Polytech Sorbonne

Projet Machine Learning

Statistiques, Apprentissage supervisé

Mohamed Djerrab, Harold Gallice

MAIN5

Année 2018 - 2019

1 Introduction

Ce rapport est un compte rendu de l'analyse d'une base de donnée et de la mise en place de diverses techniques d'apprentissage supervisé. Ici, nous nous sommes intéressés à la classification de différents types d'arbre à partir d'observations faites sur une forêt. Il s'agit d'un problème de classification supervisé multi-classe avec 7 classes.

2 Exploration de la base de donnée

2.1 Description de la base de données

Les données étudiées sont les caractéristiques de différents arbres prélevés dans la forêt du parc National Roosevelt situé au nord du Colorado. La base de données étudiée présente 581012 échantillons. Ces échantillons sont détaillés avec 54 attributs exprimés comme des entiers. Une partie de ces attributs concerne des données qualitatives. Les différents éléments de cette base de données sont détaillés dans le tableau 2.

Attribut	type	unité/dimension
Élévation	entier	mètre
Azimut	entier	degré
Pente	entier	degré
Distance horizontale a un point d'eau	entier	mètre
Distance verticale a un point d'eau	entier	mètre
Distance horizontale à la route	entier	mètre
Ombrage à 9h	entier de 0 à 255	index
Ombrage à 12h	entier de 0 à 255	index
Ombrage à 15h	entier de 0 à 255	index
Ombrage à 12h	entier de 0 à 255	index
Distance au point de feu sauvage	entier	mètre
Partie de la forêt	binaire	dimension 4
type de sol	binaire	dimension 40
type d'arbre	entier de 1 à 7	index

Figure 2: Description des caractéristiques

2.2 Data Exploration

2.2.1 Répartition des classes

Dans un premier temps, nous observons la répartition des classes au sein de la base de donnée. Nous pouvons voir que les classes les plus représentées sont les arbres de type 1,2 et 3 (cf figure 3), qui composent plus de 80% de la base de données. Ceci doit être pris en compte lors des apprentissage statistiques, dans la suite du projet.

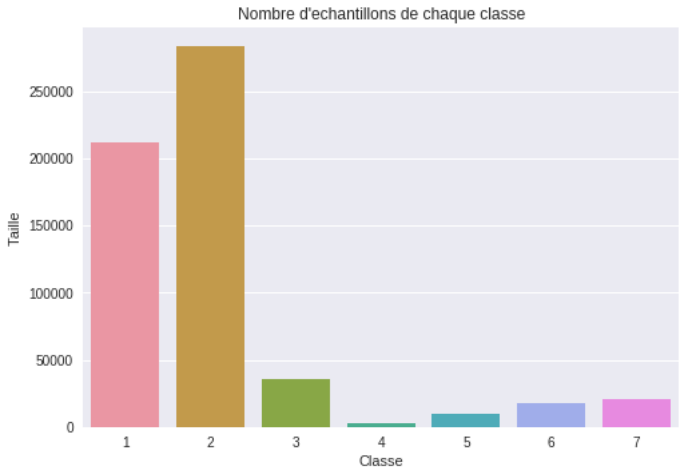


Figure 3

2.2.2 Distribution des attributs

Nous allons maintenant étudier les distributions des attributs à valeurs entières. Ceci permet de vérifier si les attributs suivent une loi normale, traduction d'indépendance entre les variables et de non corrélation. Après avoir affiché les distributions de chaque attributs à valeurs entière, nous constatons que l'*Orientation* (fig 4) a une distribution différente de celle des autres attributs. Cela peut certainement être expliqué par le fait que la variable représente un angle, qui peut donc être positive ou négative.

Par conséquent, nous pouvons dire que la distribution de l'*Orientation* n'est pas normale. Nous pouvons donc essayer de normaliser cette attribut en appliquant la racine carré sur ses variables. Nous obtenons alors une meilleure distribution (fig 5). Nous appliquons cela aux autres attributs de la base de données afin d'en améliorer la distribution.

