Simon Aubé (111 158 058)
Nassim El Massaudi (536 916 232)
Alexandre Moreau (111 225 281)

Analyse et traitement de données massives
GLO-7027

Rapport 2

Travail présenté à
M. Richard Khoury

Département d'informatique et de génie logiciel
Université Laval
Hiver 2022

Prédiction des intentions de vote de l'Étude Électorale Canadienne 2019: Rapport 2

Simon Aubé[‡*], Nassim El Massaudi[‡*], Alexandre Moreau[‡*]
[‡] Université Laval
[*] Ces auteurs ont contribué également

Abstract

De nombreux facteurs influencent les préférences des électeurs au cours d'une campagne électorale, et ceux-ci ne sont pas pleinement compris. La prédiction d'intentions de vote n'est donc guère aisée. De vastes ensembles de données combinant les intentions de vote à des attributs idéologiques et socio-économiques pour un grand nombre d'individus peuvent contribuer à élucider les déterminants du vote. Les résultats de l'Étude électorale canadienne de 2019 représentent justement un tel ensemble de données. Nous proposons donc de les utiliser afin d'élucider les relations qui unissent une variété d'attributs aux intentions de vote des Canadiens et, ensuite, de prédire ces dernières. Dans ce second rapport, nous développons et optimisons des arbres de décision ainsi que des classificateurs par forêt aléatoire afin de réaliser de telles prédictions. Nous explorons également des approches pour remplir les données manquantes et combattre le déséquilibre des classes, puis présentons la performance du meilleur modèle obtenu. Celui-ci est prometteur, puisque la majorité de ses prédictions sont justes, peu importe l'intention de vote prédite.

1. Introduction

Lors d'élections démocratiques, les choix des citoyens ont de multiples causes. Les électeurs votent évidemment selon leurs préférences et leurs valeurs, mais les positions idéologiques relatives des partis les uns par rapport aux autres [1] de même que des considérations stratégiques – visant notamment l'expression d'un vote ayant un plus grand impact [2] – entrent aussi en jeu. En conséquence, la prédiction des intentions de vote de groupes ou d'individus n'est pas nécessairement aisée. Celle-ci peut pourtant être très utile, entre autres pour les partis politiques en cours de campagne, et est donc un sujet de recherche fréquent. Les prévisions électorales se basent habituellement sur des sondages, mais ceux-ci présentent souvent des biais dus à un échantillonnage non-représentatif [3]. Le développement de nouveaux outils ne se basant pas sur l'auto-déclaration des intentions de vote – par exemple en ayant recours à des indicateurs socio-économiques [4] – est donc souhaitable. Les déterminants des intentions de vote ne sont cependant pas pleinement élucidés, ce qui complique cette tâche.

Afin de mieux comprendre les différents facteurs qui guident les choix des électeurs, l'élection fédérale canadienne de 2019 constitue un système modèle tout indiqué. Celle-ci a en effet mis en scène des partis représentant une large gamme de positions idéologiques, tandis que le système électoral canadien peut favoriser le vote stratégique. Un vaste ensemble de données sur cette campagne électorale est d'ailleurs disponible, grâce à l'Étude électorale canadienne [5]. Cette dernière contient des informations récoltées auprès de 37 822 citoyens et résidents permanents canadiens, qui concernent autant leurs intentions de vote et leur perception des partis et figures politiques que leurs positions idéologiques ainsi que leur statut socio-économique. Elle se divise en deux parties: une enquête préélectorale effectuée auprès de la totalité des répondants entre le 13 septembre et 21 octobre 2019 (jour de l'élection) ainsi qu'une enquête postélectorale réalisée du 24 octobre au 11 novembre 2019 auprès de 10337 des participants de départ. Au total, ces deux volets ont fourni 600 attributs

se rapportant aux différents répondants, bien que plusieurs soient plutôt des métadonnées sur le sondage et qu'aucun individu n'ait répondu à la totalité des questions.

Un aussi vaste ensemble de données représente une opportunité exceptionnelle d'en apprendre davantage sur les déterminants des intentions de vote au sein d'une population, et c'est justement notre objectif. Ayant d'abord identifié les attributs les plus informatifs sur les intentions de votes lors de travaux préliminaires présentés dans un rapport précédent, nous validons maintenant ces conclusions par la prédiction d'un sous-ensemble d'intentions de vote qui ont préalablement été masquées dans l'ensemble de données. À cette fin, nous développons et optimisons incrémentalement des classificateurs par arbre de décision ainsi que par forêt aléatoire, tout en identifiant une meilleure méthode pour procéder au remplissage des données manquantes et en explorant des approches de sur- et de sous-échantillonnage pour lutter contre le déséquilibre des classes. Le meilleur modèle obtenu est une forêt aléatoire entraînée sur 35 attributs et offrant une exactitude globale de 79%, dont nous détaillons et discutons la performance.

2. Méthodes

2.1. Prétraitement des données

Seuls les attributs les plus informatifs d'après nos analyses préliminaires ont été conservés (Rapport 1). Il s'agit de $cps19\_imp\_iss\_party$, $cps19\_fed\_id$, $cps19\_vote\_2015$, $cps19\_prov\_id$, $cps19\_fed\_gov\_sat$, $cps19\_issue\_handle\_8$, $cps19\_issue\_handle\_2$, $cps19\_ethnicity\_38$, $cps19\_language\_69$, $cps19\_party\_member\_38$, $cps19\_interest\_gen\_1$, $cps19\_interest\_elxn_1$ et $cps19\_age$ ainsi que des séries $cps19\_lead\_int$, $cps19\_lead\_strong$, $cps19\_lead\_trust$, $cps19\_lead\_cares$, $cps19\_party\_rating$, $cps19\_lead\_rating$ et $cps19\_cand\_rating$. Tous les attributs catégoriques ainsi sélectionnés ont été ré-encodés sous forme de vecteurs *one-hot*, puis la totalité des attributs numériques ont été normalisés par méthode min-max. On note que, lors de la conversion en vecteurs *one-hot*, les données manquantes sont ignorées: l'absence d'une réponse est encodée comme un 0 dans chaque catégorie. Ceci n'est pas entièrement faux, puisque plusieurs données manquantes dénotent une absence de réponse probablement associée à une opinion moins tranchée, mais est certainement très approximatif.

Afin de pouvoir associer une classe au plus grand nombre de répondants possibles, les 5 variantes de l'intention de vote ont en plus été combinées en un attribut composite d'intention de vote correspondant à leur somme (puisqu'au plus une des cinq réponses est donnée par chaque répondant).

Les répondants dont nous devons prédire les intentions de votes ont ensuite été retirés du jeu de données afin d'être traités séparément. Les rares répondants additionnels pour lesquels l'intention de vote était manquante ont aussi été retirés, et ignorés lors de la suite des travaux. Puis, les répondants restants ont été répartis aléatoirement en deux sous-ensembles, respectivement d'entraînement et de validation, représentant 75% et 25% des données.

