Rapport de projet - DSL

Philémon Houdaille
Matthieu Rodet

November 29^{th} 2021

1 Introduction

L'invention du Domaine Specific Language (DSL) PerfectML se place dans le cadre du projet du module DSL. Il a pour but de facilement décrire la méthode et les paramètres de classification d'un ensemble d'objets stockés dans un fichier CSV. L'utilisateur doit initialement posséder un tel fichier qu'il précisera dans son programme, ainsi qu'un fichier d'arrivée dans lequel seront envoyés les résultats. Il pourra alors dans son programme préciser quelle méthode de classification et quels paramètres utiliser.

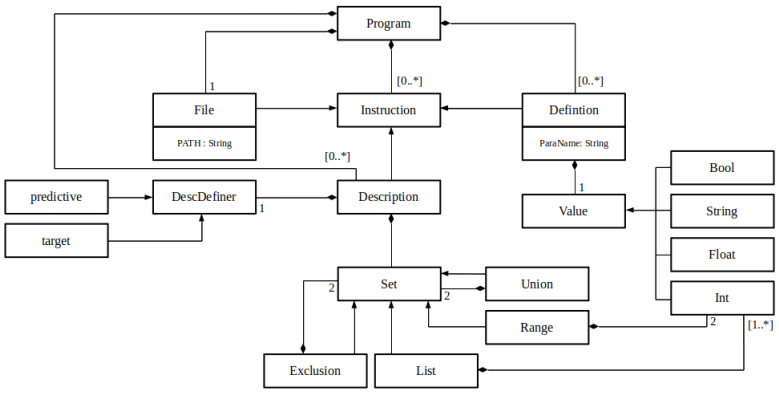
Nous avons restreint le choix des algorithmes de classification à trois : Multi-Layer Perceptron Classifier (MLPC), Support Vectors Classification (SVC) et K-Neighbours Classifier (KNC). Pour chacun, une grande variété de paramètres optionnels est proposé, 23 pour MLPC, 14 pour SVC et 7 pour KNC. Si l'utilisateur ne renseigne pas de valeur pour un paramètre, une valeur par défaut est proposée. Un dernier paramètre, commun à tous les algorithmes, est le nombre de lignes servant à l'entrainement de l'algorithme.

Le format du CSV d'entrée est donc un certain nombre de lignes égal au nombre renseigné dans le paramètre précédent dont tous les champs sont renseignés, d'autres lignes dont les colonnes indiquées comme "predictive" peuvent ou non être renseignées (elles seront ignorées). Le résultat se présente sous la forme d'un document CSV, dont le nom est le chemin d'accès est à renseigner, contenant les colonnes indiquées "predictive" des objets n'ayant pas servi à l'entrainement.

2 Présentation du langage

2.1 Syntaxe abstraite

Notre premier méta-modèle est le suivant :



Nous l'avons par la suite raffiné dans la grammaire, tout en gardant le même esprit.

2.2 Syntaxe concrète

La grammaire détaillée de PerfectML est disponible dans le fichier MyDsl.xtext. Pour la résumer, un fichier peut contenir plusieurs programmes, chacun divisé en cinq parties. La première contient le fichier à prendre en entrée. La seconde le fichier dans lequel enregistrer la sortie. La troisième contient l'algorithme à utiliser et tous ses paramètres associés. La quatrième renseigne le taux (en pourcentage) ou le nombre (en lignes) d'entrées servant à l'entraînement. Enfin, la cinquième, optionnelle, permet de décrire quels champs servent de cible et quels champs sont à prédire. Par défaut, cette dernière partie considère que seul le dernier champ est à prédire, les autres servant de cible. De plus, un champ non renseigné dans cette dernière partie sera par défaut cible, sauf la dernière colonne qui par défaut est à prédire.

3 Tests

Le premier fichier de test, PerfectMLParsingTest, test que le parsing se passe sans erreurs.

Pour le reste, les tests se concentrent sur la vérification que le code généré par le compilateur est conforme à nos attentes. Tous les codes Python attendus ont été exécutés pour vérifier de leur bon fonctionnement. Le fichier PerfectMLCompileTest teste la compilation de deux programmes à la suite dans un même fichier. Le fichier PerfectMLCompileTest2 teste la compilation des programmes des trois algorithmes avec tous leurs paramètres définis.

Conclusion

Notre langage offre finalement un moyen clair et concis de classifier à l'aide de trois algorithmes de machine learning un ensemble de données. Nous Avons développé autour de ce langage un compilateur vers python ainsi qu'un interpréteur. En outre Eclipse nous offre un environnement de programmation intelligent, avec une autocomplétion et des suggestions de mot-clefs.

Avec davantage de temps, il aurait pu être intéressant de proposer un ensemble plus complet de tests, un deuxième compilateur vers le langage R, une documentation détaillée ainsi qu'un benchmark comparant les performances de nos programmes en Python, R et de notre interpréteur.