DATA INSIGHT GROUP

PROJET STA 211

Analyse prédictive des classes de chiffres à travers le jeu de données MFEAT

EL AJI Mohamed Amine/ CHERGUI SAMIR

12/01/2019

Table de matière

Table des figures et tableaux

1. Introduction

L'objectif de cette étude est de prédire à partir d'un ensemble de caractéristiques dans un jeu de données, des chiffres manuscrits de «0» à «9».

Le jeu de données décrit les caractéristiques de chiffres manuscrits («0» - «9») extraits d'une collection de cartes utilitaires. 200 motifs par classe (pour un total de 2 000 motifs) ont été numérisés en images binaires. Ces chiffres sont représentés par les six groupes de variables comme décrit ci-dessous :

1. mfeat-fou: 76 coefficients de Fourier des formes des caractères;
2. mfeat-fac: 216 corrélations de profils;
3. mfeat-kar: 64 coefficients Karhunen-Love;
4. mfeat-pix: 240 pixels en moyenne dans 2 x 3 fenêtres;
5. mfeat-zer: 47 moments Zernike;
6. mfeat-mor: 6 caractéristiques morphologiques.

Dans chaque fichier, les 2000 motifs sont stockés en ASCI sur 2000 lignes. Les 200 premiers modèles sont de classe «0», suivis par des ensembles de 200 modèles pour chacune des classes «1» - «9». Les modèles correspondants dans différents jeux de fonctions (fichiers) correspondent au même caractère d'origine.

Afin de déterminer la classe d'appartenance de chaque chiffre manuscrit, nous allons chercher une fonction discriminante qui permettra de prédire si un chiffre manuscrit est un 0, un 1… ou un 9, ceci tout en minimisant le taux d'erreur.

S'agissant d'une analyse supervisée, plusieurs méthodes peuvent être utilisées afin de prédire la classe d'appartenance à partir des caractéristiques : Régression logistique multinomiale, analyse discriminante, arbre de décision, Gradient Boosting, SVM, réseaux de neurones, classification naïve bayésienne…

Nous allons en particulier tester trois méthodes vues en cours qui sont le SVM et les réseaux de neurones et les arbres aléaratoires.

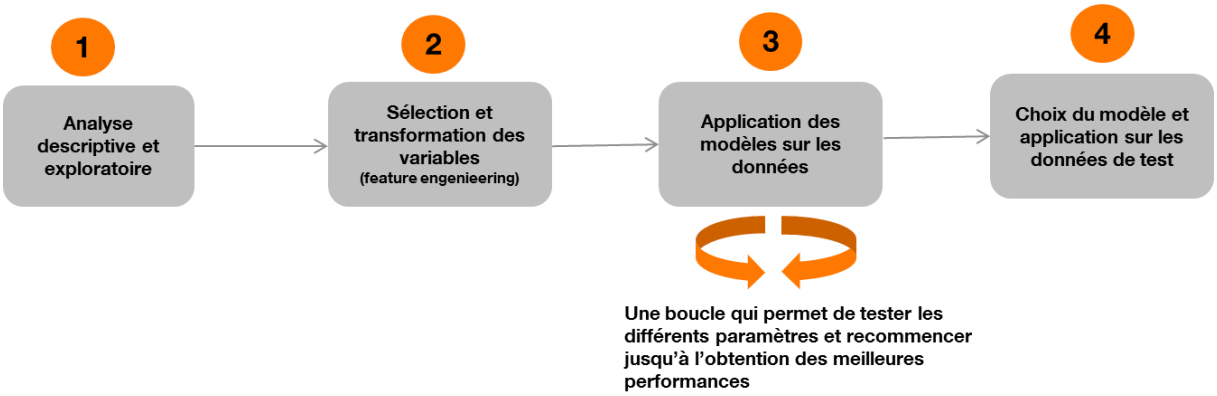
1.1 Méthodologie de traitement

Afin de procéder à cette analyse, nous allons tout d'abord nous intéresser à une étude exploratoire /descriptive des données, ce qui va nous permettre de mieux comprendre leurs structures.

