AVIGNON UNIVERSITÉ

Démissions d'un organisme bancaire

Groupe IA-CLA
Damien Dallon
Nathanaël Lefèvre

16 janvier 2023

Master 2 informatique
ILSEN/IA
UE Business intelligence & Systèmes décisionnels
ECUE Application Business Intelligence

Responsable
Vincent Labatut

UFR
SCIENCES
TECHNOLOGIES
SANTÉ

AVIGNON UNIVERSITÉ
CERI

CENTRE
D'**ENSEIGNEMENT**
ET DE **RECHERCHE**
EN **INFORMATIQUE**
ceri.univ-avignon.fr

Sommaire

1 Présentation

1.1 Contexte

Un organisme bancaire a fait appel à nous pour mettre en place une solution de Machine Learning afin de détecter ses clients sociétaires qui sont sur le point de le quitter. Le but principal pour la banque est d'adapter sa relation client auprès de ceux identifiés comme démissionnaire afin de les convaincre de rester.

Par ailleurs, l'explicabilité du modèle est également un point important pour l'organisme bancaire qui souhaite savoir quelles caractéristiques influent le plus sur la classification.

1.2 Organisation

2 Données

2.1 Caractéristiques

Nous avons accès à des données extraites en 2007 décrivant les sociétaires de l'organisme. Il s'agit de deux fichiers de données tabulaires : `table1.csv` et `table2.csv`.

- `table1.csv` contient les 30 332 *démissionnaires* de l'organisme, pour la période allant de 1999 à 2006. Un *démissionnaire* est un sociétaire ayant quitté l'organisme. Ses attributs sont décrits dans la Table 1.

- `table2.csv` contient un échantillon aléatoire de 15 022 sociétaires, incluant des démissionnaires et des sociétaires actuels (pour des raisons de simplicité, nous considérerons comme "actuels" les sociétaires étant toujours clients de l'organisme au moment de l'extraction). Ses attributs sont décrits dans la Table 2.

2.2 Nettoyage

Rappelons que notre objectif est de pouvoir identifier les sociétaires démissionnaires, afin de pouvoir détecter des profils démissionnaires parmi les sociétaires actuels. Il nous faut donc le plus de données possibles combinant sociétaires actuels et démissionnaires et il est ainsi impensable de se contenter uniquement des données contenues dans `table2.csv`. Nous devons donc fusionner les deux tables mais cela implique des incompatibilités au niveau des attributs.

Afin de réaliser cette fusion, nous avons :

- Éliminé les attributs liés à la démission. En effet, le but étant de déterminer si un sociétaire est démissionnaire ou non, tout attribut en lien avec la démission n'a pas lieu d'être car permettrait une identification bien trop évidente.

- Utilisé les attributs dont les valeurs sont des dates pour en déduire des durées. Une date étant de nature complexe, les algorithmes ne peuvent pas les utiliser. Ainsi, afin de conserver un maximum d'information et d'homogénéiser les tables nous avons réalisé les traitements suivants :

    - Calcul de la durée d'adhésion avec les attributs `DTADH` et `DTDEM` de `table2.csv`, équivalent à l'attribut `ADH` de `table1.csv`. Plus précisément, le calcul est effectué entre `DTADH` et `DTDEM` si le sociétaire est démissionnaire, sinon entre `DTADH` et 2007

Variable	Type	Nature
ID	Quantitative discret	int
CDSEXE	Qualitative nominale	int
MTREV	Quantitative discret	int
NBENF	Quantitative discret	int
CDSITFAM	Qualitative nominale	char
DTADH	Quantitative discret	date
CDTMT	Qualitative nominale	int
CDDEM	Qualitative nominale	int
DTDEM	Quantitative discret	date
ANNEEDEM	Quantitative discret	int
CDMOTDEM	Qualitative nominale	string
CDCATCL	Qualitative nominale	int
AGEAD	Quantitative discret	int
RANGAGEAD	Qualitative ordinale	string
AGEDEM	Quantitative discret	int
RANGAGEDEM	Qualitative ordinale	string
RANGDEM	Qualitative ordinale	string
ADH	Quantitative discret	int
RANGADH	Qualitative ordinale	string

Variable	Description
ID	Identifiant unique (dans ce fichier)
CDSEXE	Code relatif au sexe
MTREV	Montant des revenus
NBENF	Nombre d'enfants
CDSITFAM	Situation familiale
DTADH	Date d'adhésion à l'organisme bancaire
CDTMT	Code représentant le statut du sociétaire (catégorie)
CDDEM	Code de démission
DTDEM	Date de démission
ANNEEDEM	Année de démission
CDMOTDEM	Motif de la démission (catégorie)
CDCATCL	Type de client (catégorie)
AGEAD	Âge du client à l'adhésion, en années
RANGAGEAD	Tranche d'âge du client à l'adhésion
AGEDEM	Âge du client à la démission, en années
RANGAGEDEM	Tranche d'âge du client à la démission
RANGDEM	Date de la démission au format N AAAA (code puis année)
ADH	Durée de la période d'adhésion, en années
RANGADH	Tranche de la durée de la période d'adhésion

Table 1. Attributs présents dans le fichier `table1.csv`

(date de l'extraction des données).

- Calcul de l'âge à l'adhésion avec les attributs `DTNAIS` et `DTADH` de `table2.csv`, équivalent à l'attribut `AGEAD` de `table1.csv`. Avec suppression au préalable des individus dans `table2.csv` dont l'attribut `DTNAIS` a pour valeur "000-00-00". En effet sans la date de naissance il nous était impossible de connaître l'âge à l'adhésion, et nous avons préféré supprimer ces individus car certains algorithme ne supportent pas les valeurs manquantes.

