Projet en analyse et traitement de données massives : Traitement des données et résultats

Etienne Candat, Lucie Jean-Labadye, Alice Nollet
Soumis le 13 avril 2022

Abstract

Dans le cadre du cours "Analyse et traitement de données massives" de la session d'hiver 2022, nous devions conduire un projet ayant pour objectif de prédire le vote d'un individu en s'appuyant sur ses réponses à un sondage mené par le Consortium de la démocratie électorale avant et après l'élection fédérale de 2019. Cette enquête, menée auprès de plus de 37,000 électeurs canadiens, comprenait des questions sur leurs positions poltiques, sociales ou économiques, mais aussi des indicateurs plus factuels (âge, CSP, croyances religieuses, etc.)

Ce deuxième rapport fait état de la façon dont nous avons traité les données, ainsi que des entraînements, des tests et des réflexions que ceux-ci ont fait naître tout au long de leur mise en oeuvre.

1. Introduction

Lors de la première phase du projet, dédiée à la prise en main du jeu de données et de sa documentation, nous avions développé plusieurs axes de réflexion quant au prétraitement des attributs. Nous avions commencé par interroger la chronologie du sondage et la pertinence de séparer les données entre attributs "CPS" (campaign period survey) et "PES" (post-election survey), puis nous nous étions concentrés sur la documentation, en prenant note des réponses qui nous semblaient "intuitivement" intéressantes. Nous avions alors remarqué que ces dernières pouvaient être réparties en deux grands groupes : les attributs "d'opinion" (réponses aux question du type "Pensez-vous que...") et les attributs "factuels" (données descriptives des répondants), et que lors de notre premier tri, nous nous étions concentrés sur les attributs relevant de la première catégorie, en particulier sur ceux dont les modalités de réponses étaient identiques ou presque au format final de la prédiction (i.e. une liste de partis). Nous avions alors mené quelques tests afin d'évaluer la qualité de ces données, et nous nous sommes heurtés au critère d'intégralité, plus de la moitié des réponses étant manquantes. Nous avions alors décidé de changer de prisme, en ne se concentrant non plus sur les données qui nous semblaient pertinentes, mais sur celle présentant une grande qualité. Nous avions pris comme point de départ la liste fournie par la documentation des 6 attributs qui avaient servi à identifer les individus ayant répondu plusieurs fois au sondage[1], et ce pour deux raisons : premièrement car ces attributs avaient été considérés par le C-DEM comme suffisamment "existentiels" pour distinguer les répondants entre eux, mais aussi et surtout car les matrices de contingence entre chacun de ces attributs (à l'exception de `cps19_bornin_canada`) et la variable `cps19_votechoice` soulignaient leur signifiance statistique.

C'es avec cette base que nous nous sommes lancés dans la deuxième phase du projet dont nous allons rendre compte dans les sections suivantes.

[1] cf. Liste A.1 en annexe

2. Choix des algorithmes de traitement et des métriques d'évalutaion

Au vu de la configuration du jeu de données et des modalités de la prédiction attendue, il nous a semblé naturel de considérer ce projet par le prisme de la classification multiclasse. Nous nous sommes donc tournés vers les deux algorithmes vus en classe et considérés comme ayant les meilleures performances : les classificateurs naïfs de Bayes et les forêts aléatoires. Nous avons alors découvert que la librairie SciKit-Learn proposait plusieurs variantes du premier, et décidé de privilégier le classificateur catégorique (`sklearn.naive_bayes.CategoricalNB()`), la grande majorité des variables du jeu de données relevant de ce type. A la différence d'un classificateur non-catégorique, qui estime entre autres des paramètres tels que la moyenne ou la variance de chaque variable afin d'en déterminer la vraisemblance (en supposant par exemple que cette dernière suit une distribution normale pour le classificateur `GaussianNB()`), la variante catégorique se concentre sur les fréquences relatives d'apparition de chaque tuple (données, classe) et en déduit des probabilités d'appartenance qui seront ensuite utilisées pour classifier de nouvelles données. Si la classification par forêt aléatoire (`sklearn.ensemble.RandomFore- stClassifier()`) ne l'exploite pas de la même façon, elle s'appuie elle aussi sur la probabilité, calculée à partir de la fréquence, qu'un tuple de données appartiennent à une certaine classe : à chaque noeud, cette probabilité est utilisée pour choisir l'attribut le plus discrimiant (sélection par gain d'information) ou celui réduisant au mieux l'impureté (sélection par index de Gini). De ce fait, ce type de classification est adapté aux variables catégoriques, et n'a pas demandé de travail d'ajustement particulier.