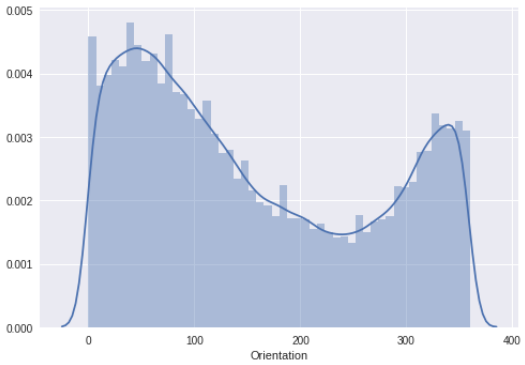


Figure 4: Distribution de l'attribut *Orientation*

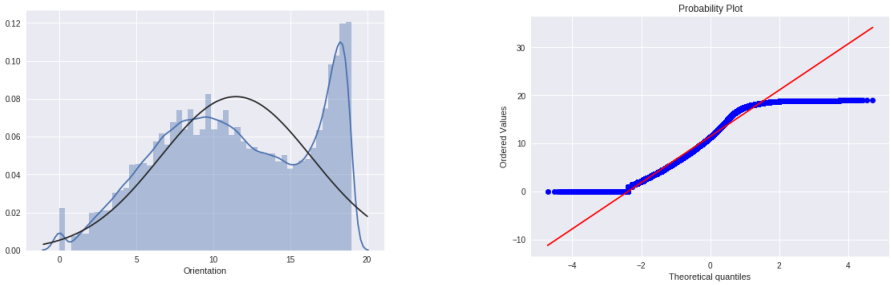


Figure 5: Travail de normalisation sur l'attribut *Orientation*

2.2.3 Distribution des attributs en fonction des classes

Dans le but de réaliser une classification supervisé, il est important d'étudier le pouvoir de discrimination des attributs. Pour cela, l'étude des boxplots de chaque attribut, en fonction de la classe, est un très bon moyen de le voir.

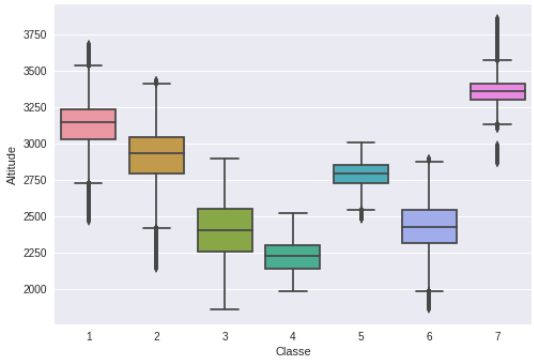


Figure 6: Boxplot *Altitude*

Nous pouvons constater dans la figure 6, que l'*Altitude* donne une distribution propre à chaque classe. Par conséquent, cet attribut est très important pour la prédiction lors d'une classification.

Pour les autres attributs, nous obtenons des distributions normale pour toutes les classes comme pour la *Pente* (fig 7).

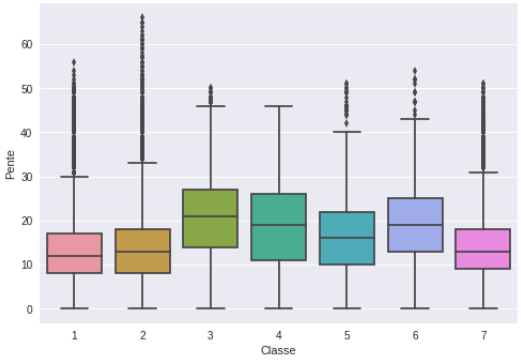


Figure 7: Boxplot *Pente*

Pour les attributs $Type\_Sol$ et $Wilderness$, leurs boxplots sont particuliers. Nous remarquons que certaines classes n'ont pas de distribution. On peut penser que les types d'arbres n'expriment pas certains attributs. Nous avons, par conséquent, un travail de sélection d'attribut à faire, car ils n'ont pas tous une grande importance.

