Université de Mons

Datawarehousing and datamining

Travaux pratiques avec Weka

Auteur :
Maxime De Wolf

28 avril 2018

Table des matières

1 Weka : Tutoriel

1.1 Questions 17.1.9 et 17.1.10

Ces questions portent sur l'arbre de décision crée à partir du fichier *iris.arff*. Voici donc l'arbre de décision obtenu :

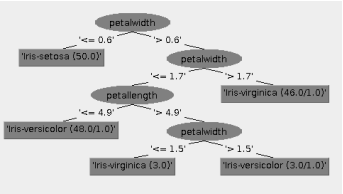


FIGURE 1 – Arbre de décision du *dataset iris.arff*

Question 17.1.9

Cette question consiste à évaluer la qualité de cet arbre (Figure 1) grâce à différentes options de tests. Ici, on effectuera ces tests une première fois avec le *dataset* complet et la 2^e fois avec la technique *10-fold cross-validation*. Nous comparons ensuite les résultats obtenus sur base des 2 *confusion matrix* :

(a) *Dataset* complet

a	b	c	
50	0	0	a = Iris-setosa
0	49	1	b = Iris-versicolor
0	2	48	c = Iris-virginica

(b) *10-fold cross-validation*

a	b	c	
49	1	0	a = Iris-setosa
0	47	3	b = Iris-versicolor
0	2	48	c = Iris-virginica

TABLE 1 – *Confusion matrix* obtenues grâce à deux méthodes de test différentes

Nous remarquons que le test sur le *dataset* complet classifie correctement 98% des instances tandis que ce chiffre descend à 96% avec le test *10-fold cross-validation*. Tester le modèle avec le *dataset* complet est une mauvaise idée car il donne une estimation optimiste de la qualité du modèle. En revanche, *10-fold cross-validation* permet de se faire une bonne idée de la généralisation du modèle et offre donc une meilleure mesure de qualité.

Question 17.1.10

En observant la localisation de ces erreurs, nous remarquons que certaines instances de classe *Iris-Verginica* ont des valeurs d'attributs équivalentes à celles d'instance de classe *Iris-Versicolor*. Le modèle n'a donc aucune chance de les différencier si nous voulons éviter l'*overfitting*. D'autre part, nous remarquons que l'instance de classe *Iris-Setosa* qui a été mal identifier aurait dû être correctement classé selon l'arbre de décision final obtenu.

1.2 Questions 17.2.4 à 17.2.11

Question 17.2.4

Le but de cette question est d'étudier la précision du classificateur *5-nearest neighbor* en fonction des attributs utilisés lors de cette classification. Ici, nous exécutons cette algorithme sur le *dataset glass.arff* et nous le test grâce à la technique *10-fold cross-validation*. Les résultats ainsi obtenus sont résumés dans la table suivante :

TABLE 2 – Précision obtenue en utilisant *IBk* pour différents sous-ensemble d'attributs

Nombre d'attributs	Attribut retiré	Précision de la classification
9	$\emptyset$	67.757
8	Si	71.4953
7	Fe	73.3645
6	Al	73.3645
5	Na	74.2991
4	Ba	74.7664
3	K	72.4299
2	Ca	71.9626
1	Mg	52.8037
0	RI	35.514

Grâce à ce tableau, nous remarquons donc que la précision de *IBk* sur le *dataset* complet est de 67.757% alors que nous obtenons une précision de 74.7664% une fois que nous retirons les attributs *Si, Fe, Al, Na, Ba* du *dataset*. Ce qui nous donne un gain d'environ 7% de précision.

Question 17.2.5

Cette question demande de critiquer la pertinence de la précision maximum obtenue dans la question précédente. Ou, en d'autres mots, cette estimation est-elle biaisée ou non ? Etant donné que le test du modèle est effectué sur le *dataset* d'entraînement, cette estimation est effectivement biaisée.

Question 17.2.6

Cette question nous demande de constater l'effet du bruit sur un modèle construit grâce à *IBk*. Pour ce faire, nous allons faire varier le pourcentage de bruits ainsi que la taille du voisinage dans les paramètres d'*IBk*. Ainsi, une estimation de la précision sera calculé avec *10-fold cross-validation*. Il est important de noter que ce test se fera sans les ajouts de bruits. Il n'y a donc présence de bruit que lors de la phase d'entraînement mais pas lors de la phase de test. La table suivante résume les résultat obtenus :

TABLE 3 – Effet du bruit sur la précision d'*IBk*, en fonction de différentes tailles de voisinage

Pourcentage de bruit	k = 1	k = 3	k = 5
0%	70.6%	72.0%	67.8%
10%	62.6%	69.6%	64.5%
20%	50.5%	63.1%	61.7%
30%	47.2%	58.4%	59.8%
40%	41.1%	54.7%	55.1%
50%	33.2%	44.4%	45.3%
60%	27.1%	35.5%	35.5%
70%	20.1%	28.5%	29.0%
80%	14.0%	21.0%	21.0%
90%	7.9%	13.6%	9.3%
100%	4.7%	7.9%	7.5%

Question 17.2.7

Cette question nous demande de critiquer les résultats obtenues lors de la question précédente. Plus particulièrement, on nous demande l'effet qu'a une augmentation du bruit au niveau de la classe. La Table 3 nous permet de constater que cette augmentation réduit la précision du classificateur *k-nearest neighbor* et ce peu importe la valeur du k.