Une fois les algorithmes choisis et ajustés, nous nous sommes dans un deuxième temps concentrés sur les métriques, afin de pouvoir commencer les entraînements en étant capables de les évaluer dès le début. La librairie SciKit-Learn dispose d'un module `metrics` permettant de calculer toutes les métriques vues en cours. Nous en avons décidé de conserver en priorité la matrice de confusion, dans la mesure où c'est l'indicateur le plus factuel, dont toutes les autres métriques découlent. Nous avons aussi choisi de garder l'exactitude afin d'évaluer notre positionnement par rapport à une classification aléatoire, mais nous l'avons doublée d'une exactitude équilibrée (`balanced_accuracy`) :

$$\text{BA} = \frac{\text{Recall}_1 + ... + \text{Recall}_n}{n} \quad \text{avec } n \text{ le nombre de classes} \tag{2.1}$$

Puisque cette métrique s'appuie sur des mesures réalisées classe par classe, son calcul, fait à l'échelle macro, permet de contrer l'effet que peuvent avoir les classes majoritaires présentes dans notre jeu de données sur le calcul de l'exactitude non-balancée. Nous avons aussi choisi de garder le rappel et la précision afin d'évaluer les performances des algorithmes sur les données observées et prédites. Enfin, nous avons implémenté le calcul du score F1, métrique particulièrement importante pour les jeux de données débalancés car elle permet de mettre en perspective l'exactitude et de signaler lorsque les bons scores de cette dernière sont atteints en privilégiant les classes dominantes.

Afin de pouvoir calculer ces métriques de façon effective, nous avons orienté nos réflexions autour du design général de l'entraînement et de son déroulé. Nous avons alors opté pour une validation croisée à k échantillons, dans la mesure où cette méthode permet d'éviter le surapprentissage en alternant les échantillons utilisés pour l'entraînement et ceux utilisés pour le test. Nous avons instinctivement posé k=10, puis avons par la suite décidé de conserver cette valeur, dans la mesure où elle représentait un bon compromis entre le temps de calcul (qui est d'environ une minute, mais qui se voit plus que doublé si on multiplie k par deux) et la nécessité d'échantillonner un certain nombre de fois notre jeu de données pour éviter l'overfitting. En suivant l'exemple présent dans la documentation de la fonction `sklearn.model_selection.KFold()`, nous avons implémenté une boucle `for` au cours de laquelle chaque classificateur est ré-instancié et toutes les métriques mentionnées ci-dessus

sont affichées pour chaque nouvel échantillon, et à la fin de laquelle sont affichés les minimums, maximums et moyennes des deux exactitudes que nous avont choisies de conserver et ceux du score F1, ainsi que le temps de calcul global. Ce dernier va d'ailleurs se révéler plus long pour la classification par forêt aléatoire que pour le classificateur naïf de Bayes, ce qui n'est pas nécessairement une surprise dans la mesure où la première instancie un certain nombre de fois l'algorithme glouton qu'est l'arbre de décision.

3. **Tests et entraînements**

Après avoir mis en place toutes les composantes nécessaires à l'apprentissage et à son évaluation, nous sommes revenus au jeu de données afin de le préparer. En effet, si les algorithmes de traitement que nous avons choisis sont capables de s'entraîner avec des données catégorielles, ils ne peuvent pas les manipuler sous leur forme brute. Il a donc fallu trouver certaines méthodes pour adapter les données aux modèles, puis d'autres pour augmenter les performances de ces derniers.

3.1. **1ère itération : exploitation de la base d'attributs constituée lors de la première phase du projet**

Comme mentionné en introduction, notre base de départ comprenait cinq attributs[2], qui avaient déjà fait l'objet d'un prétraitement lors de la première phase du projet. Cette démarche étant toutefois restée rudimentaire, dans la mesure où elle avait été menée en vue d'obtenir une idée globale des corrélations qui existaient entre les attributs et la variable cible, nous avons décidé de commencer par l'affiner, en essayant de trouver le *sweet spot* entre minimisation de la perte d'information et nécessaire réduction de la dimensionnalité. Si nous avions commencé par grouper les répondants en 3 classes d'âge, nous avons jugé lors de cette phase que la perte d'information découlant de cette division était trop importante, et avons choisi de subdiviser chacun des classes en 2 sous-classes, en se basant sur leur médiane respective. Nous avons fait le choix de cette statistique car nous savions que l'âge avait été un critère du C-DEM pour constituer l'échantillon des répondants, et nous ne voulions déséquilibrer les classes ainsi établies. Nous avons par ailleurs approfondi nos recherches quant aux grands systèmes religieux et fusionné le double tri (un premier basé sur les courants religieux, et un deuxième sur les branches au sein de chacun d'entre eux) que nous avions mis en place lors de la première phase en créant une nouvelle étiquette "extrémismes religieux", qui est venue s'ajouter au 5 déjà existantes (religions orientales, judaïsme, islam, christianisme et personnes non-religieuses). Ici s'est peut-être glissé un biais, dans la mesure où nous avons essayé de rester les plus objectifs possible, mais où il en va aussi du système de valeurs de chacun de juger si une croyance relève d'un extrémisme ou pas. Afin d'essayer de contrer cet effet, nous sommes restés prudents avec cette nouvelle étiquette et avons minimisé le nombre modalités auquel nous l'avons apposée. Nous sommes ainsi parvenus à réduire de 23-6 = 17 le nombre de dimensions de cette variable. Pour ce qui est des attributs concernant le statut d'emploi et le niveau d'éducation, nous avons jugé que le prétraitement initialement effectué induisait une trop grande perte d'information (il y a par exemple de nombreuses raisons pour lesquelles certaines personne ne travaillent pas ou plus, et il est quasiment impossible de fusionner les modalités du niveau d'éducation dans la mesure où on ne sait pas précisément ce que signifie "Some..."), et avons par conséquent décidé de ne fusionner que les modalités "Don't know / Prefer not to answer" et "Other (please specify)", gardant le reste tel que présenté dans le jeu de données original. Enfin, l'attribut décrivant le genre ne contenant que 3 modalités, nous l'avons lui aussi laissé tel quel. Nous avons aussi mené un prétraitement sur la variable cible car, après relecture des consignes,