2.2. Entraînement et optimisation d'arbres de décision

La première tentative de prédiction des intentions de vote a été faite au moyen d'arbres de décision, d'après leur implémentation dans la librairie de scikit-learn [6]. Afin d'identifier les attributs les plus utiles, 25 modèles ont été construits et entraînés successivement par l'ajout graduel d'attributs (*Tableau 1*). À chaque étape, les attributs considérés ont été sélectionnés dans le sous-ensemble d'entraînement défini ci-haut, puis seuls les répondants ne présentant aucune donnée manquante dans les colonnes correspondantes ont été conservés. L'entraînement a ensuite été fait sur l'ensemble résultant par validation croisée à dix plis, au

cours de laquelle trois métriques ont été calculées: l'exactitude globale, la moyenne macro du score F1 et l'exactitude équilibrée. Les paramètres par défaut du classificateur par arbre de décision de scikit-learn ont été utilisés, hormis pour la spécification d'un poids des classes *balanced* (inversement proportionnel à leur fréquence).

Suivant cette identification des attributs les plus utiles, une optimisation du meilleur modèle ainsi défini a été faite. De nouveaux entraînements ont été réalisés selon la même approche de validation croisée à dix plis pour un large éventail de profondeurs maximales permises à l'arbre de décision.

2.3. **Imputation des données manquantes**

Étant donné que le jeu de données contient beaucoup de NaNs pour les attributs choisis, nous avons décidé d'imputer les données manquantes. Nous avons identifié deux méthodes pour y arriver, soit imputation par KNN ou la méthode MICE. Étant donné la grande quantité de données à imputer (28% des individus contiennent des données manquantes), nous avons jugé que l'utilisation de méthodes statistiques simples comme par moyenne ou par médiane n'était pas adapté. Pour les deux méthodes, l'implémentation scikit-learn [6] a été utilisée.

L'imputation par KNN consiste à calculer la données manquantes selon la moyenne de la valeur de ses k plus proches voisins. Les plus proches voisins sont déterminés en fonction de la distance euclidienne les séparants de la valeurs à déterminer.

La méthode MICE (Multivariate Imputation By Chained Equations) est une méthode itérative pour l'imputation de données [7]. La méthode consiste à utiliser une régression linéaire pour chaque attribut pour lequel il manque des données. L'attribut sélectionné est la variable cible et les autres attributs sont utilisés comme prédicteurs.

Voici les étapes de l'algorithme MICE:

(1) Nous calculons les valeurs manquantes pour chaque attribut en calculant la moyenne de l'attribut.
(2) Nous passons à travers chaque attributs et calculons ses valeurs manquantes en fonction des autres attributs.
(3) Nous soustrayons la matrice d'attributs initiales de la matrice d'attributs finale. Ceci nous donne la différence de valeurs suite à l'application de la régression linéaire. Le but est d'arriver à 0 ou près de 0.
(4) Nous retournons à l'étape 2, en utilisant la matrice calculée à l'étape 2 précédente comme matrice initiale.

Nous répétons les étapes 2 à 4 un nombre défini de fois ou jusqu'à convergence.

Les deux tentatives d'imputation ont été réalisées suivant la même approche en trois temps. D'abord, le remplissage des données manquantes dans l'ensemble d'entraînement précédemment défini a été faite, entraînant au passage l'estimateur (méthode *fit_ transform*). Puis, cet estimateur calibré a été utilisé pour réaliser l'imputation dans l'ensemble de validation, sans possibilité d'entraînement supplémentaire (méthode *transform*). Finalement, l'ensemble mis de côté puisqu'il correspond aux prédictions demandées a été traité de la même manière (méthode *transform*).

Préalablement à cela, le meilleur hyperparamètre k – désignant le nombre de plus proches voisins utilisés – a dû être identifié pour le KNN. Pour ce faire, l'imputation a été répétée sur l'ensemble d'entraînement pour plusieurs k entre 5 et 500 (*Figure 6*), et un arbre de décision a été entraîné sur la totalité des attributs de l'ensemble de données résultant. Le remplissage des valeurs manquantes, l'entraînement de l'arbre et son évaluation ont été effectués par une validation croisée à dix plis, et ce, à l'intérieur d'un pipeline (tel que défini dans le module scikit-learn).

Pour comparer les deux méthodes de remplissage des données manquantes, l'arbre de décision optimisé obtenu précédemment a été entraîné séparément sur les deux ensembles d'entraînement imputés obtenus respectivement par MICE et par KNN. Puis, la performance de chaque arbre a été évaluée sur l'ensemble de validation imputé correspondant (MICE ou KNN selon le cas; obtenu par l'estimateur calibré sur le jeu d'entraînement). Le raisonnement derrière cette approche était qu'une meilleure imputation des données manquantes devrait être plus généralisable de l'ensemble d'entraînement vers celui de test, et donc permettre à un modèle (arbre de décision de ce cas) entraîné sur le premier de mieux performer sur le second. Puisqu'il y a une certaine stochasticité dans l'entraînement des arbres de décision, cette démarche a été répétée dix fois avec des *seeds* aléatoires différentes.

2.4. **Entraînement et optimisation de forêts aléatoires**

Un classificateur par forêt aléatoire, tel qu'implémenté dans le module scikit-learn [6] a d'abord été entraîné sur les données d'entraînement imputées par MICE à partir des meilleurs attributs précédemment identifiés lors de l'entraînement d'arbres de décision. Afin de réduire l'impact du déséquilibre des classes, la forêt aléatoire a été modifiée pour attribuer des poids distincts à chaque classe à l'intérieur de chaque arbre, de façon inversement proportionnelle à sa fréquence dans le sous-ensemble de données considéré. Les principaux hyperparamètres du classificateur ont été optimisés par une recherche en grille du nombre d'estimateurs et de la profondeur maximale des arbres. Pour chaque combinaison de valeurs, une validation croisée à quatre plis a été faite sur l'ensemble d'entraînement. La meilleure combinaison a été identifiée d'après la moyenne macro du score F1, la macro-précision et l'exactitude globale.

Afin de garantir l'optimalité du modèle obtenu, nous avons ensuite tenté de le complexifier par l'ajout successif d'attributs potentiellement utiles. Ce sont 11 modèles supplémentaires de complexité croissante qui ont ainsi été construits (*Tableau 2*). Chacun de ces modèles a été entraîné sur l'ensemble d'entraînement obtenu par MICE par validation croisée à quatre plis, et ce, pour quelques profondeurs maximales différentes. La performance de chaque classificateur a ensuite été comparée au modèle précédemment optimisé (seulement en termes d'hyperparamètres), lorsque soumis au même régime de validation croisée.

Une fois la forêt aléatoire véritablement optimale identifiée, sa performance a été évaluée sur l'ensemble de validation obtenu par MICE. Pour ce faire, le modèle a d'abord été entraîné de nouveau sur la totalité de l'ensemble d'entraînement, avant d'être testé sur cet ensemble de validation. Les matrices de confusion ont été construites et des métriques de performance (précision par classe et moyenne macro, rappel par classe et moyenne macro, précision par classe et moyenne macro ainsi que exactitude globale) ont été calculées. L'importance de Gini (basée sur l'impureté) des différents attributs a aussi été extraite directement par la propriété correspondante de l'instance calibrée du classificateur.

2.5. **Corrections par sur- et sous-échantillonnage**

Étant donné le grand débalancement de classes dans le jeu de données, nous avons étudié trois possibilités pour remédier au problème. Chaque méthode a ses avantages et inconvénients.

La première méthode est le sous-échantillonnage. C'est la méthode la plus simple, qui consiste seulement à retirer des individus dans les classes ayant plus d'individus que la classe la moins bien représentée. Ceci mène donc à un balancement de classe parfait. Cependant, cela implique de perdre beaucoup de données qui contiennent de l'information importante pour le modèle. Avec les attributs choisis, nous gardon seulement 217 individus par classe, ce qui nous donne 1953 individus totaux comparé au jeu de données non balancé qui contient 37822 individus.

