TP : Apprentissage supervisé

1 Introduction à RAPIDMINER

1.1 Pour commencer

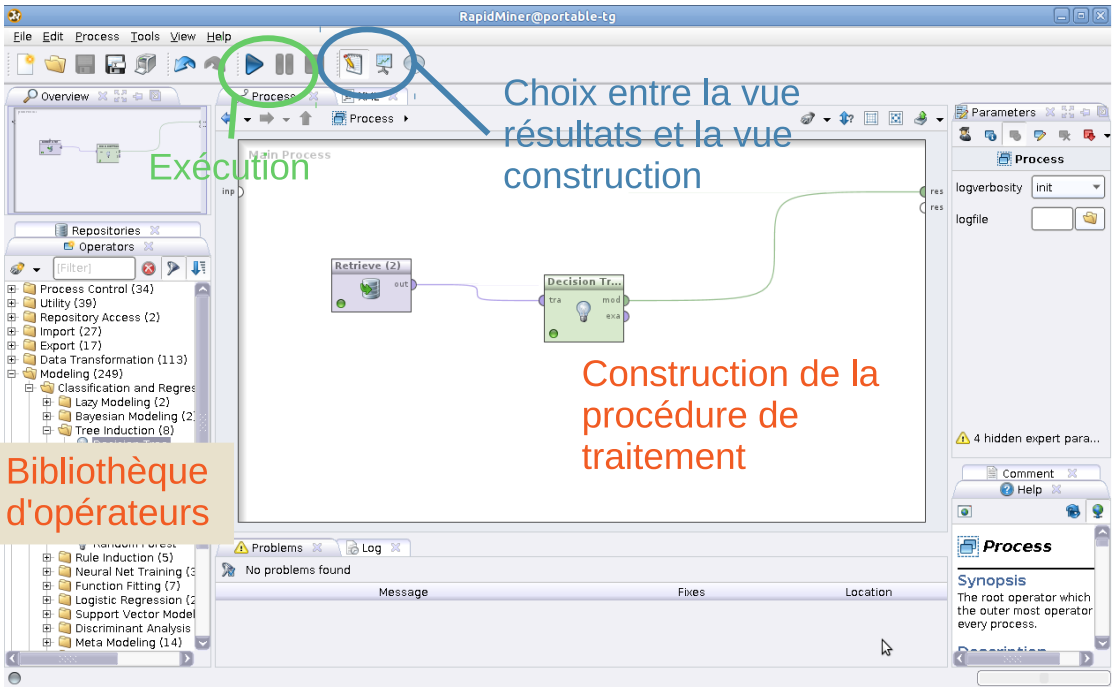
Le logiciel RAPIDMINER permet de créer graphiquement une procédure d'extraction automatique de "connaissances". L'importation des données, les prétraitements et les algorithmes se présentent sous la forme de boîtes ayant des entrées et des sorties qu'il faut connecter entre elles. L'intérêt du logiciel est d'une part de faciliter la conception d'une procédure d'extraction de connaissances et d'autre part d'offrir un grand nombre d'algorithmes d'apprentissage classiques.

RAPIDMINER est un logiciel gratuit qui fonctionne sur tous les systèmes d'exploitation (Windows, Linux, Mac). Pour ces TP, nous utiliserons une extension du logiciel qui permet d'utiliser des images geo-référencées. L'utilisation de cette extension nous contraint actuellement à fonctionner sous Linux. Pour éviter les questions fastidieuses d'installation, on vous fournit une "clé USB" sur laquelle RAPIDMINER est installé et configuré pour les TP.

Pour commencer le TP :

1. Démarrer votre ordinateur en *bootant* sur la clé USB (vous êtes l'utilisateur *liveuser* dans ce système),
2. Vous devriez trouver un icône sur le bureau pour lancer RAPIDMINER,
3. Lors de la première exécution de RAPIDMINER, il vous demande de donner un répertoire par défaut (*new local repository*) dans lequel il enregistrera vous expérimentations : créer un répertoire sur le bureau ou dans votre espace personnel (`/home/liveuser/`) et utiliser ce répertoire comme *new local repository* (les résultats seront sauvegardés sur votre clé).

La figure ci-dessous illustre l'interface du logiciel à laquelle vous parvenez.



