Rapport de Projet en Intelligence Artificielle

Julien Sahli et Gaëtan Bossy

May 24, 2020

1 Tâche 1

Nous avons utilisé *Pandas* pour importer les données d'entraînement en *.csv*, puis nous les avons parsées afin qu'elles correspondent au template nécessaire à notre algorithme *ID3*.

2 Tâche 2

Similairement à la tâche précédente, nous avons utilisé *Pandas* pour importer les données d'entraînement, puis nous les avons parsées, et enfin nous avons prédit leur label en utilisant la fonction *classifie* et l'indexation -1 permettant d'accéder au dernier élement d'une *string*.

3 Tâche 3

Par souci d'efficacité, nous avons implémenté pour cette partie deux manières différentes de générer les règles. La première consiste à parcourir l'intégralité de l'arbre et de générer une règle pour chaque nœud terminal ; c'est le moyen le plus rapide de les générer toutes. Cependant, lorsqu'il est question de calculer la règle s'appliquant à un cas particulier, plutôt que de comparer les données de ce cas particulier aux prémisses de chaque règle une par une, ce qui nécessiterait une complexité proportionelle au nombre de règles, nous parcourons à la place l'arbre de décision et reconstituons la règle cherchée, ce qui se fait en complexité logarithmique du nombre de règles.

4 Tâche 4

La fonction *diagnostic* fonctionne de la manière suivante : pour chaque règle résultant en un état de santé sain, on compare chaque prémisse de cette règle avec les données du patient à diagnostiquer. Chaque différence avec les données incrémente le nombre de changements à appliquer pour guérir le patient, exception faite de l'âge et du sexe du patient, qui ne

peuvent être changés ; dans le cas où ceux-ci sont différents des données du patient, la règle ne pourra pas être sélectionnée. Finalement, la règle ayant le plus petit nombre de changements à appliquer est sélectionnée, et affichée si le paramètre *verbose* est vrai. La fonction prend également comme paramètre *modifiable_traits*, le nombre maximum de changements possibles, afin de déterminer si le patient peut être soigné ou non et retourne une valeur différente selon si celui-ci est sain, soignable, ou non soignable. Cela permet de calculer le nombre total de patients soignables.

5 Tâche 5

Quelques modifications de *ID3*, implémentées dans la classe *ID3_advanced* et *NoeudDeDecision_advanced*, se sont révelées nécessaires pour implémenter cette tâche:

- Nous avons calculé l'entropie de chaque séparation potentielle des données pour chaque attribut plutôt que de simplement choisir un attribut et de séparer complétement pour chaque valeur de cet attribut. Ceci a nécessité une modification des fonctions calculant l'entropie et les différentes probabilités ainsi que l'implémentation de quelques fonctions similaires.

- Nous avons modifié le système qui retire les attributs de la liste des attributs potentiels pour un noeud afin qu'il ne retire un attribut que lorsque l'ensemble des données d'un noeud possède la même valeur pour cet attribut. Pour ce faire nous avons décidé de recalculer la liste des attributs et de leur valeurs potentielles à chaque création d'un nouveau noeud dans notre fonction récurrente. Il serait probablement possible d'avoir une solution légèrement plus efficiente à ce problème mais la simplicité de notre solution nous convient.

- Nous avons modifié la création des enfants pour s'assurer que chaque noeud aie zéro ou deux enfants correspondants au valeur en dessus ou en dessous du seuil de séparation du noeud.

- Nous avons adapté le système de classification pour refléter qu'il n'existait plus toujours un enfant avec la valeur exacte de l'attribut.

- Nous avons réglé quelques problèmes causé par la représentation qui était prévue pour des *strings*.

Ces cinq modifications effectuées, nous avons procédé similairement aux tâches 1 et 2 afin d'importer les données d'entraînement et de test, excepté que les données sont transformées en *float* au lieu de *string* et que cela change quelques détails d'implémentation.