Variable	Type	Nature
ID	Quantitative discret	int
CDSEXE	Qualitative nominale	int
DTNAIS	Quantitative discret	date
MTREV	Quantitative discret	int
NBENF	Quantitative discret	int
CDSITFAM	Qualitative nominale	char
DTADH	Quantitative discret	date
CDTMT	Qualitative nominale	int
CDMOTDEM	Qualitative nominale	string
CDCATCL	Qualitative nominale	int
BPADH	Quantitative discret	int
DTDEM	Quantitative discret	date

Variable	Description
ID	Identifiant unique (dans ce fichier)
CDSEXE	Code relatif au sexe
DTNAIS	Date de naissance
MTREV	Montant des revenus
NBENF	Nombre d'enfants
CDSITFAM	Situation familiale
DTADH	Date d'adhésion à l'organisme bancaire
CDTMT	Code représentant le statut du sociétaire (catégorie)
CDMOTDEM	Motif de la démission (catégorie)
CDCATCL	Type de client (catégorie)
BPADH	Signification inconnue
DTDEM	Date de démission

Table 2. Attributs présents dans le fichier `table2.csv`

Après ces calculs nous avons supprimés les attributs ayant une date pour valeur. Aussi, afin d'harmoniser les attributs des deux tables, nous avons créer les attributs `ADH` et `AGEAD` dans la table issue de `table2.csv`.

- Éliminé les attributs ID des deux tables.
- Éliminé les attributs représentant des intervalles. Ces attributs étant soit liés à la démission, soit à la période d'adhésion ou à l'âge d'adhésion que nous avons calculé, nous avons décidé qu'ils ne nous seraient pas utiles.
- Éliminé l'attribut `BPADH` car nous ne connaissons pas sa signification et ne savons pas si la valeur "0" correspond à un manque d'information ou non.
- Rajouté l'attribut `ISDEM` aux deux tables en tant que label. "1" signifie que l'individu est démissionnaire, "0" sinon.

Ainsi après nettoyage et fusion des deux tables issues des fichiers `table1.csv` et `table2.csv` nous obtenons une table avec les attributs listés dans la Table 3.

2.3 Analyse descriptive

Variable	Type	Nature
CDSEXE	Qualitative nominale	int
MTREV	Quantitative discret	int
NBENF	Quantitative discret	int
CDSITFAM	Qualitative nominale	char
CDTMT	Qualitative nominale	int
CDCATCL	Qualitative nominale	int
AGEAD	Quantitative discret	int
ADH	Quantitative discret	int
ISDEM	Qualitative nominale	int

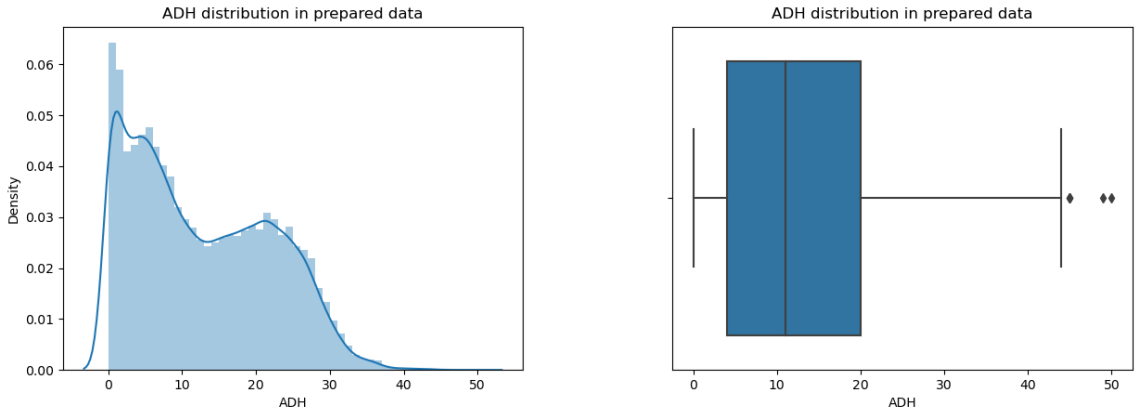
Variable	Description
CDSEXE	Code relatif au sexe
MTREV	Montant des revenus
NBENF	Nombre d'enfants
CDSITFAM	Situation familiale
CDTMT	Code représentant le statut du sociétaire (catégorie)
CDCATCL	Type de client (catégorie)
AGEAD	Âge du client à l'adhésion, en années
ADH	Durée de la période d'adhésion, en années
ISDEM	Démissionnaire ou non

Table 3. Attributs présents dans la table après nettoyage des données

2.3.1 Analyse par attribut

Une fois nos données nettoyées, nous avons analysé celles-ci pour étudier la distribution des différents attributs. Notons que voici les profils pour chaque attribut :

ADH - Durée de la période d'adhésion, en années



(a) Densité [ADH] **(b)** Diagramme moustache [ADH]

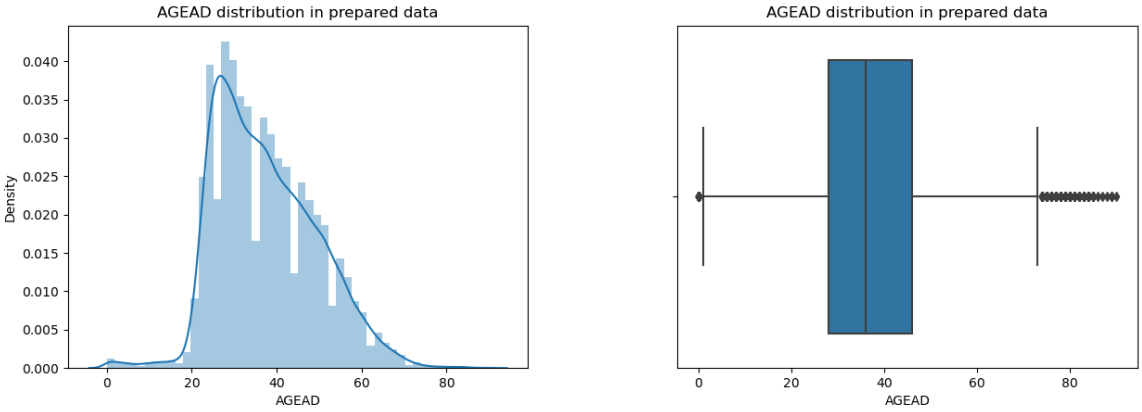
Figure 1. ADH-distribution

	count	mean	std	min	25%	50%	75%	max
ADH	45185.00	12.57	9.33	0.00	4.00	11.00	20.00	50.00