Etant donné que le jeu de données est composé d'un groupe de variables décrivant des classes d'individus, nous allons :

- Analyser la signification de chaque groupe variable pour avoir une compréhension métier fine des variables traitées.
- Faire une analyse uni-variée et bi-variée des différentes variables afin de détecter les valeurs aberrantes, celles avec peu de variance, les variables très corrélées…
- Faire une classification de variables pour détecter celles qui portent la même information.
- Faire une AFM sur les groupes de variables pour détecter par groupe, les composantes qui apportent le maximum d'inertie par groupe (cercle de corrélation…),
- Faire un partitionnement K-means pour vérifier si on peut reconstituer les différentes classes à partir des variables discriminantes identifiées,

- Transformation des données à partir des résultats des analyses exploratoires (suppression ou création des nouvelles variables, normalisation, discrétisation….)
- Diviser les données en 2 parties : Apprentissage et test.
- Appliquer les modèles arbres aléatoires, SVM et réseau de neurone avec une validation croisée.
- Benchmark des trois modèles et synthèse.



2. Analyse descriptive et exploratoire des données

2.1 Analyse uni-variée

Notre jeu de donnée d'apprentissage est composé de 1500 individus et 650 variables.
La variable d'intérêt est la variable 'class' qui prend 10 valeurs différentes.
Une description globale nous donne un aperçu général des données, la moyenne, leurs variances et les déciles à 25%,50% et à 75%.

	fac_1	fac_2	fac_3	fac_4	fac_5	fac_6	fac_7	fac_8	fac_9	fac_10
count	1500	1500	1500	1500	1500	1500	1500	1500	1500	1500
mean	272,406667	323,128	773,4773333	756,7806667	640,510667	682,342667	19,374	18,4593333	15,6006667	9,08333333
std	91,5079437	109,49436	139,9275478	109,8108716	49,0186064	89,6329401	11,4070536	7,37278225	8,84345355	4,07620729
min	70	81	500	545	437	6	0	2	0	0
25%	210	247,75	654,75	668	608	642	10	14	8,75	5
50%	274	322	765,5	741	636	676	20	18	15	10
75%	338,25	406	881	832,25	667	715	29	24	22	12
max	515	564	1264	1134	814	986	42	39	39	17

Figure 1: Extrait de la description des différentes variables du jeu de données

Suite à cette analyse, nous pouvons constater :

- Absence des valeurs manquantes dans le jeu de données
- Absence des variables avec une variance à 0

Nous avons étudié les différentes variables en calculant leurs moyennes et leurs écarts type par classe :

class	fac_1	fac_2	fac_3	fac_4	fac_5	fac_6	fac_7	fac_8
0	128,806667	169,206667	605,36	657,913333	603,14	658,086667	4,22	6,88
1	224,766667	448,846667	877,38	676,033333	619,973333	667,926667	29,9133333	27,62
2	257,586667	360,666667	738,153333	853,613333	663,806667	717,413333	19,4133333	20,1733333
3	282,033333	376,206667	894,473333	720,28	650,533333	700,3	31,3733333	16,12
4	390,666667	345,44	830,626667	816,8	677,573333	554,14	21,6866667	21,1733333
5	286,713333	269,486667	722,173333	760,08	636,026667	666,233333	14,9666667	17,1
6	300,466667	311,28	666,426667	862,9	657,666667	667,933333	9,76	15,8666667
7	391,966667	447,753333	934,346667	833,306667	660,38	840,953333	31,8	23,88
8	168,026667	200,346667	635,586667	675,64	627,393333	673,006667	8,07333333	17,0333333
9	293,033333	302,046667	830,246667	711,24	608,613333	677,433333	22,5333333	18,7466667

Figure 2: Extrait de l'étude des variables par classe

Vu le nombre élevés des variables, une représentation graphique des distributions s'est avérée difficile. Néanmoins, nous allons faire une description analytique des différentes caractéristiques de chaque variable, pour présenter par la suite celles qui nous parait pertinentes:

	fac_1	fac_2	fac_3	fac_4	fac_5	fac_6	fac_7	fac_8	fac_9
count	1500	1500	1500	1500	1500	1500	1500	1500	1500
mean	272,406667	323,128	773,4773333	756,7806667	640,510667	682,342667	19,374	18,4593333	15,6006667
std	91,5079437	109,49436	139,9275478	109,8108716	49,0186064	89,6329401	11,4070536	7,37278225	8,84345355
min	70	81	500	545	437	6	0	2	0
25%	210	247,75	654,75	668	608	642	10	14	8,75
50%	274	322	765,5	741	636	676	20	18	15
75%	338,25	406	881	832,25	667	715	29	24	22
max	515	564	1264	1134	814	986	42	39	39
skew	-0,06976611	-0,09732603	0,331668317	0,475987916	0,55253905	-0,2413984	-0,09300881	0,11907742	0,20856896
kurtosis	-0,60315417	-0,80469732	-0,673627907	-0,447227223	0,94679747	6,5155012	-1,16372503	-0,637449	-0,74193764
p-value shapiro test	6,01E-09	1,36E-12	1,13E-15	1,51E-16	7,33E-15	3,79E-28	1,16E-21	2,07E-11	2,08E-14

Figure 3: Extrait de la description analytique des variables

Nous allons nous baser sur les valeurs des variances, asymétrie et principalement du kurtosis pour détecter les variables qui peuvent contenir des valeurs extrêmes.