[2]cf ListeA.1 en annexe

nous nous sommes rendu compte qu'elle se partageait les colonnes `cps19_votechoice`, `cps19_votechoice_pr`,`cps19_vote_unlikely`,`cps19_vote_unlike_pr` et `cps19_v_advance`. Nous avons donc choisi de remplacer les valeurs manquantes par une chaîne de caractère vide et de concaténer toutes ces colonnes en une, que nous avons nommée "label".

Nous avons ensuite discrétisé les variables explicatives catégorielles (toutes à l'exception de l'âge) à l'aide de la fonction `OrdinalEncoder()` du module `preprocessing` offert par SciKit-Learn et avons mené notre premier entraînement. Comme nous pouvions nous y attendre au vu du peu d'attributs sélectionnés, les métriques étaient loin d'être convaincantes, mais dépassaient déjà celles d'un classificteur aléatoire.

Résultats pour le classificateur naïf de Bayes :

Métrique	Min	Max	Moyenne
Exactitude	0.322	0.345	0.334
Exactitude balancée	0.163	0.181	0.173
Score F1	0.144	0.164	0.154

Résultats pour la classification par forêt aléatoire :

Métrique	Min	Max	Moyenne
Exactitude	0.303	0.327	0.313
Exactitude balancée	0.153	0.172	0.164
Score F1	0.145	0.166	0.156

3.2. **2ème itération : ajout d'attributs**

Afin d'enrichir la base de données et d'ainsi augmenter les performances de nos entraînements, nous avons alors décidé de faire un grand pas en arrière en revenant aux attributs que nous avions "intuitivement" considérés pertinents au tout début de la première phase du projet. Ce retour s'est toutefois fait en adoptant un prisme beaucoup plus efficient qu'au moment de la première lecture de la documentation : nous avons tout de suite interrogé le nombre de valeurs manquantes de ces attributs, afin de se concentrer sur ceux réquérant le moins de prétraitement possible. Nous avons alors eu la surprise de constater que parmi la liste que nous avions établie, trois de ces variables[3] se distinguaient par leur intégralité: `cps19_prov_id` contenait quasiment 99,9% de données disponibles, `cps19_vote_2015` en contenait 79.2% et `cps19_fed_id` était intégralement remplie. De plus, ces attributs présentaient l'avantage de partager la forme de notre variable cible `label`. Afin de remplacer les données manquantes de `cps19_prov_id` et `cps19_vote_2015`, nous avons entraîné un classificateur naïf de Bayes "intermédiaire" à l'aide des 5 attributs de la 1ère itération et de `cps19_fed_id`, avons discrétisé les résultats obtenus à l'aide de l'`OrdinalEncoder()` et avons lancé une deuxième boucle d'entraînement, dont les résultats se sont révélés beaucoup plus satisfaisants, l'exactitude se trouvant doublée et l'exactitude équilibrée et le score F1 étant plus que triplés.

Résultats pour le classificateur naïf de Bayes :

Métrique	Min	Max	Moyenne
Exactitude	0.666	0.706	0.686
Exactitude balancée	0.522	0.594	0.552
Score F1	0.53	0.593	0.561

Résultats pour la classification par forêt aléatoire :

[3]cf. Liste A.2 en annexe

Métrique	Min	Max	Moyenne
Exactitude	0.644	0.675	0.66
Exactitude balancée	0.489	0.563	0.522
Score F1	0.506	0.574	0.536

3.3. **3ème itération : nouvel ajout d'attributs**

En élargissant la démarche de trouver des attributs présentant peu de valeurs manquantes aux variables que nous n'avions pas nécessairement retenues lors de nos affinages successifs, nous nous sommes rendus compte que les réponses aux questions siglées `cps19_spend_...` [4] étaient elles aussi intégralement présentes. Nous avons alors mené un test du χ^2 et avons obtenu des valeurs telles que $0 \leq p \leq 1.15\text{E} - 198 << \alpha = 0.05$, qui nous ont amenés à conclure que ces attributs étaient très corrélés à la variable cible, et que nous pouvions les ajouter à notre jeu de données, ainsi qu'aux attributs participant à la classification des valeurs manquantes de `cps19_prov_id` et `cps19_vote_2015`. Nous avons aussi porté notre attention sur les attributs siglés `cps19_lead_...`[5], qui présentaient eux aussi une forte corrélation avec la variable cible, et dont la forme permettait un prétraitement rapide : nous avons décidé de regrouper toutes les questions, séparées par modalités de réponse dans le jeu original, par grandes familles (`lead_cares`, `lead_trust`, etc.) et de les encoder en adaptant les principes de la fonction indicatrice[6]. Ainsi, chaque candidat qu'un répondant aura trouvé intéressant est encodé par un 1, et le reste est encodé par des 0, l'avantage de cette méthode résidant dans le fait qu'elle permet de ne pas se poser la question des valeurs manquantes. Nous avons alors lancé une troisième boucle d'entraînement, qui a débouché à son tour sur une augmentation des performances de nos algorithmes, cette fois-ci de quelques points seulement.