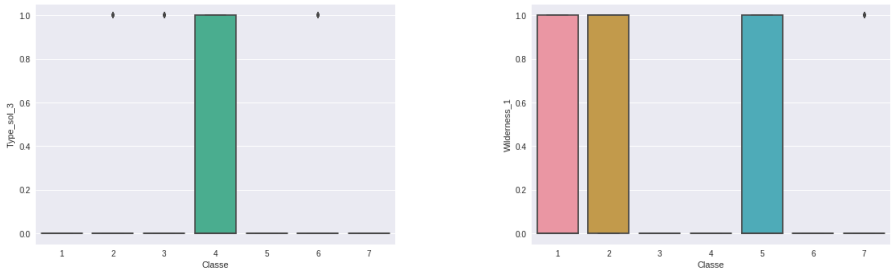


Figure 8: Boxplot des attributs $Type\_Sol.25$ et $Wilderness\_1$

2.2.4 Y a-t-il des corrélations entre les attributs ?

Le calcul d'une matrice de corrélation est un moyen d'évaluer la dépendance entre plusieurs variables en temps. Pour calculer celle basé sur le jeu de donnée du projet, nous avons utilisé le test de corrélation de Pearson et nous avons pris en compte les attributs à valeurs entière.

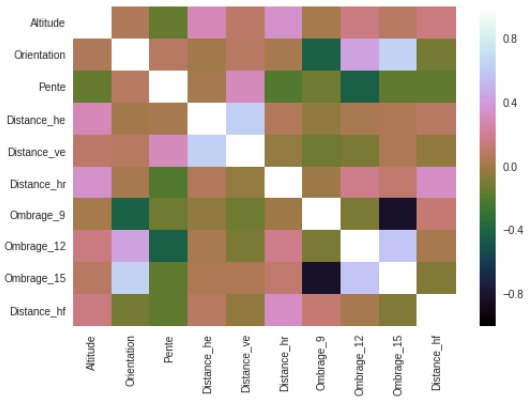


Figure 9: Matrice de corrélation

Nous constatons qu'il existe de forte corrélation entre les attributs suivant :

- Orientation - Ombrage_15 = 0.640536
- Distance_ve - Distance_he = 0.619386
- Ombrage_15 - Ombrage_12 = 0.573942
- Ombrage_9 - Ombrage_15 = -0.823424

3 Apprentissage Statistique

3.1 Un besoin d'équilibrage ?

La première analyse à effectuer est la vérification de l'équilibrage de nos classe. En effet, en vu de l'apprentissage que nous allons effectuer, l'équilibrage est primordial pour éviter les faux positifs. En effet, si une classe est présente avec 1% des échantillons dans cette classe sur la data totale, nous pouvons obtenir un score de 0.99 avec un modèle sur la data totale et pourtant ne jamais prédire qu'un échantillon appartient à cette classe sous représentées. Nous aurions donc globalement un bon score, mais localement (pour cette classe) un très mauvais score. Si le but de l'exercice est d'avoir un gros score, alors le modèle est bien choisi. Nous pouvons voir sur la figure 3 que certaines classes sont sous représentées. Particulièrement la classe 4. En effet nous avons les classes 1 et 2 avec beaucoup d'échantillons et les 5 autres classes sont bien moins présentes dans la base de données. C'est la première remarque importante à faire, nos classes vont avoir besoin d'un équilibrage.

A la vu de ce graphique, nous avons décidé d'orienter notre projet vers un but précis. En effet, nous avons compris qu'en créant un modèle uniquement basé sur les classes 1 et 2 nous pourrions obtenir un score conséquent directement. Nous avons donc décidé d'un objectif, avoir un score sur chaque classe de 0.75 minimum. C'est à dire, si nous tirons 1000 individus aléatoire d'une classe i, notre modèle sera capable de reconnaître qu'au moins 750 individus appartiennent à la classe i. L'avantage de cet objectif est de promettre un score global de 0.75 dans le pire des cas. L'inconvénient est que nous ne profitons pas

modèle	score global	classe 1	classe 2	classe 3	classe 4	classe 5	classe 6	classe 7
LogReg	0.68	0.70	0.78	0.66	0.38	0.0	0.05	0.03
MLP	0.53	0.51	0.71	0.00	0.00	0.00	0.00	0.00
Tree	0.85	0.84	0.88	0.84	0.66	0.65	0.73	0.82
KNN	0.90	0.90	0.91	0.88	0.57	0.69	0.78	0.90

Figure 10: score globaux et locaux des méthodes utilisées

de la sur-présence des classes 1 et 2 pour gonfler notre score global, mais c'est un choix que nous avons fait et expliqué.

