Fonctionnement algorithme :

L'algorithme de Boruta est une enveloppe construite autour de l'algorithme de classification de la forêt aléatoire mis en œuvre dans le paquetage R randomForest (Liaw et Wiener 2002). L'algorithme de classification random forest est relativement rapide, peut généralement être exécuté sans réglage des paramètres et il donne une estimation numérique de l'importance des caractéristiques.

Il s'agit d'une méthode d'ensemble dans laquelle la classification est effectuée par le vote de plusieurs classificateurs faibles non biaisés

- arbres de décision.

Ces arbres sont développés indépendamment sur différents échantillons de l'ensemble d'apprentissage. La mesure d'importance d'un attribut est obtenue comme la perte de précision de la classification causée par la permutation aléatoire des valeurs d'attribut entre les objets. Elle est calculée séparément pour tous les arbres de la forêt qui utilisent un attribut donné pour la classification. Ensuite, la moyenne et l'écart-type de la perte de précision sont calculés.

Alternativement, le Z-score est calculé en divisant la perte moyenne par son écart-type peut être utilisé comme mesure d'importance. Malheureusement, le Z-score n'est pas directement lié à l'importance statistique de l'importance de la caractéristique retournée par l'algorithme de forêt aléatoire, puisque sa distribution n'est pas N(0, 1) (Rudnicki, Kierczak, Koronacki et Komorowski 2006). Néanmoins, dans Boruta nous utilisons le Z-score comme mesure d'importance car il prend en compte les fluctuations de la perte de précision moyenne entre les arbres de la forêt.

Puisque nous ne pouvons pas utiliser directement le Z-score pour mesurer l'importance, nous avons besoin d'une référence externe pour décider si l'importance d'un attribut donné est significative, c'est-à-dire si elle est discernable de l'importance qui peut résulter de fluctuations aléatoires. A cette fin, nous avons étendu le système d'information avec des attributs qui sont aléatoires par conception. Pour chaque attribut, nous créons un attribut "fantôme" correspondant, dont les valeurs sont obtenues en mélangeant les valeurs de l'attribut original entre les objets. Nous effectuons ensuite une classification en utilisant tous les attributs de ce système étendu et calculons l'importance de tous les attributs.

L'importance d'un attribut fantôme ne peut être différente de zéro qu'en raison de fluctuations aléatoires. Ainsi, l'ensemble des importances des attributs fantômes est utilisé comme référence pour décider quels attributs sont sont vraiment importants.

La mesure d'importance elle-même varie en raison de la stochasticité du classificateur de la forêt aléatoire. De plus, elle est sensible à la présence d'attributs non importants dans le système d'information (également les attributs fantômes). De plus, elle dépend de la réalisation particulière des attributs fantômes. Par conséquent, nous devons répéter la procédure de remaniement pour obtenir des résultats statistiquement valides.

En bref, Boruta repose sur la même idée que celle qui est à la base du classificateur de la forêt aléatoire, à savoir qu'en ajoutant de l'aléatoire au système et en recueillant les résultats de l'ensemble d'échantillons aléatoires, on peut réduire l'impact trompeur des fluctuations et des corrélations. Ici, ce caractère aléatoire supplémentaire doit nous permettre d'avoir une vision plus claire des attributs qui sont réellement importants.

\*MZSA => « Maximum Z-score Shadow attribute » OU « Z-score maximum des attributs fantôme »

L'algorithme de Boruta comprend les étapes suivantes :