Résultats pour le classificateur naïf de Bayes :

Métrique	Min	Max	Moyenne
Exactitude	0.676	0.699	0.686
Exactitude balancée	0.583	0.623	0.597
Score F1	0.56	0.581	0.569

Résultats pour la classification par forêt aléatoire :

Métrique	Min	Max	Moyenne
Exactitude	0.708	0.741	0.718
Exactitude balancée	0.541	0.587	0.557
Score F1	0.565	0.614	0.58

Il nous semble intéressant de faire remarquer que le classificateur naïf de Bayes, qui réalisait jusqu'alors de meilleures performances que la classification par forêt aléatoire, se fait ici dépasser par cette dernière sur l'exactitude et le score F1.

3.4. **4ème itération : derniers essais et version finale**

Pour cette dernière itération, nous avons dans un premier temps poursuivi la démarche de privilégier les attributs n'ayant que peu de valeurs manquantes et en avons trouvé une petite dizaine [7] que nous avons prétraités et/ou encodés (la variable `cps19_income_cat` a par exemple nécessité un prétraitement particulier, dans la mesure où toutes ses valeurs manquantes sauf 3 trouvaient leur équivalence au sein de l'attribut `cps19_income_number`. Nous avons donc catégorisé toutes les réponses fournies à cette dernière question en fonction

[4] cf. Liste A.3 en annexe
[5] idem
[6] $\chi_F : E \to \{0, 1\}$ telle que $\chi_F(x) = 1$ si $x \in F$, 0 sinon.
[7] cf Liste A.4 en annexes

des modalités de `cps19_income_cat`, et avons attribué aux 3 répondant n'ayant ni chiffre ni catégorie celle où se trouve le revenu médian canadien, i.e. de 30,001$ à 60,000$). Nous les avons ensuite ajoutés un par un à la listes des attributs exploités pour l'apprentissage de `cps19_prov_id` et `cps19_vote_2015` et pour l'entraînement de nos algorithmes de traitement, mais les performances n'ont pas significativement augmenté. Une hypothèse que nous pouvons ici avancer est celle d'une corrélation à l'intérieur des variables explicatives, ces dix derniers attributs décrivant généralement des situations de vie associées à d'autres variables précédemment prises en compte (le statut matrimonial, le fait d'avoir des enfants ou non, le fait de vivre dans une province très urbaine, qui concentre principalement les métiers du tertiaire, etc. peut tout à fait être lié à l'âge et/ou au statut d'emploi, par exemple).

Dans un second temps, nous avons décidé de nous détacher complètement des considérations liées au prétraitement des données pour se concentrer des attributs d'opinion dont la configuration semblait propice à ce que nous voulions essayer de mettre en place : un algorithme de clustering capable lui aussi de prédire les valeurs manquantes. Pour se faire, nous avons choisi la famille d'attributs `cps19_pos`[8], dont les modalités de réponses sont fixées et ordonnables (d'accord / pas d'accord) et pour lesquelles les données manquantes représentent en moyenne 2/3 des données disponibles. Nous voulions utiliser ces attributs afin de créer une nouvelle variable "position", capable de décrire le positionnement global de chaque répondant quant à ces sujets de société. Nous avons opté pour l'implémentation d'un algorithme de type BIRCH dans la mesure où c'est un algorithme agglomérant qui ne craint pas le grand nombre de données. Nous avons posé `n_clusters = 9` car nous voulions créer autant de partitions qu'il y a de modalités de la variable cible, et nous avons lancé l"algorithme. Si celui-ci a effectivement fonctionné, une nouvelle variable "position" étant créée, mais son évaluation était impossible, dans la mesure où il n'y avait aucun point de comparaison. Nous avons donc ajouté l'attribut "position" à notre système complet, et l'avons fait tourner une dernière fois.

Résultats pour le classificateur naïf de Bayes :

Métrique	Min	Max	Moyenne
Exactitude	0.640	0.712	0.684
Exactitude balancée	0.545	0.634	0.597
Score F1	0.536	0.585	0.563

Résultats pour la classification par forêt aléatoire :

Métrique	Min	Max	Moyenne
Exactitude	0.563	0.752	0.723
Exactitude balancée	0.529	0.574	0.553
Score F1	0.548	0.601	0.576

Cette dernière itération vient confirmer la performance de la classification par forêt aléatoire en ce qui concerne l'exacitude et le score F1, gain de performance pour lequel nous avancerons des hypothèses dans la partie suivante.