La deuxième méthode est le sur-échantillonnage. La méthode consiste à inventer de nouvelles données basées sur les distributions des attributs pour mieux représenter les petites classes. Pour cela, nous utilisons une variante de l'algorithme SMOTE, SMOTENC (Synthetic Minority Over-sampling Technique for Nominal and Continuous features) qui prend aussi en compte les variables catégoriques.

L'algorithme SMOTENC est simple. Un point est choisi au hasard dans la classe. Ensuite, on sélectionne les k plus proche voisins de ce point. On prend un des voisins au hasard.

Pour les variables continues, on trace une ligne imaginaire entre le point sélectionné et le voisin sélectionné. Pour créer des nouveaux individus, nous prenons des points au hasard sur la ligne tracée. Pour les variables catégoriques, nous prenons la moyenne de l'attribut pour les k voisins sélectionnés.

Grâce à cette technique, nous pouvons générer un très grand nombre de nouveau points. La technique SMOTENC a cependant un problème marqué dans notre cas. Étant donné le grand débalancement des classes et le petit nombre d'attributs, il arrive que les mêmes points soient générés pour plusieurs classes. Ceci cause une instabilité dans le modèle car une même combinaison d'attributs peut représenter plusieurs classes.

La troisième technique est simplement un mélange des deux techniques précédentes. Nous pouvons trouver un point milieu entre la classe la plus représentée et la moins représentée. Nous réduisons les classes plus représentées à ce nombre et augmentons les autres classes avec SMOTENC. Ceci nous donne 43155 individus dans le jeu de données.

Ces trois méthodes ont été testées sur le meilleur modèle de forêt aléatoire obtenu précédemment. Pour ce faire, l'ensemble d'entraînement imputé par MICE a été ajusté par sous-échantillonnage, sur-échantillonnage ou par l'approche mixte (selon le cas). Pour les deux dernières approches, l'implémentation de SMOTENC dans la module *Imbalanced learn* a été utilisée [8]. Puis, le nouvel ensemble obtenu a lui-même été divisé en un ensemble d'entraînement (75%) et un ensemble de test (25%). Le meilleur classificateur par forêt aléatoire a été entraîné sur les nouvelles données d'entraînement ainsi obtenues, puis évalué sur les données de test ajustées correspondantes. Finalement, une seconde validation a été effectuée sur l'ensemble de validation défini lors de l'imputation par MICE. Celui-ci n'ayant pas été traité par sous- ou sur-échantillonnage, ce test supplémentaire a permis d'évaluer si un modèle entraîné sur des données artificiellement balancées était bien généralisable aux vraies données d'intentions de vote, qui sont débalancées.

3. Résultats et discussion

3.1. Un modèle préliminaire permet d'identifier les meilleurs attributs

Nous avons d'abord cherché à identifier un sous-ensemble d'attributs optimal pour la prédiction des intentions de vote, et ce, au moyen d'arbres de décision. Ceux-ci constituent en effet un modèle simple qui a intuitivement un fort potentiel pour la tâche considérée. Leur structure en arborescence est tout indiquée dans une telle situation où la valeur d'un attribut peut limiter les valeurs possibles pour d'autres attributs. Par exemple, un électeur qui dit se reconnaître davantage dans le Parti conservateur aura très peu de chances de voter pour le NPD, qui adopte des positions nettement plus à gauche.

Étant donné la popularité très variable des partis politiques et le fort déséquilibre des classes qui en résulte, nous avons ajusté le modèle de l'arbre de décision afin que chaque classe se voit attribuer un poids inversement proportionnel à sa fréquence. Ceci devrait faciliter l'apprentissage des classes les moins fréquentes, qui est habituellement plus difficile puisque celles-ci n'ont qu'un très petit impact absolu sur la performance de l'arbre sans l'ajout de tels poids. Nos travaux préliminaires, présentés dans le Rapport 1, nous avaient permis de mettre en évidence plusieurs attributs fortement corrélés aux intentions de vote. Nous avons donc pris pour point de départ un sous-ensemble minimal de ces attributs et l'avons complexifié

graduellement par l'ajout successif d'autres attributs potentiellement significatifs, de façon à construire 25 modèles (*Tableau 1*). Chacun de ces modèles a ensuite été évalué par validation croisée à dix plis. Cette démarche a révélé que, pour cet ensemble de données, la performance d'un arbre de décision plafonne très rapidement au fil de l'ajout d'attributs (*Figure 1*) et que, tant du point de vue de l'exactitude globale que de celui du score F1 macro, le gain en performance est très faible par rapport à la combinaison parcimonieuse du meilleur attribut catégorique et de la meilleur série d'attributs numériques (Modèle 1). Ces travaux mettent aussi en évidence que, malgré l'ajustement des poids, l'apprentissage des classes les moins fréquentes est plus ardu pour le modèle, tel que l'indique les scores F1 macro nettement inférieurs à l'exactitude globale (cumulative). Vu la très faible amélioration au fil du développement de modèles plus complexes, nous avons retenu le modèle 14 – à mi-chemin en termes de nombre d'attributs – pour la suite.

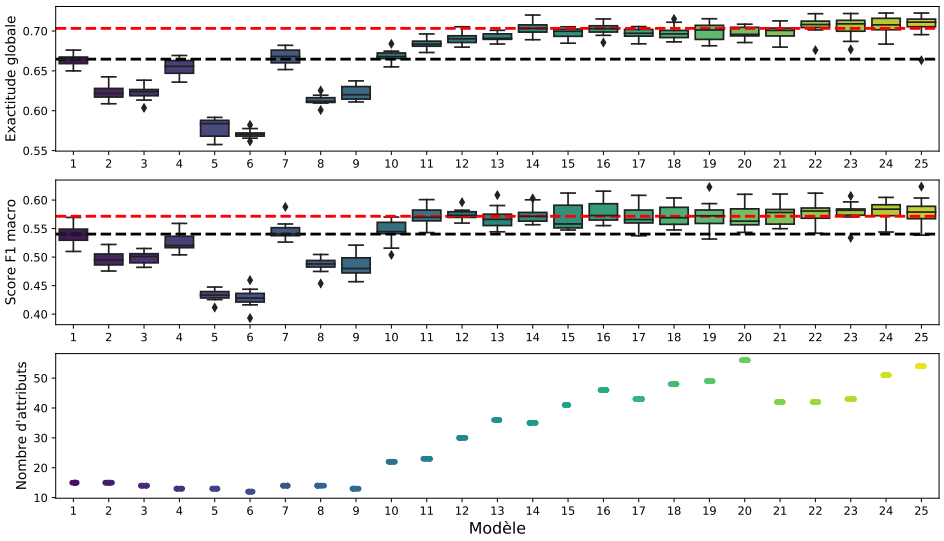

Figure 1. **L'ajout d'attributs améliore relativement peu la performance d'un arbre de décision pour la prédiction des intentions de vote.** L'exactitude globale (haut) et la moyenne macro du score F1 (milieu) ont été obtenues par validation croisée à dix plis pour 25 arbres de décision construits par l'ajout successif d'attributs potentiellement significatif. La ligne pointillée noire correspond à la performance du premier modèle utilisant les meilleurs attributs individuels, tandis que les pointillés rouges marque la performance du meilleur modèle choisi (le 14). Le nombre d'attributs croit graduellement (bas). Chacun des vecteurs *one-hot* obtenus à partir d'un attribut catégorique est compté individuellement, expliquant les augmentations parfois soudaines.