Table 4. ADH-table

Dans la figure ADH-distribution, on peut observer deux profils de sociétaires : ceux qui restent entre 0 et 13 ans et ceux qui restent entre 10 et 30 ans voire plus. Notons que les deux groupes sont assez bien réparti puisque la moyenne est à 12.57, c'est à dire à la frontière des deux groupes. (voir ADH-table)

AGEAD - Âge du client à l'adhésion, en années



(a) Densité [AGEAD] **(b)** Diagramme moustache [AGEAD]

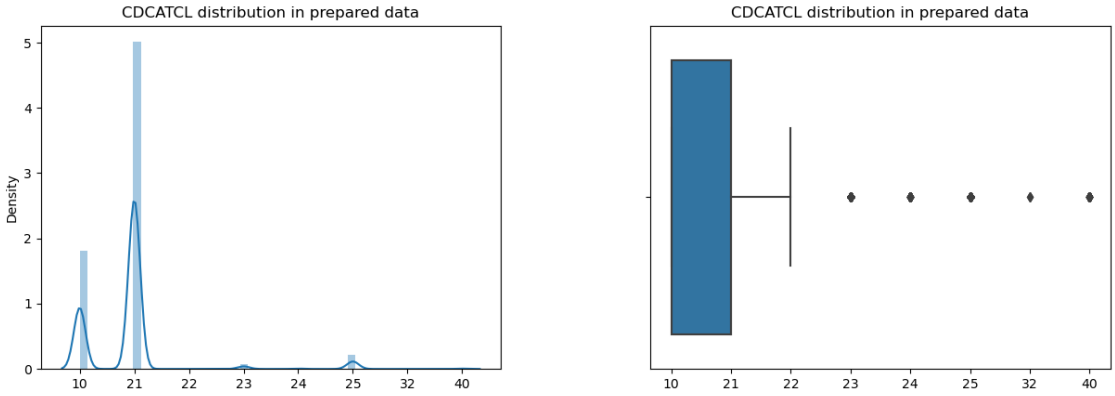
Figure 2. AGEAD-distribution

	count	mean	std	min	25%	50%	75%	max
AGEAD	45185.00	37.45	11.90	0.00	28.00	36.00	46.00	90.00

Table 5. AGEAD-table

Nous observons qu'il y a très peu d'adhésion entre 0 et 20 ans comme le montre la figure AGEAD-distribution. On peut imaginer qu'à de bas âges, ce sont surtout les parents qui ouvrent un compte à leurs enfants. Le nombre de sociétaire diminue pour un age d'adhésion allant d'environ 25 ans jusqu'à 80 ans voire plus, avec un maximum de 90 ans. notons que 75% des sociétaire ont 46 ans ou moins et que la moyenne d'age à l'adhésion est de 37.45 ans (voir AGEAD-table)).

CDCATCL - Type de client (catégorie)



(a) Densité [CDCATCL] **(b)** Diagramme moustache [CDCATCL]

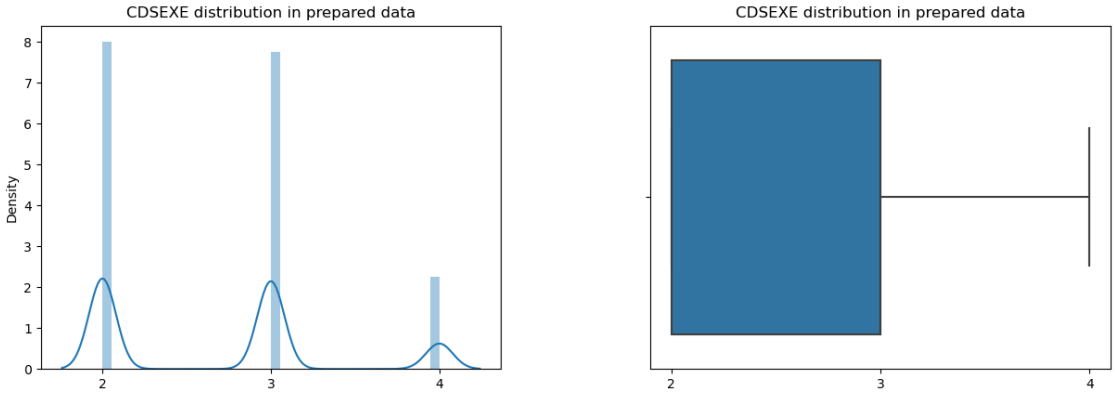
Figure 3. CDCATCL-distribution

	10	21	22	23	24	25	32	40
CDCATCL	0.25	0.70	0.00	0.01	0.00	0.03	0.00	0.00

Table 6. CDCATCL-table

L'organisme bancaire classe ses clients en 3 catégories et on observe que la catégorie 21 est largement majoritaire avec 70% des sociétaires qui en font partie. La catégorie 10 arrive en seconde position avec 25% des clients et les 5% restant sont dans les autres catégories. Ceci est illustré dans les figures CDCATCL-distribution et CDCATCL-table

CDSEXE - Code relatif au sexe



(a) Densité [CDSEXE] **(b)** Diagramme moustache [CDSEXE]

Figure 4. CDSEXE-distribution

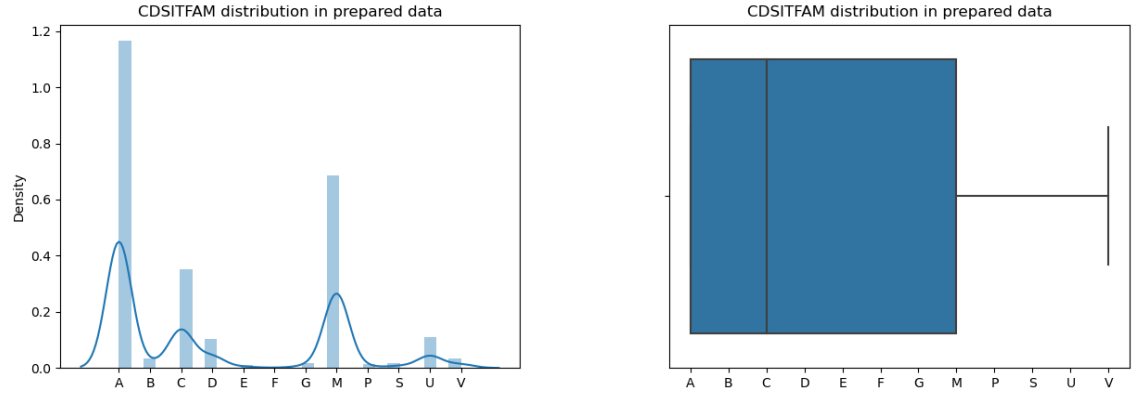
	2	3	4
CDSEXE	0.44	0.43	0.12

Table 7. CDSEXE-table

Dans les figures CDSEXE-distribution et CDSEXE-table, on observe que 88% des clients sont réparti équitablement entre les catégories de sexe 2 et 3 et les 12%restants sont dans

la catégorie 4.