Notre objectif par la suite et de réduire le nombre de dimension par groupe de variable, mais avant de procéder à cette transformation, il est primordial de traiter les valeurs extrêmes, car la méthode de réduction basée sur l'analyse de composante principale et la méthode d'analyse SVM y sont sensibles.

Avant de commencer les transformations des variables, nous allons garder 20% de notre échantillon pour tester les performances de notre modèle.

2.2 Détection des valeurs extrêmes

Comme indiqué dans le paragraphe précédent, un grand kurtosis indique la présence probable des valeurs extrêmes.

Plusieurs méthodes de détection sont possibles tel que : IQR ou z-score. Cette dernière est à écarter car nos données ne suivent pas une distribution normale (cf test shapiro).

Nous allons par la suite visualiser les variables avec un |Kurtosis|>2 qui sont :

fac_6,fac_27,fac_30,fac_54,fac_114,fac_186,fac_210,pix_1,pix_15,pix_22,pix_23,pix_24,pix_91,pix_105,pix_106,pix_121,pix_136,pix_211,pix_226,pix_240,zer_1,zer_2,zer_8,zer_14,zer_20,zer_30,zer_32,zer_39,zer_44,

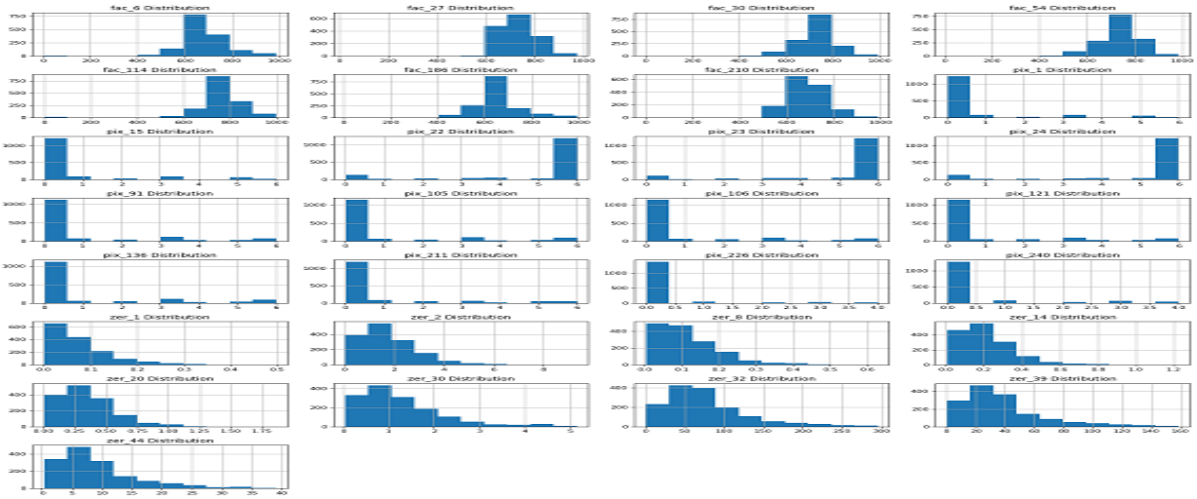


Figure 4: Histogrammes des variables avec un |kurtosis|>2

A travers les histogrammes, nous constatons :

- La présence des valeurs extrêmes pour les variables : 'fac_6','fac_27','fac_30', 'fac_54', 'fac_114','fac_186','fac_210','zer_20'
- Une variance très faible pour les variables 'pix_1', 'pix_15','pix_22', 'pix_23', 'pix_24','pix_91','pix_105','pix_106', 'pix_121','pix_136','pix_211','pix_226','pix_240'.
- Une distribution très asymétrique des variables 'zer_1','zer_2','zer_8','zer_14', 'zer_30', 'zer_32','zer_39','zer_44'.