Question 17.2.8

Cette question s'intéresse plutôt aux effets qu'a la modification de la valeur de k pour le classificateur *k-nearest neighbor*. Dans la Table 3 nous remarquons qu'augmenter la valeur de k rend le modèle obtenu plus robuste au bruit.

En effet, nous remarquons que bien qu'il n'y ait pas de différence significative entre les modèles obtenus pour k = 3 et k = 5, ils sont plus résistants au bruit que ceux obtenus pour k = 1.

Question 17.2.9

Pour cette question, nous devons comparer les classificateurs *IBk* et *J48* en fonction du pourcentage de l'ensemble d'apprentissage utilisé. Ces résultats sont encodés dans la table suivante :

TABLE 4 – Variation de la précision du modèle en fonction de la taille de l'ensemble d'apprentissage pour *IBk* et *J48*

Pourcentage de l'ensemble d'apprentissage	*IBk*	*J48*
10%	52.8%	45.3%
20%	63.6%	53.3%
30%	60.3%	59.3%
40%	63.6%	65.0%
50%	62.6%	63.1%
60%	64.5%	69.2%
70%	65.9%	67.8%
80%	67.8%	70.1%
90%	67.3%	69.6%
100%	66.8%	68.2%

Question 17.2.10

Cette question s'intéresse à l'effet de l'augmentation de nombre d'éléments de l'ensemble d'entraînement. Nous pouvons répondre à cette question grâce à la Table 4. Nous observons ainsi que plus l'ensemble d'entraînement est grand, plus la précision du modèle augmente jusqu'à un certain seuil où elle devient constante. Nous constatons que ce seuil se situe aux environs de 67% pour *IBk* et 69% pour *J48*.

Question 17.2.11

On nous demande ici d'identifier laquelle des deux techniques -*IBk* et *J48*- est la plus affectée par l'augmentation de la taille de l'ensemble d'entraînement. En nous référant à la Table 4, nous observons que *IBk* a une précision initiale de 52.8% pour arriver finalement à une précision de 66.8%. La précision du modèle *IBk* subit donc une variation de 14%. Pour *J48*, la précision du modèle varie de 45.3% à 68.2% ce qui donne une variation de 22.9%. Comme le modèle *J48* à une plus grande variation que *IBk*, nous concluons que *J48* est la technique la plus sensible à la taille de l'ensemble d'apprentissage.

1.3 Questions 17.3.1 à 17.3.11

Question 17.3.1

Pour résoudre cette question, nous utilisons le *boundary visualizer* de *Weka*. Cet outil sert à visualiser graphiquement les prédictions d'un modèle, dans notre cas, ce modèle est produit par le classificateur *1R*. La Figure 2 montre le résultat obtenu. Nous y voyons que la prédiction du modèle ne se fait uniquement qu'en fonction de la largeur des pétales (axe Y).

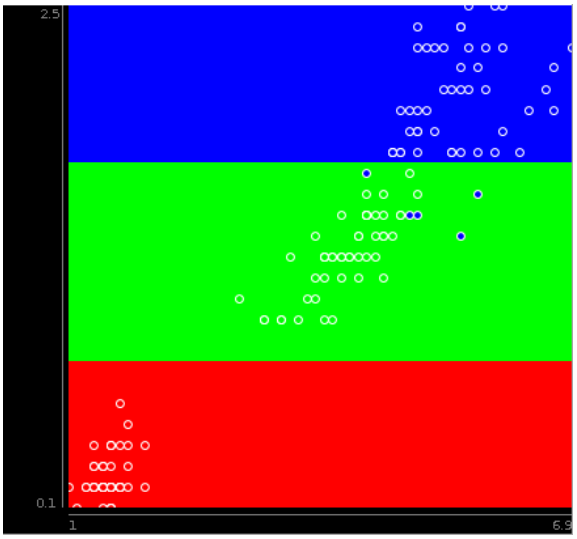


FIGURE 2 – Prédiction du type d'iris en fonction de la longueur des pétales (axe X) et de la largeur des pétales (axe Y) grâce au modèle 1R

Question 17.3.2

Pour résoudre cette question, nous avons du jouer avec le paramètre *minBucketSize* du classificateur *1R*. Après avoir testé plusieurs valeurs, nous pouvons conclure que ce paramètre ne change pas les résultats obtenus lors de la question précédente (voir Figure 2).