CDSITFAM - Situation familiale



(a) Densité [CDSITFAM] **(b)** Diagramme moustache [CDSITFAM]

Figure 5. CDSITFAM-distribution

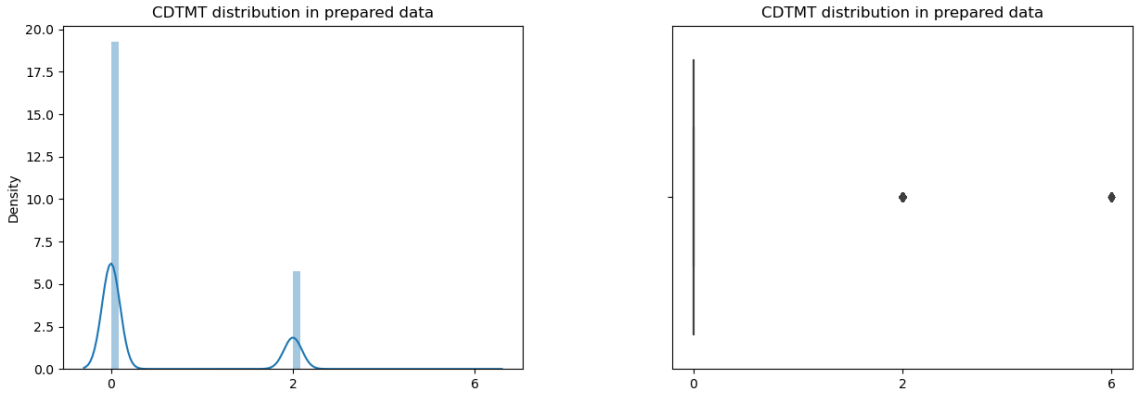
	A	B	C	D	E	F	G	M
CDSITFAM	0.46	0.01	0.14	0.04	0.00	0.00	0.01	0.27

	P	S	U	V
CDSITFAM	0.00	0.01	0.04	0.01

Table 8. CDSITFAM-table

L'analyse de la figure CDSITFAM-table révèle que parmis les client de la banque, 46% sont dans la catégorie de situation famillliale A, 27% dans la M, 14% dans la C et tous les autres clients sont réparti dans les 9 autres catégories, avec un légère prédilection pour les catégories U et D, comme le montre la figure CDSITFAM-distribution.

CDTMT - Code représentant le statut du sociétaire



(a) Densité [CDTMT] **(b)** Diagramme moustache [CDTMT]

Figure 6. CDTMT-distribution

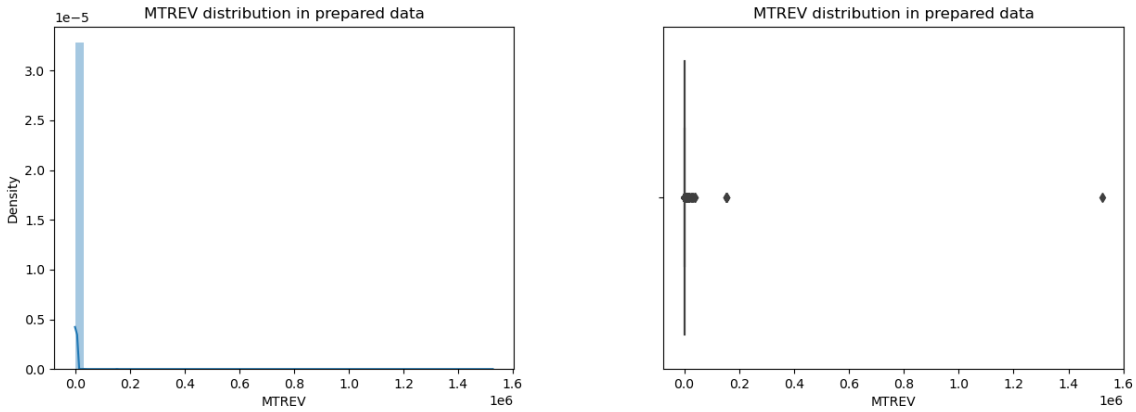
77% des clients de la banque ont un statut classé 0 par la banque et 23% classé 0, comme

	0	2	6
CDTMT	0.77	0.23	0.00

Table 9. CDTMT-table

cela peut être vu dans les figures CDTMT-table et CDTMT-distribution.

MTREV - Montant des revenus



(a) Densité [MTREV]

(b) Diagramme moustache [MTREV]

Figure 7. MTREV-distribution

	count	mean	std	min	25%	50%	75%	max
MTREV	45185.00	420.65	7377.32	0.00	0.00	0.00	0.00	1524490.00

Table 10. MTREV-table

Dans la figure MTREV-distribution, on observe un seul pic à 0 et beaucoup d'outliers, avec notamment une personne avec une valeur de revenu annuel de 1 524 490, comme le montre la figure MTREV-table. La plupart des sociétaire ont soit un revenu annuel nul, soit ne déclarent pas leurs revenus.

3 Méthodes

3.1 Outils de fouille

Nous nous sommes appuyés sur la bibliothèque sklearn pour implémenter nos algorithmes de fouille de données. Il nous a été demandé d'implémenter un SVM (Support Vector Machine), un KNN (K-Nearest Neighbors), Naive Bayes pour données catégorielles et nous avons choisi de mettre en place un MLP (Multi Layer Perceptron). Passons en revue chacun de ces algorithmes.

3.1.1 SVM (Support Vector Machine) [imposé]

Pour cette méthode, nous utiliserons l'implémentation SVC de sklearn.svm.