Suite à cette analyse, nous allons :

- Traiter les valeurs extrêmes constatées pour les variables fac_* et 'zer_20',

- Laisser les autres valeurs telles qu'elles sont, car et vu la distribution particulière des variables, ces valeurs ne sont pas considérés comme extrêmes.

2.3 Traitement des valeurs extrêmes

En couplant la visualisation des histogrammes avec les box-plot et les tests analytiques des IQR (écart interquartile), nous avons constaté que la valeur 3 de l'IQR représente un bon seuil de coupure pour les valeurs aberrantes.

En effet, nous prenons en pratique la valeur 1,5 pour les valeurs qu'on appelle « soft outliers » et 3 pour les « extreme outliers ».

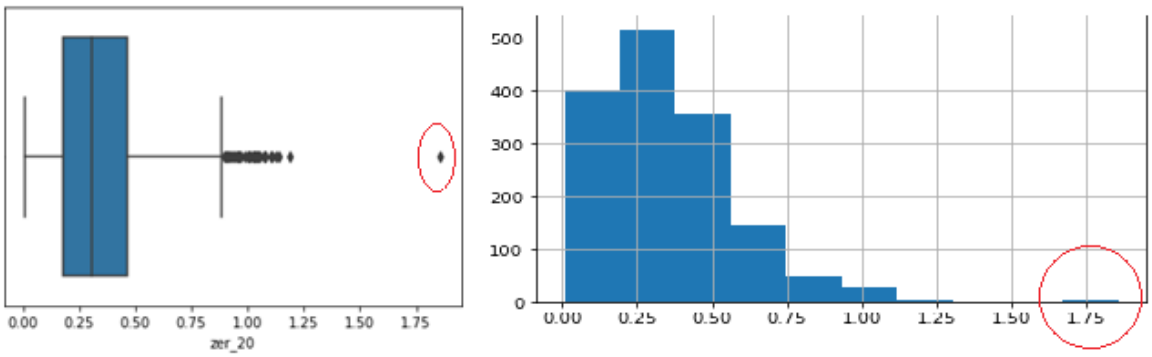


Figure 5: Exemple de valeur aberrante à traiter

Avec cette méthode, nous avons détecté 25 points extrêmes, ce qui représente 1,6% des données. Plusieurs méthodes de traitement sont possibles : winsorisation, transformation des données ou la suppression…
Vu le nombre limité des données détectées, nous allons les supprimer de notre échantillon.
Nous avons aussi effectué une analyse de variance sur ces variables par classe, et nous avons détecté très peu de variances pour certaines variables comme le montre la figure 6 :

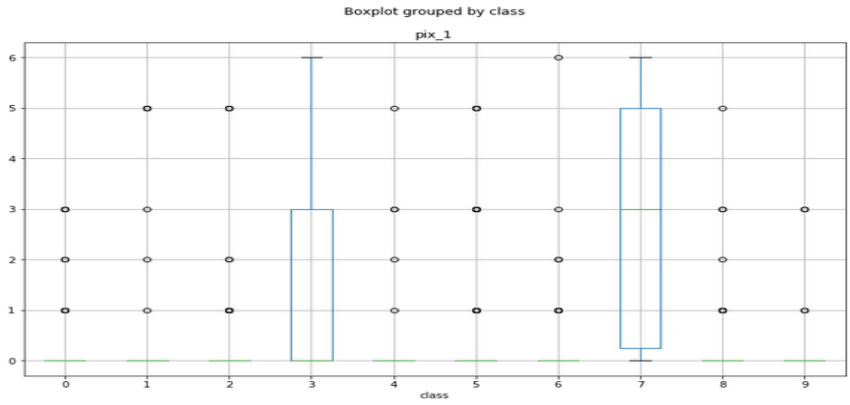


Figure 6: Analyse de variance pour la variable Pix_1

Néanmoins, en l'absence d'une connaissance métiers des variables et leurs méthodes de construction, nous avons préféré les garder dans un premier temps et faire une ACP qui pourrait mieux résumer leurs inerties.