4. **Version finale**

La version finale de notre système s'articule donc autour des deux phases classiques d'un projet de traitement de données massives : le prétraitement et l'entraînement. Au cours de la première, les attributs n'ayant aucune valeur manquante sont recatégorisés dans la mesure du possible afin d'en réduire les dimensions sans perte d'information puis sont encodés numériquement, avant d'être exploités lors de prédictions intermédiaires servant à classifier les données manquantes des variables en présentant un nombre restreint (en particulier

[8]cf. Liste A.5 en annexe

`cps19_prov_id` et `cps19_vote_2015`). Un algorithme de partitionnement se charge alors, en s'appuyant sur les attributs de type `cps19_pos_`de distribuer les répondants dans 9 groupes en fonction de leurs réponses à des questions pour lesquelles les données manquantes sont majoritaires et dont la classification exacte n'est pas évidente. Une fois ces données réunies, le jeu créé[9] est soumis aux deux modèles de classification que nous avons choisis d'implémenter, et ce dans une boucle assurant la validation croisée à k échantillons. A l'issue de cette boucle, des statistiques générales sur les métriques sont affichées, et il y a écriture des prédictions obtenues dans le fichier `"prediction.txt"`.

Nous sommes globalement satisfaits de ce système, malgré le fait que ses performances ne soient pas exceptionnelles. Pour lui apporter des pistes d'amélioration, nous avons commencé par comparer la répartition des vote observés et celle des prédictions.

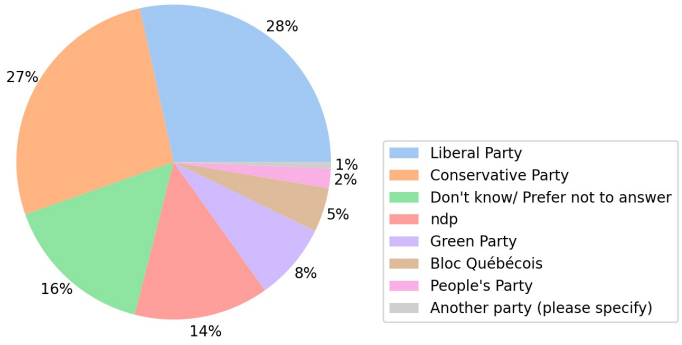


Figure 1. Répartition des votes observés

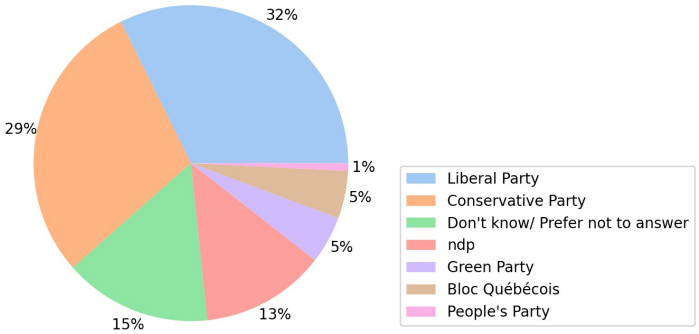
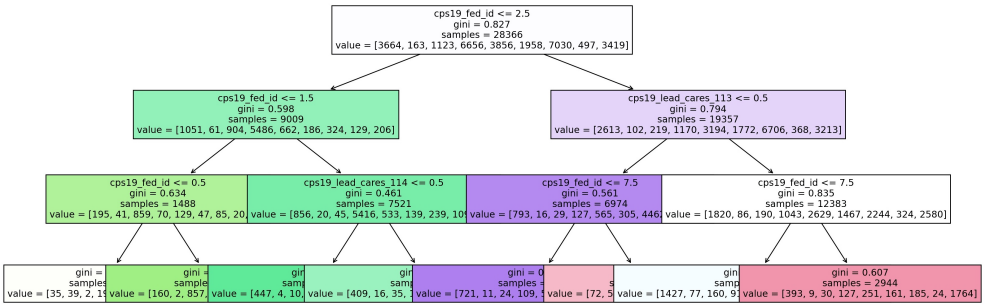


Figure 2. Répartition des votes prédits

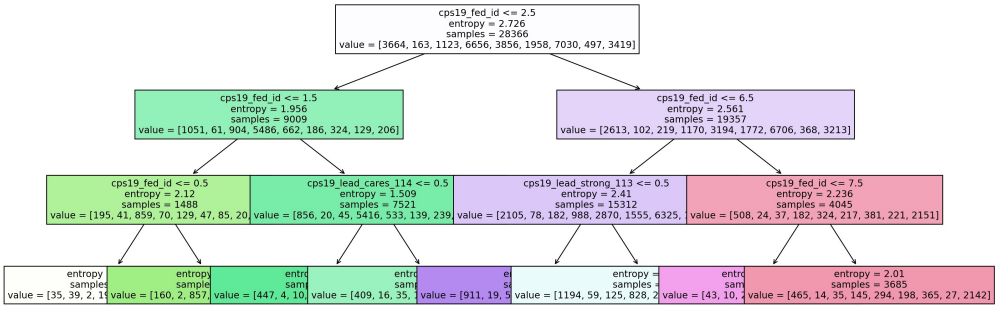
Sans surprise, il apparaît très clairement que notre système surévalue les classes majoritaire, particulièrement le parti libéral, grignotant ainsi la part des petits partis. Afin d'expliquer ce phénomène, nous avons commencé par conduire quelques tests comme celui du χ^2, mais nous nous sommes rapidement aperçus que cette démarche n'expliquerait que très partiellement les limites de notre modèle, les p-values produites étant toutes infimes et donc difficiles à comparer et à discriminer. Nous avons alors décidé de produire un arbre de décision unique, afin d'observer quels attributs sont proches de la racine, et donc jugés les plus discriminants et les plus réducteurs d'impureté.