3.2 Apprentissage par force brute

Tout d'abord, un apprentissage sans aucun traitements a été réalisé. Nos classes sont donc non équilibrées (cf figure 3) et nos individus contiennent 54 attributs. Différents modèles ont été entraînés sur une base de test contenant 10% de la base de donnée réelle. Deux types de modèles ont été mis en place. Les modèles permettant la classification de données linéairement séparables et les autres types de modèle, capable de classifier des données non-séparables par un hyperplan.

Parmi les types de modèles retenus nous pouvons citer la régression logistique ainsi qu'un réseau de neurone, multi-layer Perceptron Classifier (MLP) de couches (54, 15, 10, 12, 7). Ces deux modèles permettent la classification de données linéairements séparables. De plus, on a ajouté une fonction d'activation "relu" au perceptron pour permettre une classification moins linéaire. On les mettra en oeuvre afin de voir si une approche de la sorte est intéressante. Deux autres modèles ont été mis en place, une classification par arbre de décision et par K-nearest neighbors. Ces deux modèles peuvent classifier des données nons linéairement séparables. A l'aide de ces 4 modèles, nous devrions avoir une idée assez précise de la séparabilité de nos classes.

3.2.1 Séparation de notre jeu pour l'apprentissage

Notre jeu étant dense et ayant qualifié notre méthode de brutale, nous avons décidé de créer un jeu d'entraînement d'une façon précaire. Nous avons simplement tiré 10% des échantillons de notre jeu de donnée et nous entraînons nos modèle sur cet ensemble. Les données n'ont absolument pas été traitée, l'équilibrage laisse toujours à désirer. Les test ont été effectués d'abord globalement sur 5% de notre jeu de données tiré aléatoirement puis localement en tirant 2000 individus aléatoirement sur chaque classe et en les testant. Nous avons ainsi une connaissance du score de chaque classe individuellement. Les résultats sont accessibles dans la figure 10.

3.2.2 Analyse des résultats

Un constat saute aux yeux, les classes ne sont pas linéairement séparables. En effet les classifications grâce à des modèles linéaires, LogReg et MLP, offre des scores faibles par rapport aux méthodes non linéaires. Nous atteignons des scores plus que raisonnables pour la méthode KNN avec seulement 10% des données en apprentissage. De même pour la classification par arbre de décision,

ces deux méthodes ont des résultats très corrects mais nous apercevons comme prévu un déséquilibre sur la classification selon la classe. En effet, classifier des individus des classes 1 2 3 et 7 se fait plutôt bien. Mais les classes 4, 5 et 6 contiennent beaucoup d'erreurs quand à la classification de leurs individus. Ces résultats étaient attendus et déjà mentionnés dans la section précédente.

L'objectif proposé d'atteindre 75% dans toutes les classes n'est donc pas atteint. Mais le score est tout de même impressionnant. De plus, nous avons pu remarquer que les méthodes de classification par arbre de décision et des K plus proches voisins étaient très efficace pour notre jeu de données. Nous allons pouvoir essayer différents équilibrages avec ces méthodes.

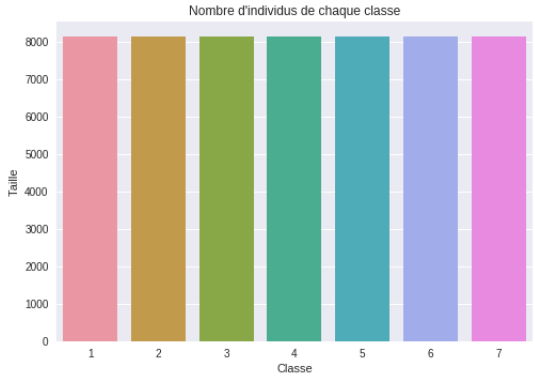
3.3 Équilibrage des classes

Il est important de préciser que pour tous nos test de performance, les données d'apprentissage n'excéderont jamais 10% du jeu de données initiale. Ainsi nos tests pourront être comparés en respectant le temps de calcul et la mémoire nécessaire à nos modèles. Comparer 2 modèles KNN n'ayant pas la même taille n'aurait aucun sens.