Le premier modèle ainsi obtenu présentait déjà une performance intéressante, mais celle-ci pouvait clairement être améliorée. Nous avons donc ensuite cherché à optimiser l'arbre en faisant varier la profondeur maximale permise lors de l'entraînement. Ceci est pertinent, puisqu'un arbre trop peu profond sera incapable d'apprendre convenablement chacune des classes montrées, tandis qu'au contraire, un arbre trop profond sera sujet au surapprentissage et sera peu généralisable en dehors de l'ensemble utilisé pour son entraînement. Ces nouveaux entraînements ont encore une fois été réalisés par validation croisée à dix plis. Conformément à nos attentes, cette approche a révélé une baisse de performance aux extrêmes de profondeur (*Figure 5*). Une profondeur excessive (100 noeuds) n'est cependant associée qu'à une faible baisse de la performance, suggérant que les possibilités de surapprentissage sont relativement limitées à partir de l'ensemble restreint d'attributs que nous

avons utilisé. Puisque l'exactitude globale et la moyenne macro du score F1 semblent tous deux atteindre un optimum pour une profondeur maximale de 14 (*Figure 5*), nous avons conservé ce choix de paramètre pour la suite. Vu le déséquilibre des classes, il est essentiel de ne pas se fier uniquement à l'exactitude globale – dominée par les partis les plus populaires – afin d'évaluer la performance des arbres. Autrement, le modèle sélectionné risquerait d'être seulement le meilleur pour identifier les partis principaux (Libéral et Conservateur), plutôt que celui qui reconnaît le mieux chacune des différentes classes. Le score F1 macro, qui fait la moyenne d'une combinaison égale du rappel et de la précision calculée pour chaque classe, vient limiter ce risque en accordant une valeur égale à la performance sur chacun des partis. En identifiant la profondeur maximale optimale, ces entraînements supplémentaires ont aussi révélé que l'optimisation des hyperparamètres de l'arbre de décision permettait relativement peu de gains en performance. Par rapport à notre modèle précédent, seul le score F1 macro a été amélioré substantiellement, et ce, de l'ordre de 0.02.

3.2. **Un remplissage des données manquantes par MICE améliore la performance**

Puisque les classificateurs du module scikit-learn [6] ne prennent pas en charge les valeurs manquantes, celles-ci doivent absolument être remplies pour que la prédiction des intentions de vote demandées soit possible. Nous avons donc procédé à l'imputation des valeurs manquantes, qui a en plus l'avantage de rendre possible l'entraînement de modèles sur un plus grand nombre d'individus – favorisant potentiellement une meilleur performance.

Nous avons comparé deux approches, respectivement par plus proches voisins (KNN) et par *Multivariate Imputation By Chained Equations* (MICE; Azur, Stuart, Frangakis, and Leaf [7]). Pour ce faire, nous avons entraîné le meilleur arbre obtenu précédemment sur l'ensemble d'entraînement obtenu par chacune des deux méthodes, puis avons évaluer sa performance sur l'ensemble de validation correspondant (Méthodes). Puisque l'algorithme d'imputation n'a lui-même été calibré que sur l'ensemble d'entraînement, nous avons postulé que le meilleur remplissage des données manquantes capturerait mieux les caractéristiques de l'ensemble de données et permettrait une meilleure généralisation à l'ensemble de validation pour un classificateur entraîné sur les données d'entraînement.

Après avoir entraîné l'arbre de décision dix fois avec l'imputation par KNN et dix fois avec l'imputation par MICE, nous avons obtenu les résultats à la figure 2.

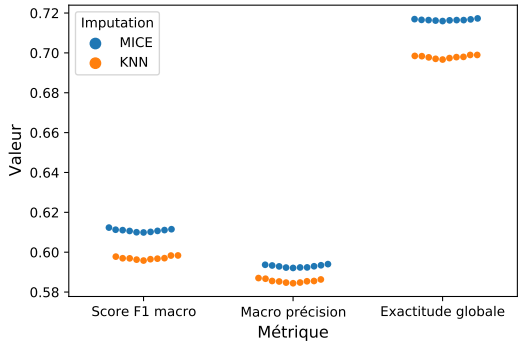

Figure 2. **L'imputation par MICE donne des données de meilleure qualité qu'une approche par KNN.** Pour l'une et l'autre des approches, le meilleur arbre de décision identifié précédemment a été entraîné sur un ensemble de données (75%) sur lequel la calibration des estimateurs et l'imputation des données avaient été réalisées simultanément (*fit-transform*), puis évalué sur un second ensemble (25%) ayant seulement été imputé à l'aide des estimateurs déjà calibrés (*transform*). Pour dix entraînements effectués avec différentes *seeds* aléatoires, le MICE performe mieux pour les trois métriques présentées.

Pour toutes les métriques identifiées, l'imputation par MICE donne une meilleure performance que l'imputation par KNN, et ce, pour chacun des dix réplicats d'entraînement. L'approche MICE semble donc plus généralisable, et c'est la méthode d'imputation que nous utiliserons pour la suite.

3.3. **Les forêts aléatoires permettent une meilleure prédiction**

Afin d'améliorer la performance de notre modèle, nous nous sommes tournés vers une approche apparentée à l'arbre de décision mais plus puissante: la forêt aléatoire (random forest). Celle-ci correspond en effet à un ensemble d'arbres chacun entraînés sur des sous-ensembles d'attributs et d'objets, et donc à même d'apprendre des distinctions plus fines entre les classes. Étant donné l'important déséquilibre des classes entre les partis politiques, l'algorithme de forêt aléatoire a été modifié afin d'accorder un poids différent à chaque classe, de façon inversement proportionnel à la fréquence de chacune. Ces poids ont été calculés individuellement pour chaque arbre, selon la fréquence des partis dans le sous-ensemble de données assigné aléatoirement à chacun.

Nous avons d'abord repris les attributs identifiés lors de l'entraînement du meilleur arbre de décision et utilisé ces derniers afin d'entraîner un classificateur par forêt aléatoire dans le but d'identifier les meilleurs hyperparamètres. Un balayage en grille du nombre d'estimateurs et de la profondeur maximale des arbres a été réalisé par validation croisée à quatre plis sur l'ensemble d'entraînement imputé par MICE. Cette démarche a révélé un impact relativement limité de ces deux hyperparamètres, en dehors des extrêmes (*Figure 7*). Dans l'optique d'optimiser la performance de notre modèle, celle-ci nous a conduits à retenir une forêt de 400 estimateurs ayant une profondeur maximale de 35 noeuds pour la suite, puisque cette combinaison maximise à la fois l'exactitude globale et la macro-précision. On note toutefois que cela se fait au coût d'une diminution du rappel, comme l'indique la baisse (de petite magnitude) de la moyenne macro du score F1. Étant donné que l'augmentation du nombre d'estimateurs et de la profondeur maximale semblait associé à une augmentation de la macro-précision et de l'exactitude globale, nous avons également entraîné une forêt aléatoire en spécifiant une combinaison extrême de ces deux paramètres: 5000 estimateurs d'une profondeur maximale de 150 noeuds. La validation croisée a cependant révélé une performance guère supérieure à ce qui était obtenu à l'intérieur de l'espace de paramètres précédemment balayé (F1= 0.676, macro-précision = 0.741 et exactitude globale = 0.787.)