Description

Les SVM (Support Vector Machines) sont un type de modèle de classification supervisée. Les SVM cherchent à trouver le meilleur plan de séparation (appelé vecteur de support)

entre les groupes de données en maximisant la marge, c'est-à-dire la distance entre les données les plus proches des groupes séparés. Les plan de séparation produits sont en réalité des hyperplans de dimension n-1, n étant le nombre d'attributs.

Justification

Le choix de cette méthode se justifie par l'idée qu'il doit exister des seuils pour chaque groupe de valeur d'attributs de sorte qu'une fois ces seuils dépassés, les individu changent de catégories. Par exemple, pour tous les attributs fixé sauf un, il existe un seuil pour ce dernier attribut qui fait changer la catégorie de l'individu classé. Pour une autre valeur des attributs fixés, le seuil du dernier attribut ne serait toutefois pas le même. Par exemple, pour tout attribut fixé sauf l'âge à l'adhésion on peut imaginer que pour un personne riche, si elle a moins de 20 ans, elle est démissionnaire car elle n'as pas choisi sa banque mais ce sont ces parents, et inversement si elle a plus de 20 ans. Si la personne est un peu moins riche, le seuil d'âge passe peut être à 21 ans, et si cette fois c'est l'âge à l'adhésion qui change, alors le seuil sur l'âge passe peut-être à 19 ans
En proposant un hyperplan de séparation des classes, les SVM permettent de prendre en compte cette vision des choses. En effet, pour n-1 attributs fixés, le fait qu'un point se trouve d'un côté ou l'autre de l'hyperplan ne dépends que de la valeur du n-ème attribut.
Si nos hypothèses sont vérifiée, un SVM devrait donc bien s'en sortir tout en demandant moins de ressources qu'un technique plus complexe par exemple.

Paramètres

Voici les différents paramètres pris en charge par le modèle SVC de sklearn. Le nom de chaque paramètre est donné avec sa valeur par défaut et une explication.

- **C**=1.0 - Le paramètre C dans le module SVM de sklearn est le paramètre de régularisation. Il permet de contrôler le trade-off entre l'évitement de l'overfitting et la maximisation de la classification correcte. Plus C est grand, plus la pénalité pour les erreurs de classification sera élevée, ce qui entraînera un modèle plus complexe et qui s'ajustera davantage aux données d'entraînement. Plus C est petit, plus la pénalité sera faible et le modèle sera plus simple et plus généralisable.

- **kernel**='rbf' - Le noyau permet de projeter les données dans un espace où celles-ci sont linéairement séparable ou plus facilement linéairement séparables. "linear" utilise une fonction linéaire pour mapper les données, ce qui est adéquat pour les données déjà linéairement séparables, "rbf" utilise une fonction gaussienne et "poly" utilise une fonction polynômiale. Il existe aussi les option "sigmoid" et précomputed, et il est possible de fournir son propre kernel sous la forme d'un callable.

- **degree**=3 - degré de la fontion polynomiale pour le paramètre kernel="poly". Les autres kernel ignorent ce paramètre.

- **gamma**='scale' - Le paramètre gamma est utilisé conjointement avec le paramètre kernel pour contrôler l'influence d'un seul exemple d'entraînement. Un gamma plus grand signifie qu'un exemple d'entraînement aura une influence plus importante sur la décision, ce qui peut entraîner un ajustement plus serré aux données d'entraînement mais avec un risque d'overfitting.

- **coef0**=0.0 - Il s'agit du terme constant dans la fonction polynomiale ou la sigmoid

- **shrinking**=True - Il s'agit d'une heuristique permettant d'accélérer l'entraînement mais

conduisant potentiellement à un résultat non optimal, puisqu'il s'agit justement d'une heuristique.

- **probability**=False - Permet de calculer la probabilité de chaque classe si réglé sur True.
- **tol**=0.001 - Permet de spécifier la tolérance pour l'arrêt de la résolution du problème d'optimisation. Il s'agit d'un seuil de convergence pour la différence entre les valeurs de la fonction de coût entre deux itérations consécutives.
- **cache_size**=200 - Il s'agit de la taille du cache en Mo pour les résultat intermédaires de la fonction kernel.
- **class_weight**=None - permet de mettre des poids sur les différentes classes.
- **verbose**=False - active le mode verbeux.
- **max_iter**=-1 - Permet de limiter le nombre d'itérations. Il n'y a pas de limite par défaut.
- **decision_function_shape**='ovr' - permet de choisir la forme de la fonction de décision. "ovr" pour "One Versus Rest" qui propose une fonction de decision entre chaque classe et l'ensemble des autres et "ovo" pour "One Versus One" qui propose une fonction de classification entre chaque paire de classes.
- **break_ties**=False - Si decision_function_shape="ovr" et que le nombre de classes est suppérieur à 2, permet de choisir si il faut départager les classes en cas de prédiction non transitive des fonctions de décision (par exemple, si pour un individu la première prédit B plutôt que A, la seconde C plutôt que B et la dernière A plutôt que C). La classification pour cet individu est alors choisi aléatoirement si le paramètre est à False et en fonction de la confiance de chaque fonction.
- **random_state**=None - Permet de gérer le caractère aléatoire du mélange des données pour le calcul des praobabilités lors que probability est réglé sur True.

3.1.2 KNN (K-Nearest Neighbors) [imposé]

Nous utiliserons l'implémentation KNeighborsClassifier de sklearn.neighbors pour cette méthode.

Description

KNN (k-Nearest Neighbors) est un algorithme de classification supervisée qui assigne une étiquette à une observation en se basant sur les étiquettes des k observations les plus proches dans l'espace des features. Il est basé sur l'idée qu'une observation ressemble davantage à ses voisins les plus proches qu'à des observations plus éloignées. Le nombre k est un paramètre choisi par l'utilisateur, il détermine combien de voisins doivent être pris en compte lors de l'attribution de l'étiquette.