2.4 Analyse bi variée

L'analyse bi-variée représente une étape importante dans le prétraitement des données.
En effet, nous avons des méthodes d'analyse qui sont sensibles aux corrélations entre variables, et des variables très corrélées peuvent impacter négativement les modèles (matrice de variance covariance non inversible, sur-apprentissage…)
La méthode SVM avec un kernel RBF en est un exemple car elle se base sur les distances entre les individus. Ainsi, si nous avons par exemple 11 attributs, mais l'un d'eux est répété 10 fois, ce dernier contribuera alors 10 fois plus à la distance que tout autre attribut, et le modèle final sera très impacté.
Vu le nombre élevé des variables, nous allons nous baser dans un premier temps sur une table analytique des corrélations comme le montre la table ci-dessous :

	fac_1	fac_2	fac_3	fac_4	fac_5	fac_6	fac_7	fac_8	fac_9	fac_10	...	zer_39	zer_40	zer_41	zer_4
fac_1	1.000000	0.486036	0.522956	0.488958	0.332718	0.093111	0.476426	0.338269	0.443714	0.374789	...	0.116709	0.488096	0.138524	-0.14733
fac_2	0.486036	1.000000	0.640521	0.368017	0.269650	0.305064	0.778466	0.661230	0.534958	0.483463	...	0.383811	0.424533	0.254603	0.23503
fac_3	0.522956	0.640521	1.000000	-0.090716	0.074830	0.270551	0.902198	0.338541	0.089727	0.395027	...	0.370992	0.485384	0.135476	0.07548
fac_4	0.488958	0.368017	-0.090716	1.000000	0.479532	0.140301	-0.005178	0.255648	0.788144	0.148614	...	-0.147298	0.133133	0.379069	0.19811
fac_5	0.332718	0.269650	0.074830	0.479532	1.000000	0.049362	0.106827	0.184453	0.463384	0.224198	...	-0.066110	0.170252	0.214960	0.28364
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	.
zer_44	0.113375	0.372646	0.364564	-0.143652	-0.065924	-0.154024	0.383008	0.377172	-0.043560	0.290069	...	0.993678	0.338192	-0.197439	-0.19799
zer_45	0.547291	0.434647	0.437930	0.189117	0.148833	0.183885	0.386998	0.256925	0.144586	0.247446	...	0.329943	0.914024	-0.146538	-0.21084
zer_46	-0.157926	0.164368	0.043638	0.168153	0.273480	0.630116	0.131107	0.087690	0.347631	-0.076402	...	-0.227567	-0.216887	0.252825	0.98884
zer_47	-0.671138	-0.357468	-0.431273	-0.518341	-0.341388	-0.343318	-0.378797	-0.030987	-0.494589	-0.164488	...	0.168857	-0.206649	-0.202005	-0.04258
cla															

Tableau 1: Table de corrélation entre les variables

Nous remarquons que les composantes corrélées sont celles qui proviennent du même groupe, donc nous allons par la suite construire une table de corrélation par groupe de variable :

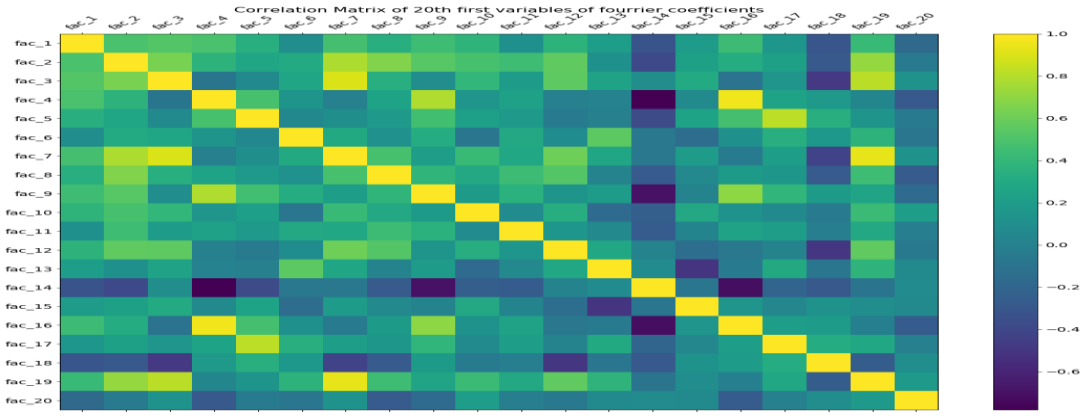


Figure 7: Table de corrélation des variables

Nous remarquons que certaines variables sont très corrélées entre elle comme le montre la matrice de corrélation sur la table 2 :

	fac_3
fac_7	0.9021980487234258
fac_39	0.9041646449468941
fac_51	0.902732380548773
fac_63	0.9583839123764822
fac_111	0.9942954143853516
fac_123	0.9872708638390028
fac_135	0.9943008682450756
fac_147	0.982329397253698
fac_183	0.9960088257973853
fac_199	0.9122656238837294

Tableau 2: Table de corrélation pour la variable fac_3

Nous allons par la suite supprimer les composantes très fortement corrélé, en fixant un seuil de corrélation linéaire de 0.95 ➔ 81 variables supprimées.