Pour créer une procédure de traitement :

- glisser les opérateurs depuis la liste des opérateurs vers la zone centrale (ces opérateurs sont organisés par catégories de traitements)
- lier les opérateurs entre eux pour indiquer la transmission des données d'une boîte à l'autre : il faut cliquer une fois (en relachant) sur une entrée ou une sortie d'un bloc, puis cliquer une seconde fois sur l'autre extremité du lien.
- vous définissez la sortie du traitement en liant la sortie d'une boîte à la sortie `res` de la zone centrale.

Lorsque votre procédure a été définie, vous pouvez exécuter le traitement en cliquant sur l'icône de flèche (dans la barre d'icônes). Pendant le traitement, il indique par une petite flèche verte la boîte en cours d'exécution, les ronds verts indiquent les boîtes exécutées.

Si un problème survient, le processus est arrêté et un message d'erreur (plus ou moins clair) apparaît.

Si tout c'est bien passé, il vous propose de passer sur la "vue résultats" pour regarder les résultats obtenus.

Pour revenir au moide d'édition du traitement, il suffit de cliquer sur l'icône en forme de bloc-note avec un crayon.

1.2 Ajout de l'extension Weka

Dans la suite, nous aurons besoin de l'extension Weka qui n'est pas installée par défaut. Cette extension comprend un grand nombre d'algorithmes d'apprentissage supplémentaires qui nous seront utile.

Une connexion internet est requise pour cette opération.

1. aller dans le menu de `Help>Update RapidMiner ...`
2. sélectionner l'extension `Weka` (bouton Select/Deselect)
3. lancer l'installation
4. redémarrer RapidMiner (proposition automatique)

2 Exercices génériques

Ces exercices sont des mises en oeuvre des méthodes présentées en cours. Ils permettront d'expérimenter les algorithmes et ils vous permettront de découvrir le fonctionnement de RapidMiner.

Exercice 1 - Aller jouer au golf

Vous êtes entraineur dans club de golf et le joueur que vous entraînez semble capricieux quant à sa venue aux entrainements. Pour éviter de vous déplacer et attendre désespéremment la venue de votre élève, vous décidez d'apprendre son comportement. Pour cela, vous notez chaque jour les conditions climatiques :

- si le temps est ensoleillé, couvert ou pluvieux (attribut `Outlook`),
- la température (en degrés Fahrenheit),
- l'humidité relative (en %),
- si il y avait du vent ou pas,

et si l'élève est venu jouer au golf ou non (attribut `Play`).

L'objectif est de construire automatiquement un arbre de décision qui vous permettra de prédire si votre élève va venir jouer au golf en fonction des autres attributs.

On utilise RapidMiner pour construire l'arbre.

Question a) *Lancer* RapidMiner *avec une expérimentation vide*

Question b) *Ajout du bloc de données*

Affichez la liste des "repositories", puis faites glisser l'élément `Samples>data>Golf` dans la zone centrale.

Connectez ensuite la sortie de ce bloc (`out`) à la sortie du processus (`res` à droite de la zone).

Lancer l'exécution de l'expérimentation (utiliser l'icône de lecture dans la barre d'outils ou la touche F11). Le logiciel vous propose d'enregistrer votre travail, faites le. Puis vous passez en mode d'affichage des résultats.

Ici, le résultat de l'expérimentation permet de visualiser les données. La vue meta-données affiche des statistiques élémentaires sur les attributs, la vue des données présente le tableau des données et la vue graphique visualise les données par des représentations graphique configurables.

Dans la vue des données, l'attribut `Play` est affiché dans une couleur différente car il s'agit de l'attribut de classe.

Question c) *Visualisation des données* À partir des données, visualiser les données à l'aide d'un "Scatter Plot" en prenant comme axes l'humidité et la température, et en prenant comme couleur l'attribut "Play" (c'est à dire le label). *Remarque : il faut respecter les majuscules/minuscules et il ne faut pas mettre les guillemets.*

Est-il possible de définir un arbre de décision permettant de classer parfaitement les classes "no" et "yes" uniquement à partir des attributs `Température` et `Humiditée` ? Si oui, quel serait son inconvénient principal ?

Pour revenir à l'édition de l'expérimentation, il faut cliquer sur l'icône en haut à droite dans la barre d'outils.

Question d) *Utilisation d'un fichier `CSV`*

- supprimer le bloc `Retrieve` et remplacer le par un bloc `Import>Data>CSVExampleSource`,
- ce type de bloc permet de charger un jeu de données enregistré dans un fichier `csv` (Comma Separate Values),
- sélectionner le bloc et sélectionner le fichier de données `Golf.csv`

Relancer l'exécution, les résultats devraient être les mêmes que précédemment dans la mesure où les données sont les mêmes.