Ayant déterminé la meilleure combinaison d'hyperparamètres pour le classificateur par forêt aléatoire, nous avons voulu optimiser encore davantage celui-ci. Raisonnant qu'un tel modèle était plus en mesure de tirer profit d'un grand nombre de dimensions qu'un arbre de décision, nous avons tenté d'ajouter incrémentalement des attributs, tout en augmentant quelque peu la profondeur maximale afin que ces derniers puissent être exploités. Un nouveau balayage en grille a été fait par validation croisée à quatre plis sur l'ensemble d'entraînement imputé par MICE. Celui a néanmoins révélé très peu de variations en termes de performance (*Figure 8*). Il semble donc que le classificateur optimal obtenu à la suite du balayage précédent avait déjà atteint la plus grande performance possible pour une forêt aléatoire à partir des attributs individuellement fortement corrélés aux intentions de vote. Nous avons donc conservé ce modèle pour la suite des travaux.

Afin de mieux connaître ses forces et ses faiblesses, nous avons évalué sa performance sur le sous-ensemble de validation, sur lequel l'estimateur MICE n'a pas été calibré lors du remplissage des données manquantes. Le modèle a été entraîné une nouvelle fois sur la totalité de l'ensemble d'entraînement, puis il a été testé sur l'ensemble de validation. La comparaison des intentions de votes ainsi prédites et des intentions de votes véritables révèle une performance intéressante. Pour les quatre principaux partis (Libéral, Conservateur, NPD et Bloc Québécois), ce sont en effet plus de 80% des électeurs qui sont correctement

identifiés (*Figure 3 A.*). Ceci est cependant très variable entre les partis – jusqu'à un extrême d'à peine 10% pour les répondants préférant une formation "Autre". La moyenne macro du rappel s'élève donc seulement à 0.65, alors que le rappel globale est de l'ordre de 0.79. Malgré des variations de performance d'une classe à l'autre, notre classificateur fait généralement des prédictions correctes, globalement dans 79% des cas. Cette exactitude intéressante s'explique notamment par sa meilleure précision auprès des classes les plus fréquentes, dont le parti conservateur (*Figure 3 B.*). On note que, malgré des variations qui font descendre la macro-précision à 0.73, plus de la moitié des prédictions faites sont correctes pour chaque classe. Ainsi, bien que le classificateur peine grandement à identifier tous les électeurs d'autres partis, près de 60% de ceux qu'il identifie comme tels le sont réellement. Dans l'ensemble, la principale faiblesse de ce modèle est son faible rappel des classes les moins abondantes (*Figure 3 A.*), comme l'illustre d'ailleurs la moyenne macro de son score F1 de 0.67, qui est nettement inférieure à la macro-précision.

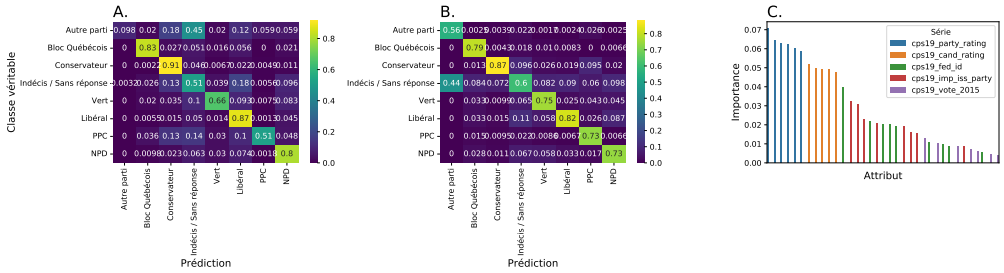

Figure 3. **La meilleure forêt aléatoire offre une bonne capacité prédictive. A.** Matrice de confusion normalisée selon la fréquence de chacune des classes. **B.** Matrice de confusion normalisée selon le nombre de prédictions. **C.** Importance des attributs.

La performance réduite du modèle pour les partis les moins populaires est fort probablement causée au moins en partie par le déséquilibre des classes, malgré l'ajustement aux poids de l'algorithme qui a été fait afin de contrecarrer ce phénomène. Même si une plus grande pondération est accordée à l'apprentissage des classes minoritaires, celui-ci sera toujours plus ardu, puisqu'il est réalisé à partir d'un moins bien grand nombre d'individus. Il est ainsi nettement plus difficile pour le modèle de découvrir des règles de décision généralisables, d'autant plus que certaines associations potentiellement utiles et significatives risquent de ne pas être représentées dans l'ensemble de données d'entraînement. Le déséquilibre dans le nombre d'instances disponibles pour chaque classe est d'ailleurs particulièrement marqué dans le cas présent. Tandis que plus de 6500 répondants sont disponibles dans l'ensemble d'entraînement pour les partis libéral et conservateur (respectivement 7001 et 6640 individus), le même ensemble ne compte en effet que 166 électeurs ayant signifié leur intention de voter pour un parti "Autre".

Pour mieux comprendre d'où provient la performance de ce modèle, il est aussi pertinent de s'intéresser à l'importance des différents attributs considérés. Ceci permet de constater une décroissance graduelle de celle-ci (*Figure 3 C.*), suggérant que l'ensemble des attributs utilisés – incluant les vecteurs *one-hot* soient potentiellement utiles. On note cependant que certains vecteurs sont associés à une importance très petite, inférieure à 0.01. Les attributs issus des mêmes séries (même question demandée au sujet de chaque partie ou vecteurs *one-hot* issus d'un attribut catégorique) sont généralement groupés les uns avec les autres dans l'ordre d'importance, suggérant que l'utilité de l'information dépend davantage de la question demandée que du parti considéré. Il faut toutefois noter que l'importance des

attributs des séries *cps*19*_party_rating* et *cps*19*_cand_rating* sont probablement surévaluées du fait de leur grand nombre de valeurs uniques, puisqu'il s'agit d'attributs numériques continus.

3.4. **Le sur- et le sous-échantillonnage ne résolvent pas le déséquilibre des classes**

Étant donné que les faiblesses du modèle développé précédemment semblent être en bonne partie dues au déséquilibre des classes, nous avons tenté de corriger celui-ci en procédant au sous-échantillonnage et au sur-échantillonnage (SMOTENC; Lemaître, Nogueira, and Aridas [8]) des individus, tant séparément qu'en combinaison.

Nous avons testé les trois méthodes, et aucune d'entre elles n'a amélioré significativement les résultats du modèle de base. Le rappel a certes pu être amélioré de quelques points de pourcentage, mais cela s'est fait au prix d'une diminution de la macro-précision et de l'exactitude globale. Ceci pourrait être en partie causé par le petit nombre d'attributs et le nombre limité de valeurs possibles pour chacun d'eux, de sorte que peu d'individus soient nécessaire pour avoir une exploration complète de l'espace du problème. Ainsi, l'ajout de nouvelles données n'apporte aucune information supplémentaire. Dans le cas du sous-échantillonnage, c'est cependant l'inverse: cette approche retire trop de données et une exploration complète de l'espace du problème n'est plus possible, car la classe la moins représentée a très peu d'individus.