2.5 Analyse AFM des données

L'Analyse Factorielle Multiple (Escofier and Pagès, 1990, 1994) apporte une solution très satisfaisante au problème de l'équilibre des groupes. Elle pourrait être vue comme une ACP dans laquelle l'influence des groupes de variables est équilibrée. C'est dans cet esprit que nous effectuons l'analyse de nos données.
Ci-dessous les étapes clés pour conduire une analyse AFM :

- Faire une analyse ACP de chaque groupe séparément,
- Equilibré chaque groupe en divisant toutes ses variables par la plus grande valeur propre.
- Faire une ACP globale sur l'ensemble des variables équilibrées.

Pour faire l'exercice, nous allons nous baser sur un module python, assez complet, pour les analyses factorielles : PRINCE[1]
Une revue du code a été effectuée pour vérifier que l'implémentation technique de l'AFM dans le module est bien conforme à la théorie.
Lors de la construction de la méthode, nous devons choisir le nombre de composantes principales à retenir suite à cette analyse.
Etant donné que nous n'avons pas à priori cette information, nous allons effectuer cette analyse deux fois : la première fois en fixant le nombre de composantes assez élevé pour avoir une information sur la répartition d'inertie entre les composantes, et une deuxième fois en fixant le nombre de composante qui sera déterminé via la règle de Kaiser.

Suite à la première analyse, nous remarquons que l'inertie est répartie entre plusieurs composantes, et que la valeur des 8 premières valeurs propres est supérieure à 0,06 (I/P car l'implémentation de l'ACP dans le module PRINCE n'est pas normée).
Nous allons garder les 91 premières valeurs propres pour réduire la dimension des données.
Nous remarquons aussi que les deux premières composantes résument uniquement 19% de l'inertie globale.
Ceci indique que la représentation graphique sur un plan factorielle serait partielle, mais elle permet néanmoins de visualiser la répartition des différentes classes sur le plan (cf figure 7).
A travers cette présentation factorielle, nous constatons que :

- Hormis les classes 5 et 9 au centre qui sont superposées, les autres sont moyennement séparées.
- Le premier axe factoriel est caractérisé les groupes variables 4 et 2 (fou_* et mor_*).

[1] https://github.com/MaxHalford/prince

- Le deuxième axe factoriel est caractérisé par le groupe de variable 3 (kar_*).
- Le premier axe caractérise les classes 8 et 7
- Le deuxième axe caractérise les classes 4,6 et 1.

La présence de quelques points aberrants, non détectés lors de la première analyse. Vu la dimension réduite et le peu d'inertie, nous allons les laisser dans un premier temps et valider, s'il le faut, un autre modèles sans ces données.

Figure 8: Répartition des différentes classes sur le plan factoriel

Figure 9: Contribution des groupes de variables aux axes factoriels

A travers cette analyse, nous avons pu synthétiser les données et visualiser sur un plan factoriel les différentes classes.
Nous avons pu aussi à travers toute cette analyse exploratoire, mettre en évidence les différentes transformations, celles-ci seront effectuées uniquement sur les données d'apprentissage.

2.1 Classification des individus via une méthode géométrique non supervisée

Afin de vérifier si les différentes classes peuvent être reconstruites d'une manière automatique, nous allons implémenter la méthode K-means.
K-Means est un algorithme d'apprentissage automatique non supervisé qui regroupe les données en k clusters (k entier strictement positif). Le nombre de clusters est défini par l'utilisateur et l'algorithme essaiera de regrouper les données même si ce nombre n'est pas optimal pour le cas spécifique.
Afin de trouver le K optimal, nous allons nous appuyer sur la méthode du coude, en exécutant un clustering k-means pour une gamme de clusters k (de 3 à 15) et pour chaque valeur, nous calculons la somme des distances au carré de chaque point à son centre assigné (distorsions).
Nous allons effectuer cette classification en deux étapes :

- En utilisant un seul groupe de variables, en l'occurrence, le groupe 1 qui caractérise le premier axe factoriel.
- En utilisant l'ensemble des individus.

