Fouille de Données Massives
TD - Fondamentaux en Apprentissage

M2 Informatique - SISE (2022-2023)

Guillaume Metzler
Institut de Communication (ICOM)
Université de Lyon, Université Lumière Lyon 2
Laboratoire ERIC UR 3083, Lyon, France

guillaume.metzler@univ-lyon2.fr

A propos des scripts

La correction se compose de trois fichiers python

- *functions.py*
- *main.py*
- *latex.py*

Le fichier *functions.py*. Il contient l'ensemble des fonctions qui permettent de **charger et de mettre en forme vos données**. Il y a aussi deux fonctions qui permettent de créer **l'ensemble des n-uplets** d'hyper-paramètre à tester lors du processus de cross-validation ainsi qu'à **appliquer et évaluer les performances** de votre algorithme selon différents critères :

F-measure, G-measure, G-mean et Balanced Accuracy

Vous êtes bien sûr libre d'ajouter d'autres mesures de performances selon le contexte, à vous de les définir. N'hésitez pas à modifier ce script si vous souhaitez ajouter d'autres algorithmes/méthodes.

Le fichier *main.py*. Il sera utilisé pour définir les valeurs d'hyper-paramètres que vous souhaitez tester en fonction des algorithmes testés (voir *listParams*).

Le fichier *latex.py*. Ce dernier script vous permettra de passer des résultats stocké dans python à un fichier *.pdf* dans lequel vous pourrez voir vos résultats sous la forme d'une table. Voir la section suivante pour l'obtention. Je ne donne pas de détail sur ce code, cela n'est pas l'objet de la séance et c'est plus un outil pour vous.

Exécution des scripts

La procédure ci-dessous est effectuée directement dans un terminal (Mac/Linux)

1. Le fichier *main.py* dépend de deux arguments (voir l'usage de *sys.argv* au début du fichier main) qui seront fournis au moment de l'exécution du fichier python.
   (a) le premier argument représente la graine employée
   (b) le deuxième argument représente la mesure de performance employée : **f1** ou **ba** ou **gm** ou **g1**.
   Les noms sont définis dans le fichier *functions.py* et vous pouvez en définir d'autres.

   Par exemple, pour exécuter le code vous pourrez exécuter la commande suivante pour lancer votre procédure en mettant la *seed* à la valeur 1 et en évaluant les modèles en terme de F-measure.

   ```
   # python3 main.py 1 f1
   ```

   Les résultats sont affichés dans le terminal, mais peu lisibles ... mais ce n'est as grave. Après avoir fait appel au fichier main, vos résultats sont contenus dans un répertoire *results* mais dans un format peu lisible.

2. Exécuter le fichier latex.py

   ```
   # python3 latex.py
   ```

   Il va créer un répertoire *latex* dans lequel va se trouver un document *.pdf* qui contiendra les tableaux de vos résultats. Et on pourra s'arrêter là.

Idéalement, on effectue la procédure plusieurs fois (en général 10) afin que nos modèles soient testées sur plusieurs ensembles tests et on regarde les performances moyennes et l'écart-type obtenus sur les 10 runs.
La procédure précédente permet de faire cela. Exécuter les commande suivantes dans votre terminal :

```
# for i in {1..10}; do python3 main.py $i f1; done

# python3 latex.py
```

Cela va permettre d'exécuter 10 fois votre script python mais en changeant la valeur de la graine.
Un équivalent sous Windows est donné par :

```
# for /L %i in (1,1,5) do python main.py %i f1

# python3 latex.py
```

On pourrait aller encore plus loin dans le processus en exécutant les scripts en parallèle (sur 2 coeurs) comme suit :

```
# for i in {1..10}; do screen -d -m bash -c "export OMP_NUM_THREADS=2;
python3 main.py $i f1"; done

# python3 latex.py
```

Les programmes seront exécutés en "arrière plan", vous pouvez suivre l'avancement avec la commande

```
# top
```