Pour le sur-échantillonnage par SMOTENC ainsi que l'approche mixte, des résultats préliminaires suggéraient une amélioration intéressante, permettant notamment un rappel presque parfait pour trois classes – dont la catégorie "Autre" si difficile à prédire – avec cette dernière méthode (*Figure 4*). Cette plus grande performance était obtenue lors de tests sur des données soumises à la même approche de rebalancement artificiel. Une validation subséquente sur l'ensemble de validation issue de l'imputation MICE – non ajusté et donc débalancé – a toutefois montré que cette amélioration n'était pas généralisable (*Figure 4*) et s'apparentait donc à du surapprentissage. Cela est très probablement dû au fait que nous avons appliqué l'algorithme SMOTENC après la transformation des attributs catégoriques en vecteurs *one-hot*. Dans ce cas-ci, les données peuvent être faussées, puisque SMOTENC peut inventer une instance possédant une valeur pour plusieurs vecteurs issus du même attribut catégorique initial. Or, il est impossible, par exemple, que quelqu'un ait voté pour deux partis à l'élection de 2015. Dans un travail futur, il serait intéressant de corriger ce problème et d'évaluer si SMOTENC peut améliorer la performance dans cette situation.

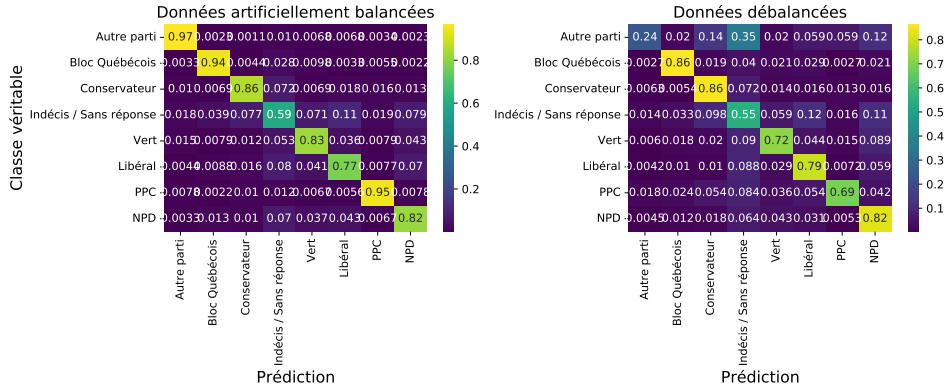

Figure 4. **Le balancement des classes par méthode mixte n'améliore pas les résultats comparé au modèle précédent.** Comparaison des résultats de validation obtenues sur des données balancées par l'approche mixte (gauche) et sur des données de validation débalancées (droite).

3.5. **La plupart des prédictions du modèle final sont correctes**

Étant donné l'échec du recours aux approches de sur- et de sous-échantillonnage (voir section précédente), notre meilleur modèle est bien le classificateur par forêt aléatoire optimal obtenu précédemment. Si sa performance est loin d'être parfaite, elle est tout de même fort intéressante. Tel que cela a été montré plus haut, ce modèle est en effet en mesure d'identifier correctement environ 80% des électeurs des principaux partis, et même plus de la moitié des électeurs des partis mineurs (*Figure 3 A.*). De plus, la majorité de ses prédictions sont aussi correctes – quelle que soit la classe qui est prédite –, même s'il y a une importante variation d'un parti à l'autre (*Figure 3 B.*).

Ces difficultés rencontrées par le modèle lors de la prédiction des classes plus rares s'expliquent entre autres par la faible fréquence de celles-ci ainsi que par l'hétérogénéité des répondants qui y correspondent. Cela est d'ailleurs particulièrement évident chez les électeurs comptant voter pour un parti "Autre" et n'ayant pas été correctement identifiés comme tels. Chez ce groupe, on constate plusieurs combinaisons rares d'attributs, pour lesquelles l'apprentissage d'une règle de décision spécifique sera donc rarement favorisée. Une telle combinaison s'observe notamment chez le répondant 8178, qui a une approbation maximale à la fois du parti conservateur et du NPD et a erronément été attribué au parti conservateur. Puisque ces deux formations politiques sont presque diamétralement opposées idéologiquement, très peu d'électeurs sont susceptibles de les approuver toutes les deux. Ce répondant 8178 montre d'ailleurs très bien comment quelques réponses contradictoires peuvent fausser la classification malgré la présence de toute l'information nécessaire. Celui-ci a en effet répondu qu'il a voté pour un autre parti en 2015 et qu'il ne s'identifie à aucun parti: deux caractéristiques fortement associées à une intention de vote "Autre". En plus de présenter des combinaisons rares de valeurs, les répondants "Autre" sont en plus très peu cohérents les uns avec les autres. Par exemple, l'individu 29650, faussement classifié en tant que libéral, n'approuve pratiquement pas les conservateurs (7%), et appartient malgré tout à la même classe que l'électeur 8178.

L'exactitude réduite dans la classification des partisans des formations mineures peut aussi parfois s'expliquer partiellement par la proximité idéologique avec d'autres partis. C'est notamment ce qu'on observe chez le PPC, qui présente des similitudes avec le parti conservateur. Plusieurs électeurs du PPC s'identifient en effet à ce parti majeur et sont ainsi faussement classifiés comme des conservateurs. C'est entre autres le cas des répondants 7773, 6173 et 15818. Ceux-ci ont en plus la particularité d'avoir une perception très positive (65% de leur candidat conservateur local, ce qui rend leur classification exacte encore plus ardue.

La meilleure performance du modèle sur les classes plus fréquentes s'explique quant à elle par la plus grande popularité des partis correspondants, mais encore davantage, par l'homogénéité de leurs électeurs. Chez le parti conservateur, ce sont en effet 81%, 62% et 70% des électeurs qui s'identifient au parti, disent avoir voté pour lui en 2015 et jugent qu'il traite le mieux de l'enjeu qui leur tient le plus à coeur, respectivement. Chez les répondants ayant correctement été classifiés comme conservateurs – par exemple les individus 31454 et 19751 –, ces pourcentages grimpent d'ailleurs à $87,5\%$, 76% et 67%, respectivement.

3.6. **Rétrospective**

À première vue, le problème semblait assez simple. Nous avions trouvé quelques attributs bien corrélés avec la variable cible et nous nous étions dit que c'était dans la poche.

Lors de nos premiers tests, nous avons réalisés que les attributs que nous avions trouvés, même si très corrélés individuellement avec la variable cible, ne contiennent pas toute l'information nécessaire pour obtenir un score parfait. Cependant, les attributs que nous avions identifiés permettent tout de même d'obtenir un score respectable.

Le débalancement de classes est très important, et nous aurions dû mettre encore plus d'emphase sur ce problème. Il y a des classes "fourre-tout", soit les classes "I don't know" et "Another party" pour lesquelles des analyses plus fines aurait pu être faites afin d'identifier certaines corrélations entre attributs qui aideraient à distinguer les indécis du reste des individus.

De plus, nous n'avons pas considéré la dimension temporelle des données, mais la performance sous-optimale de notre modèle nous amène à envisager une telle amélioration. Les votes passés ainsi que l'identification à un parti sont en effet plus figés dans le temps que les intentions de vote, qui dépendent des tendances de même que de considérations stratégiques influencés par les intentions des autres électeurs. Des données de sondage – idéalement au niveau de chaque circonscription – aurait donc potentiellement pu contribuer à raffiner nos prédictions, en identifiant par exemple les électeurs susceptibles voter pour le parti libéral malgré leur préférence pour le NPD.