3.3.1 Des classes totalement équilibrées

Une première méthode serait d'équilibrer totalement nos classes. Ainsi chacune de nos classes constituerait $\frac{1}{7}$ de notre jeu de donnée d'entraînement.

Ce jeu d'entraînement est facile à mettre en place. Il suffit de choisir le bon nombre d'échantillons aléatoirement sur chaque classe. On effectuera du bootstrap sur les classes ne présentant pas assez d'échantillons pour compléter la taille requise.



Les résultats de cette simulation avec les modèles KNN et classification par arbre de décision se trouve dans le tableau figure 11. Nous sommes directement frappés par les scores des classes 4, 5, 6 et 7 qui sont maintenant très bien prédites. En effet, ces classes étaient très peu représentées dans le jeu de donnée précédent, ainsi la méthode KNN avait du mal à trouver des voisins immédiats aux nouveaux individus. Maintenant, la taille de ces classes pèse autant que les autres. De plus, tous les individus de la classe ou presque sont présent dans le jeu de test, la méthode KNN est donc très puissante pour trouver des individus similaires. Malgré tout, le score est bien plus faible. En effet les classes 1 et 2 sont maintenant moins bien prédites. Ce résultat était lui même attendu, nous avons un sous échantillon de nos classes 1 et 2. Nous passons de 28000 individus pour notre précédent test à 8000. Maintenant certains individus avec des caractéristiques propre ne sont plus représentés. Cette méthode n'est donc pas adaptée, sauf si le but était de seulement prédire avec efficacité les individus des classes minoritaires. Ce n'est pas notre objectif.

modèle	score global	classe 1	classe 2	classe 3	classe 4	classe 5	classe 6	classe 7
Tree	0.72	0.71	0.65	0.86	0.99	0.99	0.92	0.95
KNN	0.63	0.63	0.52	0.82	0.98	0.99	0.91	0.97

Figure 11: Scores globaux et locaux des méthodes utilisées sur le jeu équilibré

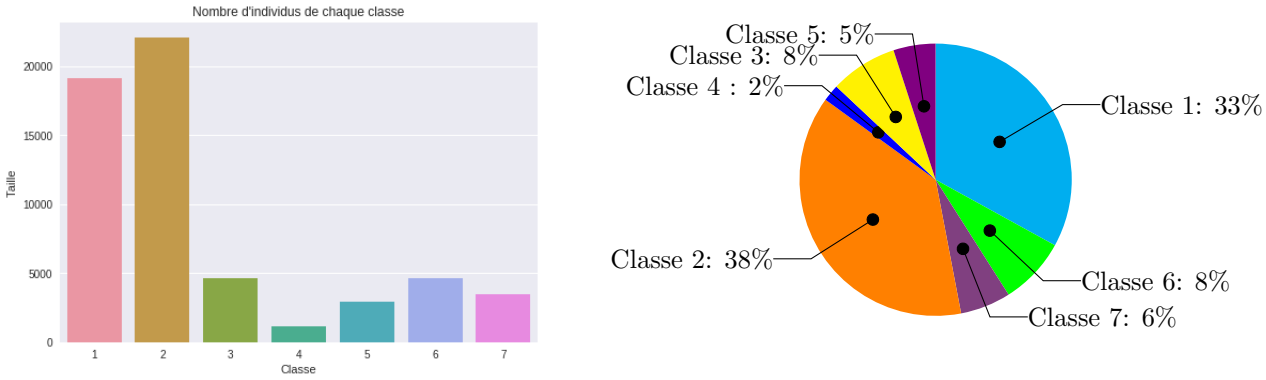


Figure 12: Équilibrage partiel utilisé

3.3.2 Des classes partiellement équilibrées

Après l'échec d'un équilibre parfait, nous tentons d'équilibrer nos classes avec un certain poids. Les classes majoritairement présentes dans le jeu de données de base auront un poids plus fort pour un meilleur entraînement et inversement pour les classes moins présentes. Le poids pour les classes faibles permettra tout de même un meilleur échantillon que si nous tirons aléatoirement comme fait précédemment. Après plusieurs essais nous statuons sur une répartition comme indiquée figure 12. Les résultats avec cette répartition sont disponible dans le tableau figure 13. Nous pouvons remarquer que la méthode par les arbres de décisions avec cette répartition permet une certaine homogénéité dans les scores des classes. Nous avons obtenu ce que nous cherchions.