En revanche, pour les fichiers `CSV`, le logiciel ne "sait" pas quel attribut doit servir d'attribut de classe. Il n'y a pas d'attribut en "rose". Il faut donc lui indiquer explicitement, **cela sera nécessaire pour l'utilisation des méthodes d'apprentissage supervisé**.

- Ajouter un bloc `Data Transformation>Name and role conversion>Set role`
- Connecter la sortie de `CSVExampleSource` à ce bloc et la sortie de ce bloc à la sortie du processus,
- Configurer le bloc pour indiquer que l'attribut "Play" (attention aux majuscules) est de type "label".

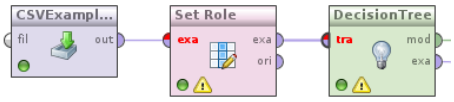


FIGURE 1 – Processus de lecture d'un fichier CSV

Question e) *Apprentissage d'un arbre de décision*

Ajouter un bloc d'apprentissage d'arbre en faisant glisser l'opérateur : `>Modeling >Classification >Tree Induction >DecisionTree`. Puis connectez le bloc `Set role` au bloc `Tree Induction`

Relancer l'exécution de l'expérimentation et observer le résultat.

- Tous les exemples sont-ils classés correctement par ce modèle ? Sinon, rechercher un exemple mal classé dans les données ?
- Quelle modification du jeu de données n'aurait aucune conséquence sur l'apprentissage ? Que peut-on en déduire sur le comportement de l'élève ?

Question f) *Comparaison d'algorithmes*

Ajouter un nouveau bloc `>Modeling >Classification >Weka >Trees >W-J48`. L'algorithme *J48* est un autre algorithme pour construire des arbres de décision.

NB : Il est possible de connecter la sortie d'un bloc à plusieurs blocs en utilisant un `IO Multiplier`. Lors de la connection de ce nouveau bloc aux données d'entrée, le logiciel vous propose de résoudre un problème de connexion : choisissez "*Insert IO multiplier ...*" pour pouvoir faire comparer les deux algorithmes.

Configurer ce bloc en sélectionnant le paramètre `B` du bloc `W-J48`. Ce paramètre oblige l'algorithme à construire un arbre binaire (c'est-à-dire qui dérive systématiquement deux branches à partir d'un nœud).

- Comparer les deux résultats ?
- Que constate-t-on sur l'utilisation de l'attribut `Outlook` ?
- Tous les exemples sont-ils classés correctement par ce modèle ? Sinon, rechercher un exemple mal classé dans les données ?

Continuer l'expérimentation ajoutant un troisième algorithme d'apprentissage d'arbre de décision. Par exemple, utiliser l'apprentissage d'arbre par la méthode de *W-RandomTree* (localisé dans `>Modeling >Classification >Weka >Trees >W-RandomTree`). Même questions que précédemment :

- Comparer les deux modèles ?
- Que constate-t-on sur l'utilisation de l'attribut `Outlook` ? Que peut on en déduire ?
- Tous les exemples sont-ils classés correctement par ce modèle ? Sinon, rechercher un exemple mal classé dans les données ?

Recommencer l'expérimentation ajoutant l'algorithme d'apprentissage d'arbre de décision `ID3` et relancer l'expérimentation. Que signifie le message d'erreur ?

Question g) *Utilisation du modèle* Quel modèle des données vous semble le plus approprié ? À partir des différents modèles construits, indiquer si vous devez attendre votre élève si il pleut, que l'humidité est supérieure à 90, qu'il fait 65 degrés Farenheit et qu'il n'y a pas de vent.

Question h) *Conclusions sur l'apprentissage d'arbres de décision ?*

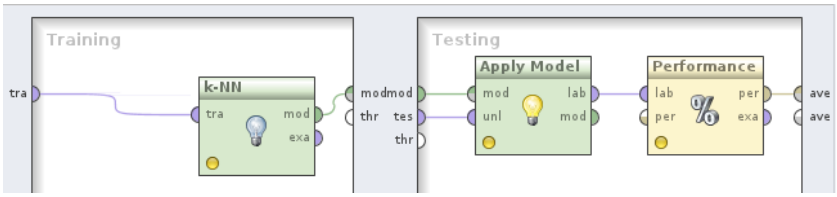
Exercice 2 - Validation croisée avec RAPIDMINER

On construit maintenant une expérimentation avec RAPIDMINER utilisant l'ensemble des données des Iris (150 exemples) pour évaluer la méthode de K-plus-proches voisins par validation croisée.