Justification

L'utilisation de cette méthode est justifiée par le fait qu'il existe certainement des groupes de personnes similaires dans chaque catégories. Ainsi, si une personne ressemble à un groupe en particulier et que ce groupe fait partie des démissionnaires, alors on peut espérer que la personne le soit également.
Si l'hypothèse avancée dans le choix des SVM n'est pas vérifiée, alors pour tout attribut fixé

sauf un, il existe pour l'attribut restant plusieurs plages de valeurs pour lesquelles l'individu est démissionnaire. Par exemple, pour tout attributs fixés à une certaine valeur sauf l'âge, il est possible que les personnes entre 10 et 20 ans et celles entre 30 et 40 ans soient démissionnaires et que les autres ne le soient pas.
Puisque les KNN ne définissent pas de seuils, ils sont plus adaptés que les SVM dans le cas où cette nouvelle supposition est vérifiée. En effet, pour tout attributs similaires aux valeurs des attributs fixés précédemment, si la personne a entre 10 et 20 ou 30 et 40 ans, on peut espérer qu'elle sera proche de personnes démissionnaires par exemple.

Paramètres

Voici les différents paramètres pris en charge par le modèle KNeighborsClassifier de sklearn. Le nom de chaque paramètre est donné avec sa valeur par défaut et une explication.

- **n_neighbors**=5 - nombre de voisin à prendre en compte pour prendre la décision.
- **weights**='uniform'- poids des voisins dans la décision. "uniform" donne des poids égaux à tous les voisins, "distance" donne un poids inversement proportionnel à la distance et il est possible de fournir un callable.
- **algorithm**='auto' - permet de choisir l'algorithme utilisé pour trouver les plus proches voisins. "auto" tente de sélectionner l'algorithme le plus efficace en fonction des données d'entraînement.
- **leaf_size**=30 - permet de choisir la taille des feulles des arbres pour certains algorithmes du paramètre algorithm.
- **p**=2 - paramètre de puissance pour la métrique Minkowski.
- **metric**='minkowski' - choix de la métrique pour calculer les distances entre les points.
- **metric_params**=None - paramètres supplémentaires pour la métrique de distance.
- **n_jobs**=None - Permet de régler le nombre de job parallèle pour la recherche des k plus proches voisins.

3.1.3 Naive Bayes [imposé]

Nous ferons appel à l'implémentation CategoricalNB de sklearn.naive_bayes.

Description

Naive Bayes est un algorithme de classification statistique basé sur l'utilisation de la théorie de Bayes pour l'estimation des probabilités. Il est considéré comme "naïf" car il suppose que toutes les variables sont indépendantes les unes des autres, ce qui n'est généralement pas le cas dans les données réelles. Naive Bayes est capable de s'appuyer sur des probabilités d'observation conditionnelle observées dans un échantillon pour prédire la condition en fonction des observation.
Par exemple, dans la détection de spam, on peut calculer la probabilité de chaque mot sachant que le mail est un spam et celle sachant que le courrier n'est pas un spam. Naive Bayes est alors capable, étant donné les mots observés dans un mail, de donner la probabilité que le mail soit un spam.

Justification

Cet algorithme, dans sa version catégorielle, s'applique très bien ici puisque l'on a beaucoup de données de ce type et que nous pouvons facilement adapter les autres données. Cet algorithme est intéressant car il propose une toute autre approche que les SVM et KNN puisque celle-ci est probabiliste.
Dans le cas où les attributs n'ont réellement aucune corrélation entre eux, alors d'une, il est inutile de vouloir définir des seuils pour chaque attribut basé sur les valeurs des autres (là où exellent les SVM), et il y a potentiellement peu de chance de trouver des individus proches puisque la valeur d'un attribut influe pas sur les autres. KNN ne serait donc pas d'une grande aide dans ce cas. En revanche, Naive Bayes serait tout à fait capable, indépendamment de toute définition de seuil et indépendemment de la distance entre individus, de définir les probabilité de chaque valeur d'attribut sachant que la personne est démissionnaire et inversement, afin d'établir la probabilité qu'une personne soit démissionnaire ou non, sachant les valeurs des attributs.

Paramètres
Voici les différents paramètres pris en charge par le modèle CategoricalNB de sklearn. Le nom de chaque paramètre est donné avec sa valeur par défaut et une explication.

- **alpha**=1.0 - paramètre de smoothing permettant d'attribuer une probabilité non nulle pour les valeurs non rencontrées dans l'ensemble d'apprentissage.

- **force_alpha**='warn' - par défaut, alpha a une valeur bornée par 1E-10. Mettre ce paramètre à True permet de supprimer cette limite. Si alpha est trop proche de zero, celà peut revenir à simplement désactiver le smoothing (ce qui peut efectivement être fait en mettant alpha à exactement 0 et ce paramètre à True).

- **fit_prior**=True - indique si l'algorithme doit apprendre les probabilités à priori des classes (True) ou utiliser une probabilité uniforme (False).

- **class_prior**=None - Permet de préciser la probabilité à priori des différentes classes si elles sont connue. Dans ce cas, fit_prior est ignoré.

- **min_categories**=None - indique le nombre de catégorie minimum par feature. Par défaut, il est déduit des données d'entraînement.

3.1.4 MLP (Multi Layer Perceptron) [selectionné]

Nous ferons appel à l'implémentation MLPClassifier de sklearn.neural_network pour la mise en oeuvre de cette méthode.

Description
Les réseaux de neurones multi-couches (abbrévié MLP en anglais) sont un type de réseau de neurones artificiels qui utilisent plusieurs couches de neurones connectés entre eux. Il sont formés de couches d'entrée, cachées et de sortie. Les données d'entrée sont traitées par les couches cachées, qui utilisent des fonctions d'activation pour produire des sorties sont ensuite traitées par les couches suivantes pour finir par la couche de sortie pour produire une réponse ou une prédiction. Les réseaux de neurones permettent en général une bonne généralisation grâce à leur capacité à apprendre des fonctions de degré arbitraire dépendant seulement de la taille de leurs couches et de leur nombre. d'autres paramètres influent bien évidement la précision, la vitesse d'apprentissage, et d'autres aspects.

Justification

L'utilisation d'un MLP se justifie dans le cas ou un simple séparateur n'est pas suffisant, qu'il est difficile de faire des groupes d'individus et que les relations entre attributs sont complexes. À l'heure actuelle, les réseaux de neurones sont généralement très performants et nous sommes curieux de voir si ils permettent d'obtenir de meilleurs résultat pour la tâche qui nous a été donnée.