Classification avec le groupe 1

Le groupe 1 est constitué des facteurs de transformation de fourrier.

Nous allons nous baser sur ces variables pour faire une classification des différents groupes.

Trouver le nombre de groupe optimal

Afin de trouver le nombre de K optimal, nous allons visualiser la variation de la distorsion des différents groupes avec K :

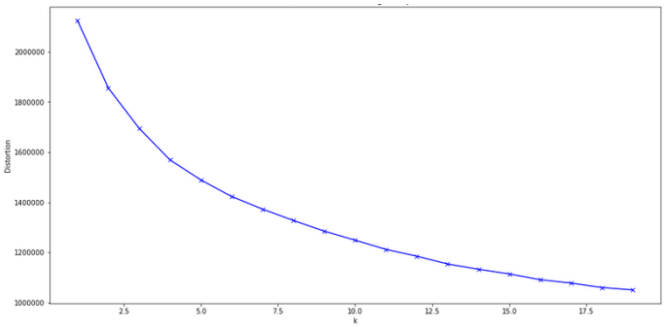


Figure 10: Graphe de distorsion en fonction de K

Nous remarquons que la valeur optimale se situe entre 4 et 5.
En prenons 5 classes, nous aurons 5 classes de moins que celle que nous avons dans notre jeu de donnée.
Une visualisation des classes trouvées permet, comme le montre la figure xxx, de constater que seule la classe 9 est séparée.

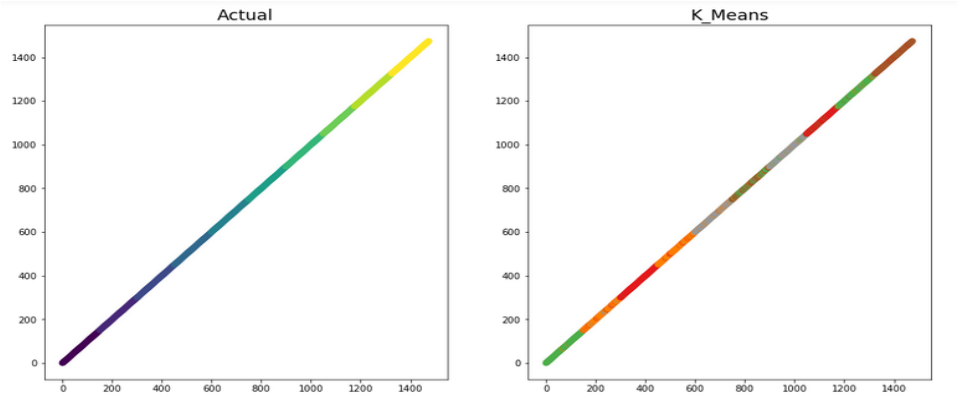


Figure 11: Classes actuelles vs représentées avec K=5

En passant à K=10, nous remarquons que les classes 4,6 et 7 contiennent beaucoup de points mal classés.

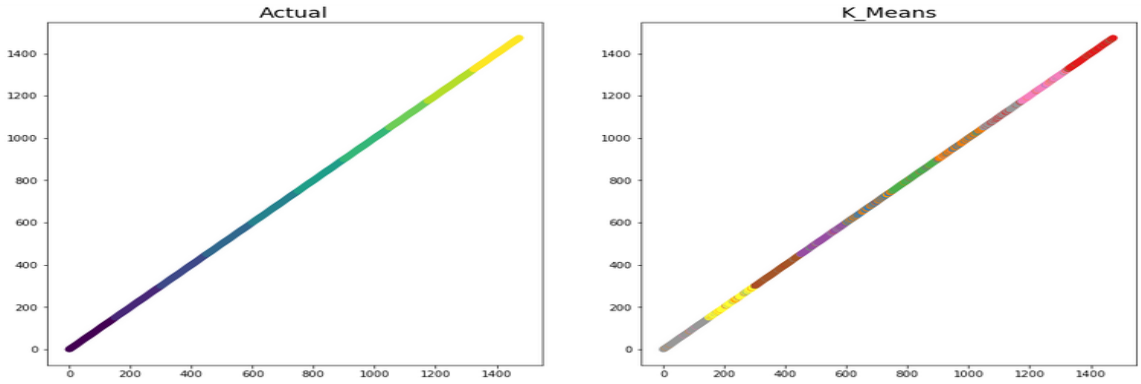


Figure 12: Classes actuelles vs représentées avec K=10

Nous allons par la suite englober toutes les variables pour étudier si la classification pourrait s'améliorer.