Comme mentionné plus haut dans le rapport, si c'était à refaire, nous modifierions certainement SMOTENC pour qu'il soit appliqué avant la transformation des attributs en vecteurs *one-hot*, afin que cette approche soit implémentée correctement et soit ainsi plus susceptible d'améliorer les résultats.

References

[1] L. Ezrow and G. Xezonakis. "Citizen Satisfaction With Democracy and Parties' Policy Offerings". In: *Comparative Political Studies* 44.9 (2011), pp. 1152–1178. DOI: 10.1177/0010414011405461.

[2] A. Gallego, G. Rico, and E. Anduiza. "Disproportionality and voter turnout in new and old democracies". In: *Electoral Studies* 31.1 (2012). Special Symposium: Germany's Federal Election September 2009, pp. 159–169. DOI: https://doi.org/10.1016/j.electstud.2011.10.004.

[3] W. Jennings and C. Wlezien. "Election polling errors across time and space". en. In: *Nature Human Behaviour* 2.4 (Apr. 2018), pp. 276–283. DOI: 10.1038/s41562-018-0315-6.

[4] P. Hummel and D. Rothschild. "Fundamental models for forecasting elections at the state level". In: *Electoral Studies* 35 (2014), pp. 123–139. DOI: https://doi.org/10.1016/j.electstud.2014.05.002.

[5] L. B. Stephenson, A. Harell, D. Rubenson, and P. J. Loewen. *2019 Canadian Election Study - Online Survey*. 2020. DOI: https://doi.org/10.7910/DVN/DUS88V.

[6] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[7] M. J. Azur, E. A. Stuart, C. Frangakis, and P. J. Leaf. "Multiple imputation by chained equations: what is it and how does it work?" eng. In: *International journal of methods in psychiatric research* 20.1 (Mar. 2011). Publisher: John Wiley & Sons, Ltd, pp. 40–49. ISSN: 1557-0657. DOI: 10.1002/mpr.329. URL: https://pubmed.ncbi.nlm.nih.gov/21499542.

[8] G. Lemaître, F. Nogueira, and C. K. Aridas. "Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning". In: *Journal of Machine Learning Research* 18.17 (2017), pp. 1–5. URL: http://jmlr.org/papers/v18/16-365.html.

Appendix A. Optimisation des arbres de décision

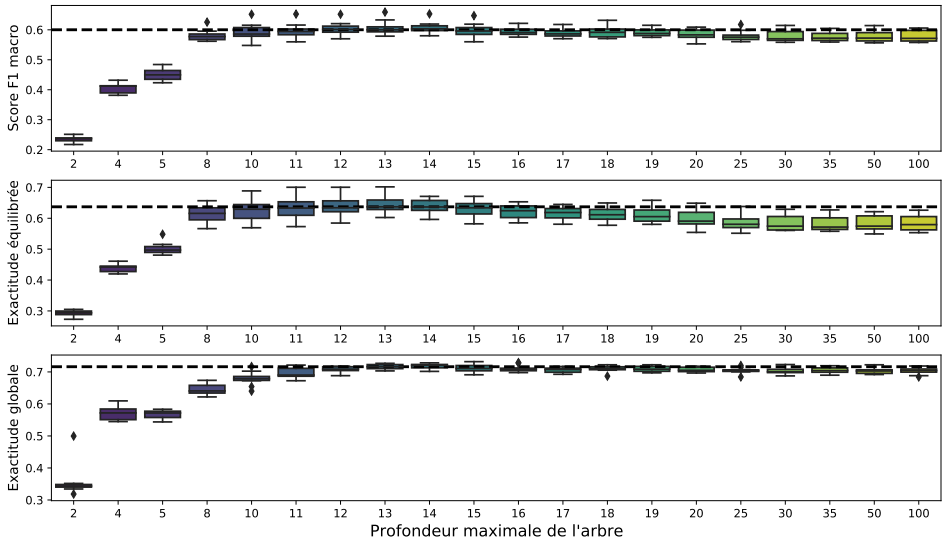

Figure 5. **En dehors des extrêmes, la profondeur maximale permise à l'arbre de décision a un impact limité sur sa performance.** Moyenne macro du score F1 (haut), exactitude équilibrée (milieu) et exactitude globale (bas) pour différentes profondeurs maximales d'arbres de décision entraînés sur les meilleurs attributs identifiés précédemment (modèle 14 de la *Figure 1*) par validation croisée à dix plis. La ligne pointillée noire correspond à la médiane obtenue pour la profondeur optimale identifiée (14 noeuds).

Table 1. Ensembles d'attributs pour les arbres de décision

Modèle	Attributs
Modèle 1	cps19_fed_id, cps19_party_rating
Modèle 2	cps19_fed_id, cps19_lead_rating
Modèle 3	cps19_fed_id, cps19_cand_rating
Modèle 4	cps19_vote_2015, cps19_party_rating
Modèle 5	cps19_vote_2015, cps19_lead_rating
Modèle 6	cps19_vote_2015, cps19_cand_rating
Modèle 7	cps19_imp_iss_party, cps19_party_rating
Modèle 8	cps19_imp_iss_party, cps19_lead_rating
Modèle 9	cps19_imp_iss_party, cps19_cand_rating
Modèle 10	cps19_fed_id_party, cps19_vote_2015, cps19_party_rating
Modèle 11	cps19_fed_id_party, cps19_imp_iss_party, cps19_party_rating
Modèle 12	cps19_fed_id_party, cps19_imp_iss_party, cps19_vote_2015, cps19_party_rating
Modèle 13	Modèle 12 + cps19_lead_rating
Modèle 14	Modèle 12 + cps19_cand_rating
Modèle 15	Modèle 14 + cps19_lead_rating
Modèle 16	Modèle 15 + cps19_fed_gov_sat
Modèle 17	Modèle 15 + cps19_ethnicity_38, cps19_language_69
Modèle 18	Modèle 15 + cps19_fed_gov_sat, cps19_ethnicity_38, cps19_language_69
Modèle 19	Modèle 15 + cps19_lead_strong
Modèle 20	Modèle 15 + cps19_fed_gov_sat, cps19_ethnicity_38, cps19_language_69, cps19_lead_strong
Modèle 21	Modèle 15 + cps19_age
Modèle 22	Modèle 15 + cps19_interest_elxn_1
Modèle 23	Modèle 15 + cps19_age, cps19_interest_elxn_1
Modèle 24	Modèle 15 + cps19_age, cps19_interest_elxn_1, cps19_lead_strong
Modèle 25	Modèle 23 + cps19_ethnicity_38, cps19_language_69, cps19_lead_trust, cps19_party_member_38

