayes.

DEUXIEME PARTIE

Dans cette deuxième et dernière partie, nous allons voir de plus près et expliquer comment l'algorithme Naive Bayes fonctionne.

Explication

Pour déterminer si un client aura plus de chance de choisir le produit d'investissement C0 ou C1, il procède par un calcul qui est une simple multiplication. Ce calcul sera répété pour chaque valeur de la variable cible, donc deux fois dans notre cas. Ci-dessous, on a les différents éléments qui composent ce calcul :

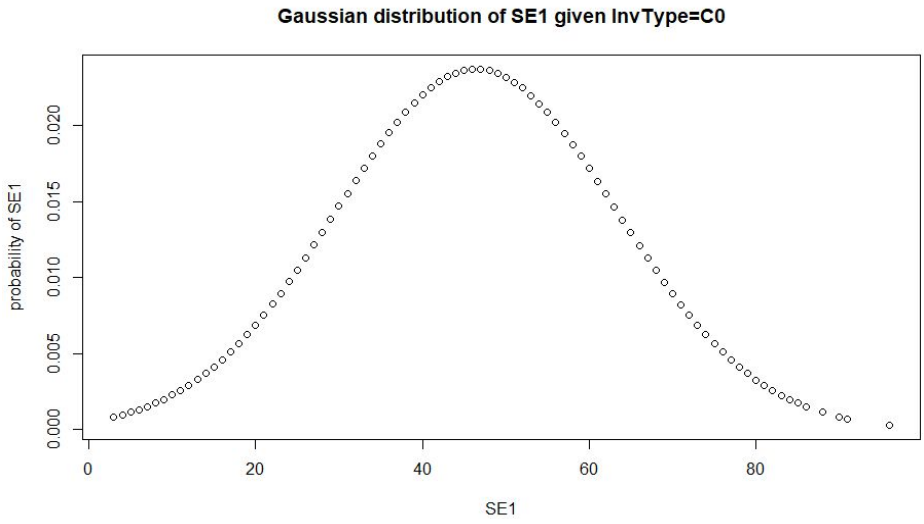
1. La probabilité de invType = C0 ou C1

2. Pour les variables discrètes, nous allons prendre les probabilités conditionnelles qui sont disponibles dans le modèle et qui correspondent aux valeurs des variables de l'instance. Par exemple :

$$P(IA3 = 0 \mid InvType = C0) \text{ ou } P(IA3 = 0 \mid InvType = C1)$$

3. Pour les variables continues, nous devons d'abord dessiner la courbe de Gauss de chacune d'elles et nous récupérerons la probabilité qui nous intéresse. Prenons par exemple une instance dont SE1 = 18 :

Le graphe ci-dessous représente la distribution normale pour SE1 sachant invType = C0 :



a. Selon la courbe, la probabilité que SE1 = 18 sachant invType = C0 est 0.005
b. C'est cette valeur que nous inclurons dans le calcul

Demonstration

Pour mieux comprendre l'algorithme, nous allons choisir une instance au hasard dans le jeu de données et voir, étape par étape, comment il sera classifié.

Voici l'instance que nous avons choisi :

```
     SE1 SE2 BA1 BA2   BA3  BA4  BA5 BA6  BA7 PE1 PE2 PE3 PE4 PE5 PE6 PE7 PE8 PE9 PE10 PE11 PE12 PE13 PE14 PE15 IA1 IA2
2771  62 G20   7   0 11868 7332 7332   0 7332  I0  I0  I0  I0  I0  I0  I0  I0  I0   I0   I0   I0   I0   I0   I0   0   0
     IA3 InvType
2771   0      C1
```

Nous pouvons voir que la valeur d'invType pour l'instance n°2771 est C1.

Faisons comme si Naive Bayes n'était pas au courant de cette information. Pour prédire l'invType de cette instance, il va procéder par le calcul suivant :

P(SE1, SE2, BA1, BA2, BA3, BA4, BA5, BA6, BA7, PE1,..., PE15, IA1, IA2, IA3 | invType = C0)

= P(invType = C0) * P(SE1 = 62 | invType = C0) * ... * P(IA3 = 0 | invType = C0)

Ce calcul est répété pour invType = C1. La prédiction de Naive Bayes sera la valeur de la variable cible pour laquelle le résultat de calcul est le plus élevé.

Comme on peut le voir ci-dessous, Naive Bayes a prédit C1, ce qui est pareil que le résultat réel. La prédiction est donc un succès pour cette instance.

```
> predict(nb,myData[2771,])
[1] C1
Levels: C0 C1
```

CONCLUSION

Dans ce TP, nous avons vu deux méthodes de classification, Naive Bayes et le Default Classifier. Pour notre jeu de données, Naive Bayes a été plus performant même si son niveau de précision n'était pas si élevé.

Chaque algorithme possède ses avantages et inconvénients, et pour Naive Bayes le fait qu'il traite toutes les variables de manière indépendante peut être l'un ou l'autre selon le jeu de données.