Paramètres

- **hidden_layer_sizes**=(100,) - Vecteur de tailles des couches cachées de dimension n-2 (les tailles des couches d'entrées et de sortie dépendent des données et de la tache de prédiction).

- **activation**='relu' - type de fonction d'activation à la sortie des couches cachées.

- **solver**='adam' - type de solver pour l'optimisation des poids. Certains solvers utilisent un learning rate adaptatif et permettent de converger plus vite vers la solution dans certains cas, ou de dépasser des minimum locaux par exemple. "adam" est rapide à converger et "SGD" est plus lent mais généralise mieux, par exemple.

- **alpha**=0.0001 - Force du paramètre de régularisation L2. Plus il est élevé, plus la régularisation est élevée et les poids sont forcés à être petits. Le but de la régularisation est d'éviter le sur-apprentissage.

- **batch_size**='auto' - Taille des minibatch pour l'apprentissage. Utiliser des minibatch permet notamment de réduire le temps nécéssaire à l'apprentissage .

- **learning_rate**='constant' - Uniquement utilisé pour le solveur "sgd", permet de choisir la stratégie d'adaptation du learning rate. "constant" n'adapte pas le learning rate (lr), "invscaling" l'adapte en le faisant diminuer à chaque étape et "adaptive" le garde constant le plus possible puis le divise par 5 à chaque fois que, pour deux époques consécutives, la loss ne diminue pas assez.

- **learning_rate_init**=0.001 - Taux ou pas d'apprentissage. Influe sur la vitesse à laquelle sont modifiés les poids.

- **power_t**=0.5 - Paramètre utile pour "invscaling.

- **max_iter**=200 - Nombre maximum d'itération. Permet de stopper l'apprentissage si le MLP ne converge pas dans un temps convenable.

- **shuffle**=True - Indique si les exemples doivent être mélangés à chaque itération (Seulement si le solver est "sgd" ou "adam").

- **random_state**=None - Graine numérique pour l'initialisation des poids et biais.

- **tol**=0.0001 - Permet de spécifier la tolérance pour l'arrêt de la résolution du problème d'optimisation. Il s'agit d'un seuil de convergence pour la différence entre les valeurs de la fonction de coût entre deux itérations consécutives.

- **verbose**=False - Permet d'activer le mode verbeux.

- **warm_start**=False - Si activé, permet d'utiliser la solution du dernier appel à la méthode fit pour initialiser les poids. Peut notamment permettre de faire de fine tuning sur de nouvelles données.

- **momentum**=0.9 - Momentum pour la descente du gradient avec "sgd". Le momentum permet de régler le poids de la prise en compte des anciennes valeurs du learning rate pour sa mise à jour.

- **nesterovs_momentum**=True - Version alternative pour l'algorithme du momentum (seulement si solver = "sgd").

- **early_stopping**=False - réserve validation_fraction% des données pour de la validation croisée et arrête l'entrainement lorsque n_iter_no_change itération consécutives n'ont pas suffit à améliorer le score sur l'ensemble de validation.

- **validation_fraction**=0.1 - Pourcentage des données réservé à la validation lorsque early_stopping est à True.

- **beta_1**=0.9 - Paramètre pour le premier momentum de Adam.

- **beta_2**=0.999 - Paramètre pour le second momentum de Adam.

- **epsilon**=1e-08 - Paramètre de stabilité pour Adam.
- **n_iter_no_change**=10 - Nombre d'itération consécutives pour tenter de contenter le paramètre tol avant arrêt de l'algorithme en cas de non contentement.

- **max_fun**=15000 - Nombre maximum d'appel à la fonction de coût pour le solver "lbfgs".

3.2 Recodage

Les méthodes évoquées précédemment nécessitent un recodage des données afin d'être appliquées. Nous avons utilisé quatre types de recodage :

- La standardisation pour des attributs quantitatifs. Il s'agit, pour chaque valeur d'un attribut, de soustraire la moyenne des valeurs de cet attribut puis de diviser par l'écart-type. Cela permet d'obtenir des valeurs sur la même échelle pour un attribut.

- L'encodage One Hot pour des attributs qualitatifs. Il s'agit de créer un nouvel attribut par valeur possible pour un attribut et d'en indiquer la présence ou non pour un individu. Cela permet de convertir des attributs qualitatifs nominaux en quantitatifs sans y introduire une notion d'ordre.

- L'encodage ordinal pour des attributs qualitatifs. Il s'agit d'assigner une valeur discrète par valeur possible pour un attribut. Cela permet de convertir des attributs qualitatifs en attributs quantitatifs.

- La discrétisation pour des attributs quantitatifs. Il s'agit de créer des intervalles de valeurs pour un attribut de façon à les considérer comme des classes. Cela permet de convertir les attributs quantitatifs en attributs qualitatifs.

Ainsi pour chaque outil de fouille listé dans la section précédente nous appliquons les

transformations suivantes :

- SVM et MLP : Le SVM et le MLP ne supportant que des attributs quantitatifs et étant plus efficace pour des valeurs sur la même échelle, nous avons appliqué une standardisation pour les attributs quantitatifs et un encodage One Hot pour les attributs qualitatifs.

- KNN : Le KNN fonctionnant avec des coordonnées et une notion de distance, il ne supporte également que les attributs quantitatifs et est plus efficace si les valeurs sont sur la même échelle. Ainsi nous avons appliqué une standardisation des attributs quantitatifs. De plus nous avons opté pour une transformation des attributs qualitatifs par un encodage ordinal. En effet, le KNN fonctionnant avec des coordonnées et étant moins performant sur un grand nombre de dimensions, il est préférable d'éviter l'encodage One Hot qui apporte un grand nombre de dimensions et dont l'absence de notion d'ordre n'apporte pas d'avantage.

- Naive Bayes : Le Naive Bayes, dans notre cas catégoriel, fonctionne sur des attributs quantitatifs discrétisés. Ainsi les attributs qualitatifs subissent un encodage One Hot et les attributs quantitatifs subissent une discrétisation.

3.3 Évaluation

- Expliquez la méthode expérimentale utilisée pour évaluer la qualité des résultats, en **justifiant** vos choix (décomposition des données en apprentissage/validation/test, validation croisée, etc.).
- Décrivez la (ou les) mesure(s) utilisée(s) pour quantifier les performances, en **justifiant** là encore. Vous devez notamment donner une description **formelle** de la mesure (i.e. sa formule).
- Le cas échéant, indiquez la (ou les) méthode(s) statistiques utilisée(s) pour comparer ces mesures entre elles, en **justifiant** votre décision.