Question a) *Mise en place d'une validation croisée dans* RAPIDMINER

Réaliser les étapes suivantes (dans l'ordre afin d'éviter des complications). À la fin de la manipulation, vous devriez avoir une "operator tree" ressemblant à la figure plus bas.

1. Ajouter un nouveau de données `Samples>data>Iris`,
2. Ajouter un nouveau bloc `Evaluation>Validation>XValidation` (Cross validation). Ce bloc est un meta-bloc : vous pouvez cliquer dessus pour "rentrer dans ce bloc" et y placer deux sous-blocs : l'un avec la méthode d'apprentissage (à gauche), l'autre avec la méthode d'évaluation,
3. Méthode d'apprentissage : double cliquer sur le bloc `XValidation`, ajouter un bloc `modeling>classification>Lazy>NearestNeighbors` et configurer la valeur de K.
4. Méthode d'évaluation :
   - Ajouter un bloc `ModelApplier` (dans `Modeling>Model Application`) qui applique le modèle appris aux données de test fournies par la validation croisée.
   - Ajouter un bloc `Evaluation>Validation>PerformanceEvaluator` qui calcule une matrice de confusion.



Lancez l'expérimentation !

- La matrice de confusion vous semble-t-elle correspondre à une bonne classification ?
- Quelle classe est la plus facile à apprendre ?

Question b) *Modifier manuellement la valeur de `K` pour voir l'effet sur la qualité des classifications proposées.*

Question c) *Automatisation des tests sur les `K`* On fait de l'informatique pour ne pas avoir à répéter les opérations. Nous allons donc créer un processus pour automatiser les tests sur différentes valeurs de `K`

Pour cela, nous allons avoir besoin du bloc `Process Control>Parameters>Optimize Parameters (Grid)`. Il s'agit de nouveau d'un meta-bloc.

1. Insérer un bloc `Optimize Parameters (Grid)`
2. Couper le bloc `XValidation`
3. Entrez dans le bloc `Optimize Parameters (Grid)`

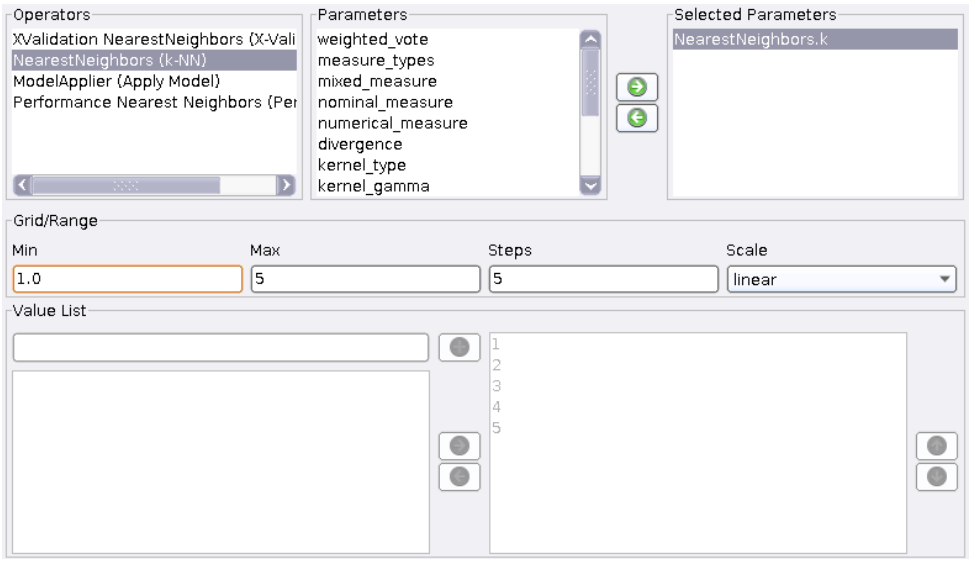


FIGURE 2 – Parametrage des valeurs de k à tester.