[9]cf toutes les liste de l'annexe A pour la liste complète des attributs

Arbre de décision sur critère de Gini, max_depth = 3 :



Arbre de décision sur critère d'entropie, max_depth = 3 :



On remarque tout de suite que, quelque soit le critère, l'attribut `cps19_fed_id` est considéré comme le plus pertinent, au point d'apparaître plusieurs fois. Or, il s'avère que c'est un attribut pour lequel le pari Libéral est sur-représenté par rapport à la variable cible : la modalité "Liberal Party" représente 32% des réponses apportées, soit exactement le pourcentage des prédictions allant au parti Libéral faites par notre modèle. En outre, en descendant dans les feuilles, on constate que les attributs les plus significatifs après `cps19_fed_id` sont ceux relevant de la famille `cps19_lead_X`. Dans la mesure où Justin Trudeau était Premier Ministre depuis 4 ans au moment du sondage, on peut avancer deux hypothèses. Une première serait de dire qu'il est plus facile d'évaluer le leadership et les façons de gouvernener de quelqu'un qui détient effectivement le pouvoir, mais cette réflexion marche dans les deux sens, et aurait pu aussi être très pénalisante pour le parti Libéral. On pourrait supposer dans un second temps que même les Canadiens qui ne s'identifient pas comme Libéraux à l'échelle fédérale ont fini par apprécier la gouvernance de M. Trudeau et le rapport qu'il entretient avec les citoyens du Canada. On pourrait aussi postuler que nous accordons plus facilement du charisme et du pouvoir à quelqu'un qui le détient déjà, créant ainsi un cercle vertueux (ou vicieux?)

Quoiqu'il en soit, il existe plusieurs façons de remédier à ce déséquilibre : nous avons pensé à échantillonner, mais cette démarche ne peut relever que de l'*undersampling* dans la mesure où nous ne pouvons inventer des réponses à des électeurs de petits partis dont on vient artificiellement gonfler le nombre. Il nous faudrait donc tirer 217 personnes de chaque modalités de la classe `label`, (217 étant la plus petite fréquence, associée à la réponse "Another party (please specify)"), et créer un jeu de données à partir de $217 \times 8 = 1736$ répondants, soit moins de 5% du total, représentant potentiellement une grande perte d'information. Nous

arions aussi pu mettre en place un système de poids ou de pénalité, rendant par exemple la classification comme Libéral ou Conservateur très peu probable une fois que les propostions observées sont atteintes (cette démarche suppose de privilégier le classificateur naïf de Bayes dans la mesure ou l'algorithme de l'arbre de décision de backtracke pas). Une lecture plus approfondie des méta-données présentes dans la documentation, en particulier des informations concernant la qualité des données pourrait aussi fournir des pistes d'amélioration.

5. **Etudes de cas**

Outre le débalancement des données d'origine, qui a poussé les algorithmes à privilégier les classes majoritaires, d'autres facteurs doivent il nous semble être pris en compte pour expliquer qu'il n'est pas facile d'arriver à un excellent niveau de prédiction pour ce genre de projet : les facteurs humain et chronologique. Les démocraties électorales se cristallisent au cours de grands moments où tous les citoyens sont amenés à exprimer une opinion "facilement classifiable" et où des sondages et des enquêtes rigoureuses sont menés, mais ces moments ne sont pas suffisamment proches dans le temps pour réellement suivre l'évolution de la position politique d'un individu. De plus, les provinces ayant un rôle important à jouer, il est aussi tout à fait possible de préférer une politique locale classée à gauche de l'échiquier politique, et une politique fédérale classée à droite, ou inversement. Ainsi, nos modèles ont majoritairement réussi à converger pour les personnes dont les attributs ne laissent que peu de place au doute, comme pour le répondant se trouvant à l'index 38, qui a déclaré s'identifer comme libéral quelque soit l'échelle de l'élection et avoir voté pour le parti Libéral en 2015, et qui a choisi Justin Trudeau pour toutes les questions liées au leadership : il a été classé à raison comme votant pour le parti Libéral par nos modèles. A l'inverse, certains répondants ont exprimé vouloir voter pour un parti dont ils sont traditionnellement éloignés. C'est le cas du répondant se trouvant à l'index 2296, qui s'est identifié comme conservateur à l'échelle provinciale et fédérale et a indiqué avoir voté pour ce parti en 2015, et qui a donc été classé comme électeur du parti Conservateur en 2019, alors qu'il a en réalité répondu qu'il voterait pour le parti Libéral. Enfin, ce genre d'erreur de classification peut aussi se retrouver pour des individus dont les attributs laissent beaucoup de doute, et dont la convergence est loin d'être évidente. Le répondant se trouvant à l'index 76 a par exemple attribué à Justin Trudeau et Jagmeet Singh toutes les caractéristiques liées aux attributs `cps19_lead`, a exprimé avoir voté pour le parti Libéral en 2015 et s'identifier comme proche du Nouveau Parti Démocratique à l'échelle provinciale mais considérer que le parti progressiste-conservateur est plus proche de ses intérêts à l'échelle fédérale. Du fait de l'importance des variables liées au leadership, cet individu a été classé comme électeur du NPD, mais a pourtant annoncé vouloir voter pour les Conservateurs.