Nous pensons que c'est parce les valeurs que peuvent prendre la largeur des pétales (axe Y) varient entre 0.1 et 2.5. Il ne faut donc que 3 *buckets* pour discrétiser ces valeurs. Etant donné le nombre d'instances de ce jeu de données, la valeur de *minBucketSize* n'est utilisée que si elle n'est supérieure ou égale à 50. 50 est la valeur frontière car il y a 50 instances associés à chaque label.

Question 17.3.3

On doit répondre à la question "Pourquoi n'y a-t-il pas plus de régions lorsque le *bucket* a une petite valeur minimale ?". Cela s'explique car il s'agit d'une valeur minimale et donc, comme sur ce jeu de données on a relativement beaucoup d'instances par *bucket*, la valeur de ce paramètre n'est pas prise en compte.

Question 17.3.4

Le nombre minimum de région est 1 et la valeur minimum du paramètre *minBucketSize* pour l'obtenir est 51. Cela peut s'expliquer simplement par le fait que chaque classe ne définit chacune que 50 instances.

Question 17.3.5

En utilisant le classificateur *IBK* avec k=1, nous obtenons la prédiction visible à la Figure 3. Comme expliqué dans l'énoncé, chaque point ne possède qu'une des trois couleurs (rouge, vert ou bleu) car k vaut 1 et donc, le modèle ne prédit pas un point pouvant appartenir à plusieurs classe.

Nous remarquons néanmoins une zone du graphique qui fait exception à cela car sa couleur est une nuance de bleu (et pas totalement bleu). Nous pouvons justifier cela car il existe plusieurs instances qui ont les même longueur et largeur de pétale mais qui sont associés à des classes différentes. La zone possédant une couleur nuancée est en fait la zone qui contient ces instances.

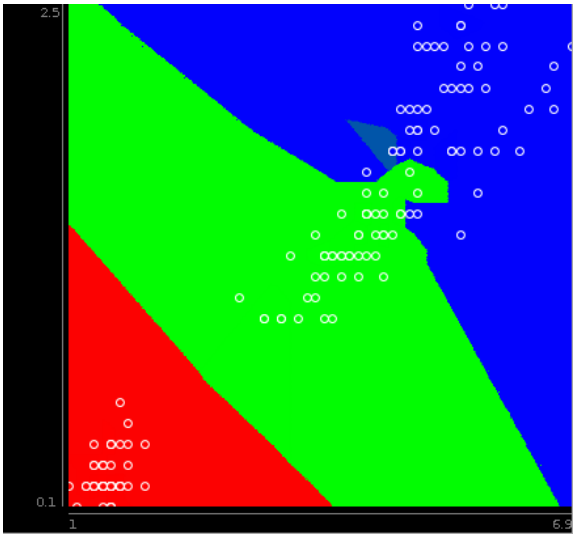


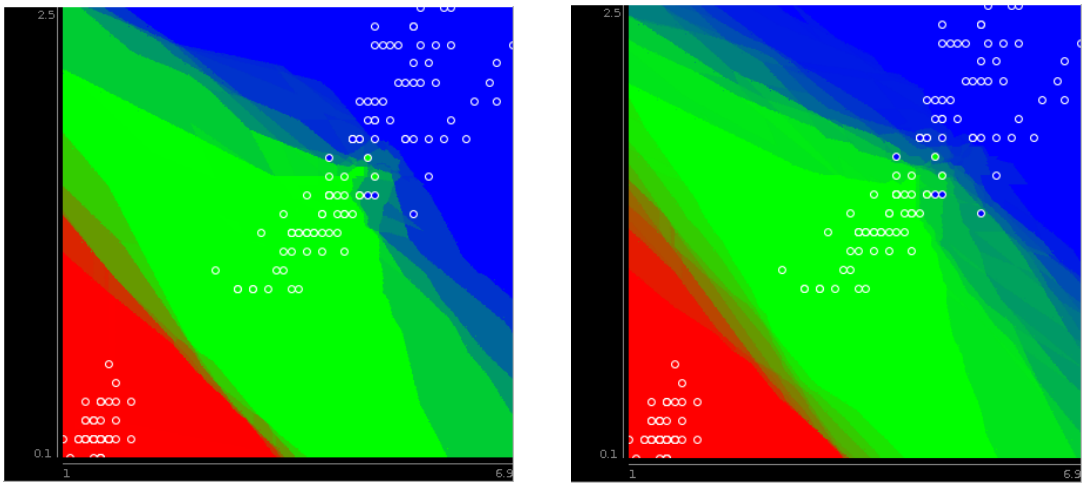
FIGURE 3 – Prédiction du type d'iris en fonction de la longueur des pétales (axe X) et de la largeur des pétales (axe Y) grâce au modèle IBK (avec k=1)

Question 17.3.6

En augmentant les valeurs du paramètre k, nous remarquons que les zones de la prédiction sont de plus en plus nuancées comme nous le montre la Figure 4.

Question 17.3.7

2 CoIL Challenge 2000



(a) k=5 (b) k=10

FIGURE 4 – Prédiction du type d'iris en fonction de la longueur des pétales (axe X) et de la largeur des pétales (axe Y) grâce au modèle IBK