4. Coller le bloc `XValidation` (il contient toujours toute la partie Apprentissage)
5. Dans le bloc `Optimize Parameters (Grid)`, relier l'entrée au bloc `XValidation`
6. Editer les paramètres du bloc "d'optimisation" en le configurant comme illustré à la Figure 2

Bon maintenant, on va avoir besoin d'un bloc qui va retenir les performances de notre classifieur pour chaque valeur de k. Pour cela, on utilise un bloc `Utility > Logging > Log`. Ce bloc est connecté en entrée au bloc `XValidation` et en sortie de l'intérieur du bloc de `Optimize Parameters (Grid)`.

La configuration de se bloc est réalisé comme ci-dessous :

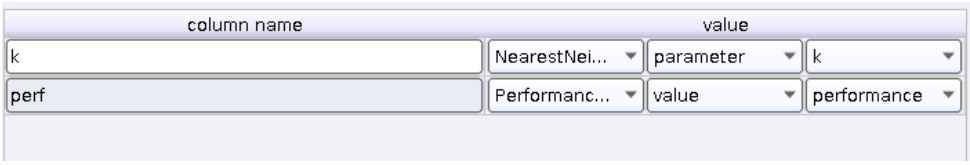


FIGURE 3 – Configuration du bloc de log

Exécuter le processus. Vous devriez alors obtenir une table de valeurs. Sur les cas des données l'iris, il n'y a pas grand chose à dire sur le résultat ...

Question d) *Comparaison de modèles*

Ajouter de nouveau bloc de validation croisées dans lequel vous utiliserez d'autres algorithmes d'apprentissage. Notez que vous pouvez copier-coller le bloc validation croisée. Les algorithmes qui peuvent être utilisés sont, par exemple, les suivants : *ID3Numerical* et *NaiveBayes* (*cf.* Figure ci-dessous).

- Selon le critère de précision (*accuracy*), quel est le meilleur modèle pour ces données ?
- Es-ce toujours les mêmes classes qui posent problème ?

Exercice 3 - Construction d'un réseaux de neurones pour classer les iris

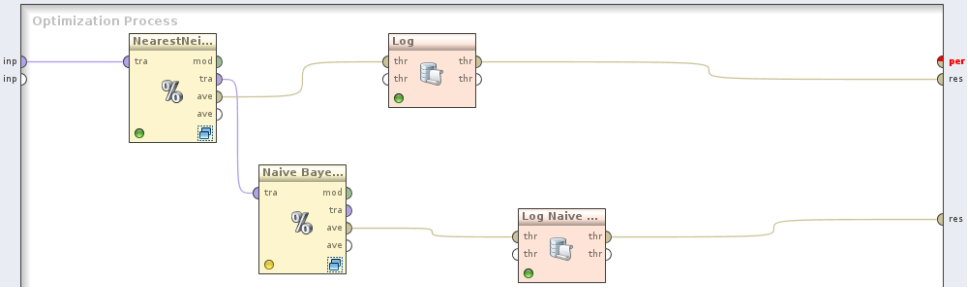


FIGURE 4 – Process XValidation avec plusieurs blocs d'apprentissage à comparer (vue générale)

Dans cet exercice, on vous fait construire un réseau de neurone (perceptron multi-couche).

Question a) *Construction d'un réseau multi-couche* Dans un premier temps, on construit une chaîne de traitement très simple :

1. Ajouter un nouveau bloc de données avec les Iris.
2. Ajouter un nouveau bloc `Modeling>Classification>Neural Net Training>Neural Net`.
3. Le nombre de couche et le nombre de neurones par couche peut être configuré en configurant les `hidden_layer` : cliquer sur `add` pour ajouter une couche avec dans la colonne de gauche son nom, et dans la colonne de droite le nombre de neurones pour la couche.

Vous pouvez alors lancer les traitements.

En cliquant sur les neurones du réseau, vous pouvez obtenir des informations sur les valeurs qui ont été apprises.

Que déduire du résultat construit par RAPIDMINER ?

Question b) *Validation croisée*

Reprendre la chaîne de traitement permettant de faire une comparaison de modèle de l'exercice précédent et appliqué le avec un apprentissage d'un réseau de neurone.

Quelle méthode donne les meilleurs résultats ?

3 Application à la classification d'une image de télédection

Dans la liste des opérateurs, vous devriez avoir une catégorie nommée "Geospatial Data Mining". Cette catégories d'opérateurs permet d'importer ou d'exporter des données spatiales sous une forme tabulaire.