3.3.1 Méthode expérimentale

Pour évaluer la qualité des résultats, nous avons décomposé nos données en un ensemble d'entrainement contenant 60% des données, un ensemble de validation contenant 20% des données et un ensemble de test contenant les 20% restants. Cette séparation est plutôt classique et nous permet d'entrainer nos modèles sur la majorité des données puis de les évaluer sur une partie des donnée. Une fois le meilleur modèle choisi en fonction de sa performance sur l'ensemble de validation, nous le testerons sur l'ensemble de test.
Afin d'assurer que nos 3 ensembles aient la même distribution, nous avons séparé les démissionnaires des non démissionnaires, séparé chacun des deux groupes en ensembles d'entrainement, test et validation puis fusionné et mélangé les données de chaque sous-groupe.
Assurer la même distribution permet d'être sûr d'avoir des individus de la classe minoritaire dans chaque ensemble et aide notamment Naives Bayes qui s'appuie sur des probabilités.

3.3.2 Mesure de performance

Afin de mesurer les performances de nos modèles, nous avons récupéré une multitude de métriques afin d'avoir un aperçu de ce que nous avions à notre disposition. Nous avons ensuite déterminé celles qui nous intéressaient réellement sans pour autant supprimer les autres méthodes de notre processus. En effet, garder les autres méthodes permet de

conserver un point de vue différent sur nos modèles.
Voici les différentes métriques que nous avons explorées :

- **accuracy** - L'accuracy, ou "précision" en français, se calcule comme $Accuracy = \frac{N_{correct}}{N_{total}}$, où $N_{correct}$ est le nombre de prédictions correctes et N_{total} est le nombre total de prédictions. Le problème de cette métrique dans notre cas n'est pas très adaptée car nous avons une classe très minoritaire. Ainsi, si un modèle prédit toujours la classe majoritaire, son accuracy sera très élevée.
- **Matrice de confusion** - La matrice de confusion $\begin{pmatrix} TP & FP \\ FN & TN \end{pmatrix}$

3.4 Implémentation

- Décrivez le script rendu, en expliquant quel traitement est réalisé, notamment quelles classes de quelles bibliothèques sont utilisées, et comment elles s'enchaînent.
- Incluez dans cette description les éventuels prétraitements (en plus des méthodes de classification proprement dites).
- Attention, vous devez **décrire** votre script, et non **pas** inclure du code source dans votre rapport.

4 Résultats

> **Attention :** de façon générale, dans cette section, ne vous contentez pas de donner des résultats bruts. Vous devez montrer que vous êtes allés plus loin que cela en expliquant comment vous interprétez vos résultats par rapport au contexte (données, objectifs, application...).

4.1 Performances individuelles

- Donnez les résultats obtenus pour les différents algorithmes appliqués sur le jeu d'apprentissage (du moins : pour ceux qui possèdent une étape d'apprentissage), en présentant ça sous forme compacte au moyen de tableaux.
- Commentez et interprétez ces résultats. Détectez-vous des cas de *sous*-apprentissage ?
- Définissez une sous-section par algorithme.

4.2 Comparaison

- Donnez les résultats individuels obtenus pour les différents algorithmes/paramétrages appliqués sur le jeu de **validation**. Discutez l'évolution par rapports aux résultats obtenus sur le jeu d'apprentissage.
- Là encore, vous devez donner votre interprétation des résultats, et ne pas vous arrêter à une succession de tableaux et de graphiques. Détectez-vous des cas de *sur*-apprentissage ?
- Comparez les résultats obtenus par les différents algorithmes/paramétrages, de manière à identifier celui qui semble le plus adapté à nos besoins.

4.3 Généralisation

- Donnez les résultats pour l'algorithme/paramétrage sélectionné sur le jeu de test. Pour rappel, il ne doit y en avoir qu'**un seul** : il ne s'agit plus de comparer les modèles entre

eux, mais d'évaluer le pouvoir de généralisation du meilleur modèle obtenu à l'étape précédente.
- Discutez de sa faculté de généralisation : les résultats obtenus sur le jeu de test sont-ils du même niveau que ceux obtenus auparavant sur les autres jeux de données ? Statistiquement parlant, sont-ils **significativement** différents ou pas ?

4.4 Interprétation

- Décrivez les résultats de votre analyse destinée à identifier les attributs (et leurs valeurs) pertinents pour effectuer la prédiction demandée.
- Discutez ces résultats, notamment la nature des attributs et valeurs identifiés. Par exemple, la nature des attributs est-elle surprenante ou pas, relativement au problème posé ? Quels enseignements pouvez-vous en tirer du point de vue applicatif, toujours pour le problème posé dans le sujet ?

5 Conclusion

- Résumez très brièvement le travail accompli.
- Critiquez le projet : indiquez ce que vous avez apprécié, expliquez ce que le projet vous a apporté, précisez les aspects qui posent problème ou qui étaient ignorés mais que vous auriez voulu aborder. Ce point-là ne sera pas pris en compte pour l'évaluation du projet, mais permettra de l'améliorer le semestre prochain.
- Critiquez votre travail en indiquant les points positifs et les points négatifs (notamment les aspects que vous n'avez éventuellement pas traités).
- Proposez des solutions permettant de résoudre les limitations de votre travail.
- Proposez des perspectives sur ce projet, en indiquant comment le travail pourrait être étendu : analyses supplémentaires, problèmes connexes, etc.

Bibliographie. En ce qui concerne les références bibliographiques :
- Listez toutes les références bibliographiques citées dans le reste du document (en utilisant **BibTeX** si vous écrivez le rapport en LaTeX : par exemple [1], cf. le tutoriel fourni).
- Toute référence listée doit être citée **explicitement** et **à propos**, quelque part dans votre document.

Références

[1] Y.-C. Wei et C.-K. Cheng. « Towards efficient hierarchical designs by ratio cut partitioning ». In : *IEEE International Conference on Computer Aided Design*. 1989, p. 298-301. doi : 10.1109/ICCAD.1989.76957.