1. Étendre le système d'information en ajoutant des copies de toutes les variables (le système d'information est toujours étendu par au moins 5 attributs fantômes, même si le nombre d'attributs dans l'ensemble initial est inférieur à 5).
2. Mélangez les attributs ajoutés pour supprimer leurs corrélations avec la réponse.
3. Exécutez un classificateur de forêt aléatoire sur le système d'information étendu et rassemblez les Z-score calculés.
4. Trouver le Z-score maximum parmi les attributs fantômes (MZSA), puis attribuer un résultat à chaque attribut dont le score est supérieur à MZSA.
5. Pour chaque attribut d'importance indéterminée, effectuez un test bilatéral d'égalité avec le MZSA.
6. Considérer les attributs dont l'importance est nettement inférieure à la MZSA comme "sans importance" et les supprimer définitivement du système d'information.
7. Considérer les attributs dont l'importance est significativement supérieure à la MZSA comme "importants".
8. Supprimez tous les attributs d'ombre.
9. Répétez la procédure jusqu'à ce que l'importance soit attribuée à tous les attributs, ou que l'algorithme ait atteint la limite précédemment fixée pour les exécutions de la forêt aléatoire.

En pratique, cet algorithme est précédé de trois cycles de démarrage, avec des critères d'importance moins restrictifs. Les cycles de démarrage sont introduits pour faire face aux fortes fluctuations des scores Z lorsque le nombre d'attributs est important au début de la procédure. Au cours de ces tours initiaux, les attributs sont comparés respectivement au cinquième, troisième et deuxième meilleur attribut fantôme ; le test de rejet n'est effectué qu'à la fin de chaque tour initial, tandis que le test de confirmation n'est pas effectué du tout.

La complexité temporelle de la procédure décrite ci-dessus dans des cas réalistes est d'environ O(P - N), où P et N sont respectivement les nombres d'attributs et d'objets. Cela prend du temps pour les grands ensembles de données ; mais cet effort est essentiel pour produire une sélection statistiquement significative de caractéristiques pertinentes.

Pour illustrer les propriétés de mise à l'échelle de l'algorithme de Boruta, nous avons réalisé l'expérience suivante en utilisant l'ensemble de données Madalon. Il s'agit d'un ensemble de données artificielles, qui était l'un des problèmes du NIPS2003.(Guyon, Gunn, Ben-Hur et Dror 2005).

Le jeu de données contient 2000 objets décrits par 500 attributs. 500 attributs. Nous avons généré des sous-échantillons de l'ensemble Madelon contenant 250, 500, 750, . . . , 2000 objets. Ensuite, pour chaque sous-échantillon, nous avons créé sept ensembles étendus contenant respectivement 500, 1000, . . . , 3500 attributs superficiels obtenus comme un bruit aléatoire uniforme. Nous avons ensuite effectué sélection de caractéristiques standard avec Boruta sur chacun des 64 ensembles de test et mesuré le temps d'exécution.

Les résultats de l'expérience sont présentés dans la figure 1. On peut voir une échelle linéaire presque parfaite pour le nombre croissant d'attributs. D'autre part, les temps d'exécution augmentent plus vite que le nombre d'objets, mais la différence n'est pas très grande et semble converger vers une échelle linéaire pour un grand nombre d'objets.

Les temps sont rapportés en heures de CPU. En utilisant les valeurs du plus grand ensemble de données, on peut estimer le temps nécessaire pour exécuter Boruta sur un seul cœur d'un processeur moderne à une heure pour un million (attributs × objets). Heure pour un million (attribut × objets).

Il convient de noter que dans de nombreux cas, en particulier pour les problèmes biomédicaux, le temps de calcul ne représente qu'une petite fraction du temps nécessaire à la collecte des données. On peut également noter que la principale raison d'utiliser l'algorithme de sélection de caractéristiques "révélateur" n'est pas la réduction du temps de calcul (bien que cela puisse être réalisé).

La raison principale est de trouver tous les attributs pour lesquels leur corrélation avec la décision est plus élevée que celle des attributs aléatoires. De plus, alors que Boruta est généralement un algorithme séquentiel, le classificateur de la forêt sous-jacent est une tâche trivialement parallèle, et Boruta peut donc être distribué même sur des centaines de cœurs, à condition d'utiliser une version parallèle de l'algorithme de forêt aléatoire.