Un même individu peut donc exprimer une grande pluralité politique à un instant t, mais aussi dans le temps. Nous avions par exemple vu pendant la première phase du projet que l'âge était un critère important, dans la mesure où les 55 ans et plus avaient plus tendance à voter Conservateurs et, de façon plus générale, à s'orienter vers les deux grands partis traditionnels ; tandis que les 18-24 ans se situaient globalement plus à gauche de l'échiquier politique et dispersaient plus leurs voix. Les idées et les appartenances sont mouvantes, et ainsi difficiles à classifier parfaitement.

6. **Rétrospective et conclusion**

Ce projet était le premier de ce type pour tous les membres de l'équipe, et a par conséquent été très riche en enseignements.

D'un point de vue des connaissances, les allers-retours entre mise en oeuvre des algorithmes vus en classe et enseignements théoriques nous ont permis d'enrichir chacune de

ces deux parties : si les connaissances fournies par le cours nous ont permis de mettre en place plus rapidement et efficacement les différentes phases du projet et nous ont amenés à avoir une compréhension plus fine des résultats obtenus, l'application de ces dernières a aussi jeté un éclairage nouveau sur des concepts qui pouvaient paraître un peu abstraits. Nous prendrons ici l'exemple de la classifictaion et du partitionnement dont les différences théoriques et d'usage nous avaient été expliquées au début de la session, mais dont nous avons vraiment saisi les enjeux en les mettant en oeuvre. De la même façon, pouvoir visualiser un arbre de décision sur un ensemble de données connu et pleinement appréhendé et expérimenter les conclusions que l'on pouvait en tirer nous ont beaucoup éclairé sur le fonctionnement théorique de ce classificateur et de la classification par forêt aléatoire de façon générale. Enfin, il nous faut aussi mentionner les connaissances acquises sur les différents sites spécialisés, parcourus lorsque nous cherchions à savoir pourquoi telle chose fonctionne de cette façon, ou comment telle conclusion peut être tirée de tel test.

En outre, mener à bien ce projet nous a appris les réflexes pratiques à avoir pour de futures études. Nous nous sommes par exemple avant tout concentrés sur une lecture humaine de la documentation, en oubliant complètement de prendre en compte la réalité statistique des données que nous allions manipuler. Cette étape n'a pas été vaine, dans la mesure où elle nous a fourni une liste d'attributs avec lesquels nous sommes venus enrichir nos modèles, mais nous aurions pu commencer plus tôt à prendre en main le jeu de données si nous nous étions concentrés sur d'autres critères de sélection plus viables d'un point de vue traitement des données, comme l'intégralité, que nous avons questionnée dans un second temps mais qui s'est finalement révélée être un de nos critères principaux. Nous en avons tiré la conclusion que les données les plus "pertinentes" ne sont pas nécessairement les plus intéressantes à privilégier, et que la qualité des données est un critère extrêmement important, dans la mesure où des données de grande qualité peuvent tout à fait être utilisée pour prédire ou partitionner d'autres données, de qualité moindre mais ayant une plus grande valeur prédictive. De plus, l'écriture de ce rapport nous a appris la nécessité de tenir à jour des logs propres et régulièrement actualisés, permettant de retrouver des résultats en un coup d'oeil sans savoir à reconstruire le programme brique par brique et aidant à suivre l'évolution du développement et des métriques.

Si nous pouvions recommencer ce projet en ayant ces réflexes, nous passerions indéniablement moins de temps sur la sélection des attributs et commencerions par automatiser la recherche des variables de plus grande qualité, conjointement à une automatisation des tests de corrélation afin de déterminer quelles sont les variables offrant le meilleur compromis entre qualité et pertinence. Ce temps gagné, nous aurions alors pu nous consacrer plus tôt à la mise en oeuvre d'algorithmes un peu plus complexes, ou en tout cas plus fins et plus adaptés, afin de tirer le maximum d'information de nos données. Nous avions par exemple songé à déployer des techniques de TAL sur les nombreuses zones de texte libre qui émaillent le jeu de données, mais ne l'avons pas fait par manque de temps. Nous aurions eu aussi le temps de réfléchir aux façons de poser le problème autrement : au lieu d'une classification multiclasse, nous aurions pu par exemple tenté une classification binaire, ne répondant plus à la question "Pour qui cet individu va-t-il voter?" mais plutôt "Cet individu va-t-il voter pour ce parti?", en la répétant pour toutes les modalités possibles. Enfin, nous aurions aussi pu aussi considérer le prisme des algorithmes de recommendation, ou celui du flux de données.