modèle	score global	classe 1	classe 2	classe 3	classe 4	classe 5	classe 6	classe 7
Tree	0.80	0.81	0.83	0.90	0.84	0.84	0.84	0.82
KNN	0.76	0.79	0.74	0.67	0.76	0.85	0.80	0.81

Figure 13: Scores globaux et locaux des méthodes utilisées sur le jeu partiellement équilibré

3.4 Analyse de l'arbre

Nous avons décidé de regarder brièvement l'arbre de décision. Vous pouvez trouver dans la figure 14 l'arbre jusqu'à la profondeur 3. La première décision parvient déjà à séparer les classes 1, 5 et 7 d'un coté et de l'autre la classe 4. Cette décision est basé sur l'emplacement des arbres. Nous avons donc un emplacement (X[12]) qui est l'unique endroit ou pousse les arbres de la classe 4. De plus aucun arbre des classes 1, 5 et 7 ne pousse à cet emplacement. La

décision suivante sur la partie droite, les arbres de classe 2, 3, 4 et 6, porte sur la pente (X[2]). On peut voir que les arbres de la classe 4 peuvent pousser sur une pente de moins de 15 degré. En général les autres classes sont sur des pentes de plus de 15 degré. En regardant nos données, on remarque que les classes 2, 3 et 6 représentent des sapins. Ces pins semblent donc pousser sur un sol pentu. De l'autre coté, les arbres sont séparés selon le type de sol.

L'étude pourrait continuer encore longtemps, d'autant plus que l'arbre initial à une profondeur max de 42. Néanmoins le raisonnement semble assez logique et les résultats correspondent à l'objectif que nous avions fixé.

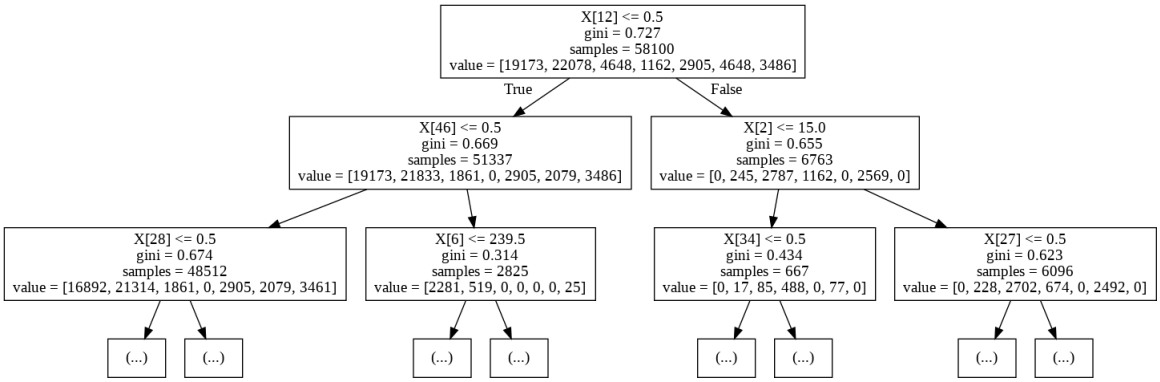


Figure 14: Arbre de décision profondeur 3

4 Conclusion

En fonction du jeu de données d'apprentissage, nous obtenons différents scores de prédictions. En ne prenant pas en compte l'équilibre entre les classes dans la base de données, les arbres de décisions et le modèle de plus proche voisin donnent de bons scores de prédictions globaux mais des prédictions par classe non homogènes. Un équilibrage de la base de donnée nous permet par la suite d'obtenir des scores de prédiction par classe supérieur à 80% en utilisant un arbre de décision, ce qui répond à notre objectif.