L'exemple de la Figure 5 permet de charger des données sous une forme correcte :

- le bloc `ShapeSampler` permet de charger les données issue d'un shapefile contenant la vérité terrain (ground truth)
- les données de se bloc sont utilisées deux fois :
  - une fois pour prendre l'information de la classe à l'aide du premier bloc `Select Attribute`
  - une fois pour servir au bloc `GeoTiffReader` comme coordonnées pour aller chercher les informations des bandes d'une image GeoTiff
- Les deux données (la classe provenant du shapefile et les données des bandes) sont ensuite réunies pour constituer un tableau unique (bloc `Union_ExampleSet`),

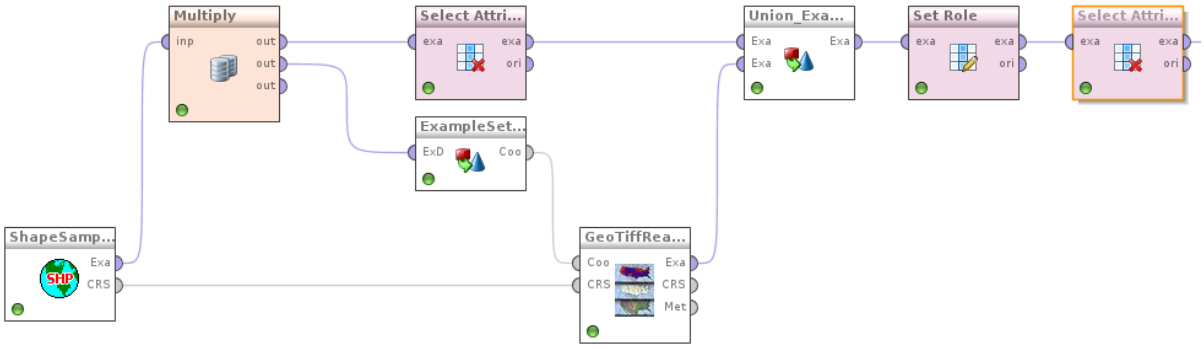


FIGURE 5 – Processus de chargement des données geospatiales

– Finalement, on définit un attribut comme label de classe pour faire de l'apprentissage supervisé et on supprime les deux attributs de coordonnées.

Exercice 4 - Classification supervisée pour la détection de plantes invasives

L'objectif de l'exercice est de montrer l'utilisation de RAPIDMINER avec des images satellite et de mettre en œuvre une méthodologie (rapide) de d'évaluation et de comparaison de méthodes d'apprentissage. Les données utilisées ont été acquises et préparés par E. Athané au laboratoire PSN, en partenariat avec l'IAV en 2012.

Question a) *Chargement des données*

1. Utiliser le menu `File>Import process ...` pour importer le processus `TestGeo.rmp` correspondant au processus de la Figure 5.
2. Configurer correctement les fichiers d'images :
   – utiliser le fichier `ROI_4_classes.shp` comme shapefile,
   – utiliser le fichier `glenac_hyspex_15b.tif` comme geotiff, il s'agit d'une image contenant les 15 premières bandes [1] d'une image HYSPEX (? ? ? spécification des bandes ? ? ?).
3. Tester la procedure pour vérifier qu'elle permet bien de charger les données

Question b) *Classification par SVM* Compléter le processus pour utiliser un algorithme de SVM (utiliser la version `LibSVM`) et d'évaluer, par validation croisée, les performances d'un SVM (noyau gaussien, paramètres par défaut).

Question c) *Choix des paramètres du SVM* Évaluer pour quels couples de paramètres du SVM (gamma et C) les performances sont les meilleurs (dans le temps du TP, vous vous limiterez à tester 5 valeurs de chaque paramètres, soient 25 tests).

Question d) *Classification d'une image* Modifier le processus de sorte à ajouter la construction d'une couche raster contenant le résultat de la classification de tous les pixels de l'image.

Question e) *Changement de l'image* Recommencer les expérimentations en utilisant l'image `MNF_glenac_hyspex_15b.tif`. Cette image correspond aux 15 bandes d'une image traitée par un algorithm MNF [2] (Maximum Noise Fraction Transform) (Green et al., 1988).

Question f) *Comparer les performances avec un autre algorithme d'apprentissage supervisé*

1. Je limite à 15 bandes pour des raisons de temps de calcul et de contrainte de mémoire vive.
2. MNF voir `http://www2.dmu.dk/rescoman/scanner/scan/postproc/Image/Noise/sc1maxnoisefractrans.htm`.