Appendix B. Optimisation de l'imputation par KNN

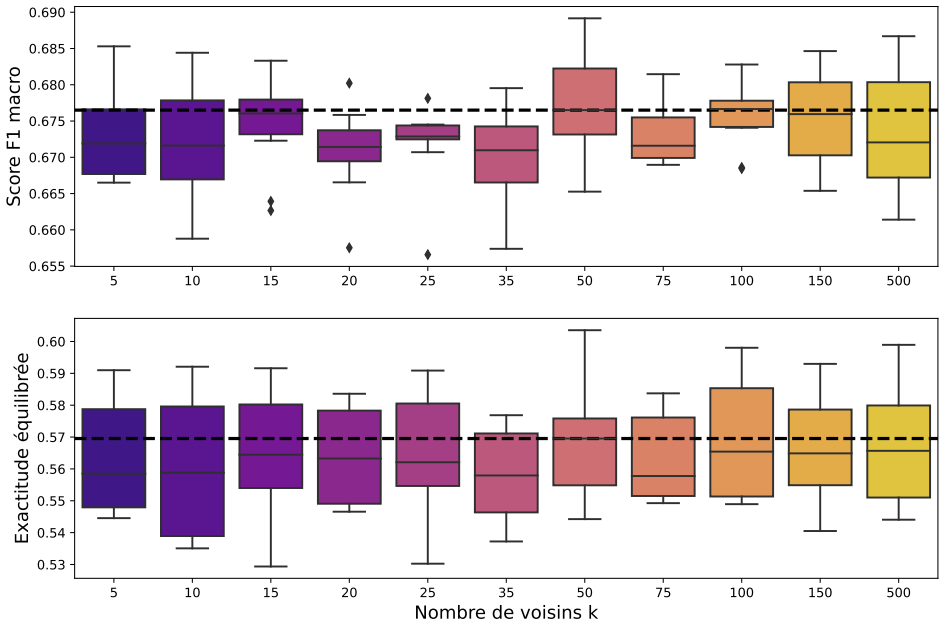

Figure 6. **Le nombre de voisins considérés lors de l'imputation par KNN a peu d'impact sur la qualité des données.** Moyenne macro du score F1 (haut) et exactitude équilibrée (bas) pour un arbre de décision entraîné par validation croisée à dix plis sur l'ensemble des attributs considérés lors de l'imputation des données manquantes. La ligne pointillée noire désigne la performance médiane pour la méthode d'imputation retenue ($k = 50$).

Appendix C. Optimisation des classificateurs par forêt aléatoire

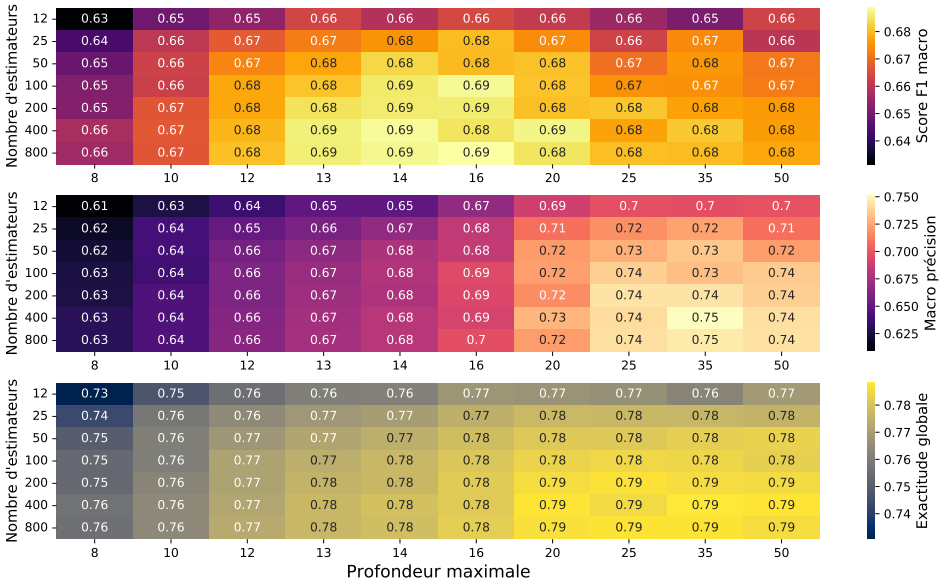

Figure 7. **Optimisation du nombre d'estimateurs et de la profondeur maximale de la forêt aléatoire**. La moyenne macro du score F1 (haut), la macro précision (milieu) et l'exactitude globale (bas) ont été calculées suivant l'entraînement de modèles définis avec différentes combinaisons d'hyperparamètres. Chacun des entraînement a été réalisé par validation croisée à quatre plis sur l'ensemble de données d'entraînement imputées par MICE.

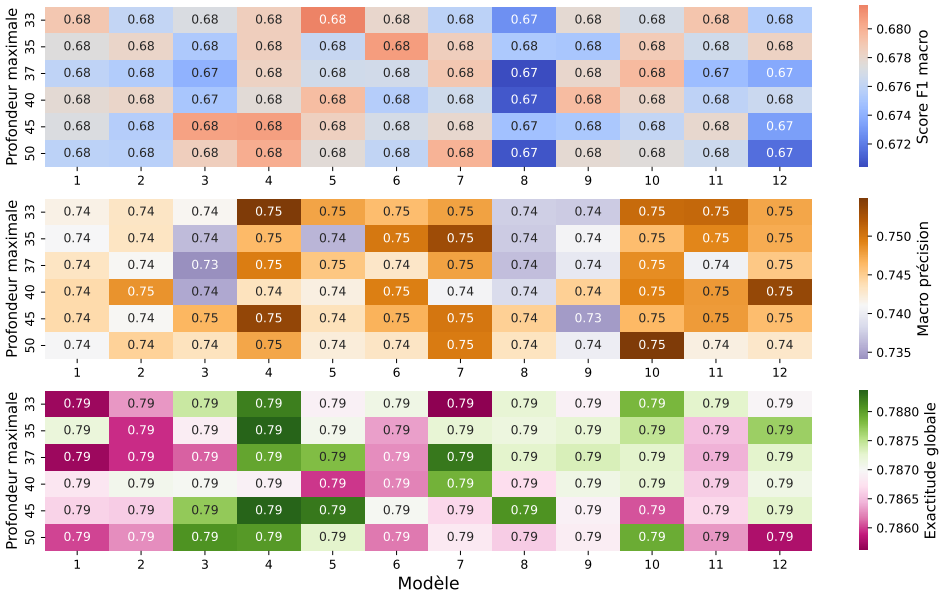

Figure 8. **L'ajout d'attributs et des variations subséquentes de la profondeur maximale ont très peu d'influence sur la performance du classificateur par forêt aléatoire.** La moyenne macro du score F1 (haut), la macro précision (milieu) et l'exactitude globale (bas) ont été calculées suivant l'entraînement de modèles définis avec différentes combinaisons d'attributs et de profondeurs maximales. Chacun des entraînement a été réalisé par validation croisée à quatre plis sur l'ensemble de données d'entraînement imputées par MICE.

Table 2. Ensembles d'attributs pour les forêts aléatoires

Modèle	Attributs
Modèle 1	Meilleur modèle DecisionTree (Modèle 14 dans *Tableau 1*)
Modèle 2	Modèle 1 + cps19_fed_gov_sat
Modèle 3	Modèle 1 + cps19_party_member_38
Modèle 4	Modèle 1 + cps19_age, cps19_interest_elxn_1
Modèle 5	Modèle 1 + cps19_ethnicity_38, cps19_language_69
Modèle 6	Modèle 1 + cps19_issue_handle_2
Modèle 7	Modèle 1 + cps19_issue_handle_8
Modèle 8	Modèle 1 + cps19_lead_rating
Modèle 9	Modèle 1 + cps19_lead_cares
Modèle 10	Modèle 2 + cps19_age, cps19_interest_elxn_1
Modèle 11	Modèle 1 + cps19_issue_handle_2, cps19_issue_handle_8
Modèle 12	Modèle 10 + cps19_issue_handle_2, cps19_issue_handle_8