Quoiqu'il en soit, nous avons été et sommes restés intéressés par ce projet du début jusqu'à la fin et pensons qu'il constitue une entrée en matière assez idéale : le sujet était intéressant, le jeu de données suffisamment vaste pour nous amener à nous poser de sérieuses questions de (pré)traitement et à rencontrer des problèmes que nous aurions fini par rencontrer tôt ou tard, et suffisamment multimodal pour imaginer un grand panel de solutions possibles. Toutefois, une fois pris en main et appréhendé, il restait aussi suffisamment manipulable

pour ne pas rester bloqué très longtemps sur une question et ainsi se frustrer et suffisamment pertinent pour avoir tout de même des résultats finaux satisfaisants, même si imparfaits.

Appendix A. Attributs évoqués et équivalence dans la documentation (par ordre d'apparition dans ce rapport)

A.1. Liste 1 - Attributs permettant d'identifier un individu

- `cps19_yob` Tout d'abord, en quelle année êtes-vous né(e)?
- `cps19_gender` Êtes-vous...
- `cps19_education` Quel est votre plus haut niveau de scolarité complété?
- `cps19_employment` Quel est votre statut d'emploi actuel?
- `cps19_religion` Quelle est votre religion, si vous en avez une?
- `cps19_bornin_canada` Êtes-vous né au Canada?

A.2. Liste 2 - Attributs ajoutés à la deuxième itération

- `cps19_prov_id` En politique provinciale, vous considérez-vous habituellement comme étant:
- `cps19_fed_id` En politique fédérale, vous considérez-vous habituellement comme étant :
- `cps19_vote_2015` Pour quel parti avez-vous voté?

A.3. Liste 3 - Attributs ajoutés à la troisième itération

- `cps19_spend_educ` Combien le gouvernement fédéral devrait-il dépenser en éducation?
- `cps19_env` Combien le gouvernement fédéral devrait-il dépenser en environnement?
- `cps19_spend_just_law` Combien le gouvernement fédéral devrait-il dépenser pour la criminalité et la justice ou la police et les forces de l'ordre ?
- `cps19_spend_defence` Combien le gouvernement fédéral devrait-il dépenser en défense?
- `cps19_spend_imm_min` Combien le gouvernement fédéral devrait-il dépenser pour les immigrants et les minorités?
- `cps19_lead_int` Parmi les chefs de partis fédéraux énumérés ci-dessous, le(s)quel(s) trouvezvous intelligent(s)?
- `cps19_lead_strong` Parmi les chefs de partis fédéraux ci-dessous, le(s)quel(s) manifeste(nt) un leadership fort ?

- `cps19_lead_trust` PParmi les chefs de partis fédéraux énumérés ci-dessous, le(s)quel(s) trouvezvous digne(s) de confiance?

- `cps19_lead_cares` Parmi les chefs de partis fédéraux énumérés ci-dessous, le(s)quel(s) se souci(ent) vraiment des gens comme vous?

A.4. **Liste 4 - Attributs ajoutés à la 4ème itération**

- `cps19_income_number` Quel est le revenu total de votre ménage avant impôts en 2018? Cela doit inclure toutes les sources de revenus au millier de dollars près.

- `cps19_income_cat` Nous n'avons pas besoin du montant exact; le revenu de votre ménage se situe-t-il dans l'une des catégories suivantes?

- `cps19_province` Dans quelle province ou territoire habitez-vous présentement?

- `cps19_children` Avez-vous des enfants?

- `cps19_marital` Êtes-vous présentement marié(e), vivant avec un(e) conjoint(e), divorcé(e), séparé(e), veuf (veuve), ou n'avez-vous jamais été marié(e)?

- `cps19_union` Appartenez-vous à un syndicat?

- `cps19_sexuality` Vous considérez-vous comme étant :

- `cps19_demsat` Dans l'ensemble, quel est votre niveau de satisfaction quant au fonctionnement de la démocratie au Canada?

A.5. **Liste 5 - Attributs partitionnés par BIRCH**

- `cps19_pos_fptp` Le Canada devrait changer son système électoral afin de passer du mode de scrutin actuel à un système de "représentation proportionnelle".

- `cps19_pos_life` Les individus en phase terminale devraient pouvoir mettre fin à leurs jours avec l'aide d'un médecin.

- `cps19_pos_cannabis` La possession de cannabis devrait être une infraction pénale.

- `cps19_pos_carbon` Afin d'aider à réduire les émissions de gaz à effet de serre, le gouvernement fédéral devrait maintenir la taxe sur le carbone.

- `cps19_pos_energy` Le gouvernement fédéral devrait en faire davantage afin d'aider le secteur énergétique canadien, notamment en construisant des oléoducs.

- `cps19_pos_envreg` La réglementation environnementale devrait être plus stricte, même si elle oblige les consommateurs à payer des prix plus élevés.

- `cps19_pos_jobs` Lorsqu'il existe un conflit entre la protection de l'environnement et la création d'emplois, les emplois devraient avoir la priorité.
- `cps19_pos_subsid` Le gouvernement fédéral devrait mettre fin à toutes les subventions au développement des entreprises et de l'économie.

- `cps19_pos_trade` Il devrait y avoir plus de libre-échange avec d'autres pays, même si cela nuit à certaines industries au Canada.