Classification avec l'ensemble des variables

En prenant en compte toutes les variables, nous remarquons que le niveau de coupure reste le même qu'auparavant : entre 4 et 5.

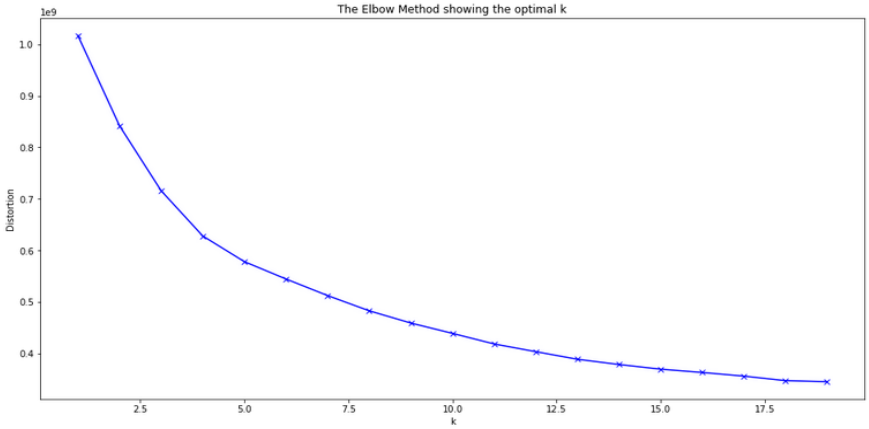


Figure 12: Graphe de distorsion en fonction de K

En faisant une analyse K-means avec 10 classes, nous remarquons que les classes 4,5,7 et 8 contiennent beaucoup de points mal classés. L'amélioration apportée par l'ajout des autres groupes de variables n'est pas très significative :

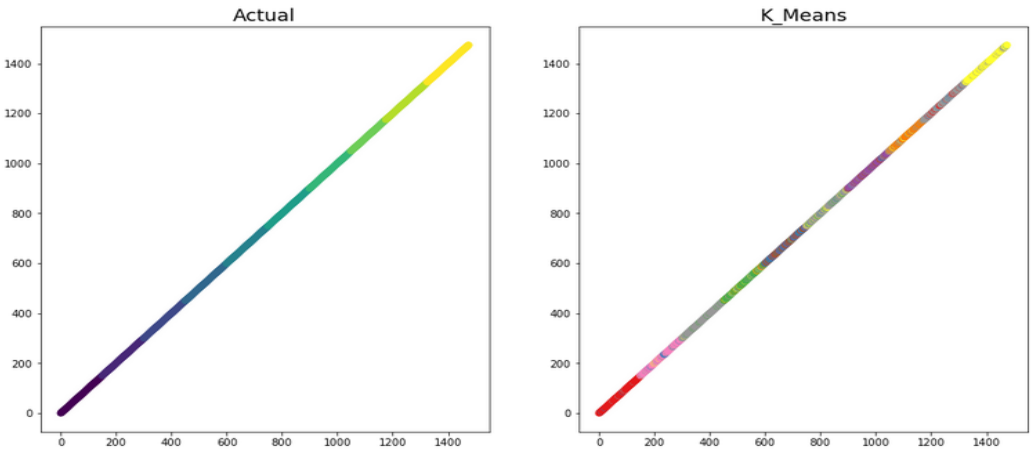


Figure 13: Classes actuelles vs représentées avec K=10

Suite à cette analyse non supervisée, nous avons pu conclure qu'une séparation en 10 classes n'était pas possible à travers le modèle K-means.

Ceci prouve aussi que se baser uniquement sur les distances entre les points, n'est pas un critère suffisant pour classifier notre jeu de données en 10 classes.

3. Elaboration d'un modèle statistique supervisé

3.1 Modèle SVM

Avant de commencer notre modélisation, nous allons tout d'abord énumérer les méthodes et les hyper-paramètres qu'on souhaite tester pour cette famille de méthode :

- SVM avec séparation linéaire.
- SVM avec séparation RBF avec une variation du paramètre de régularisation C.
- SVM avec séparation polynomiale avec une variation du degré du polynôme.

D'une façon plus globale, la stratégie d'étude des différents modèles qu'on développera par la suite suivra les schémas ci-dessous[2] :

[2] https://scikit-learn.org/stable/modules/cross_validation.html

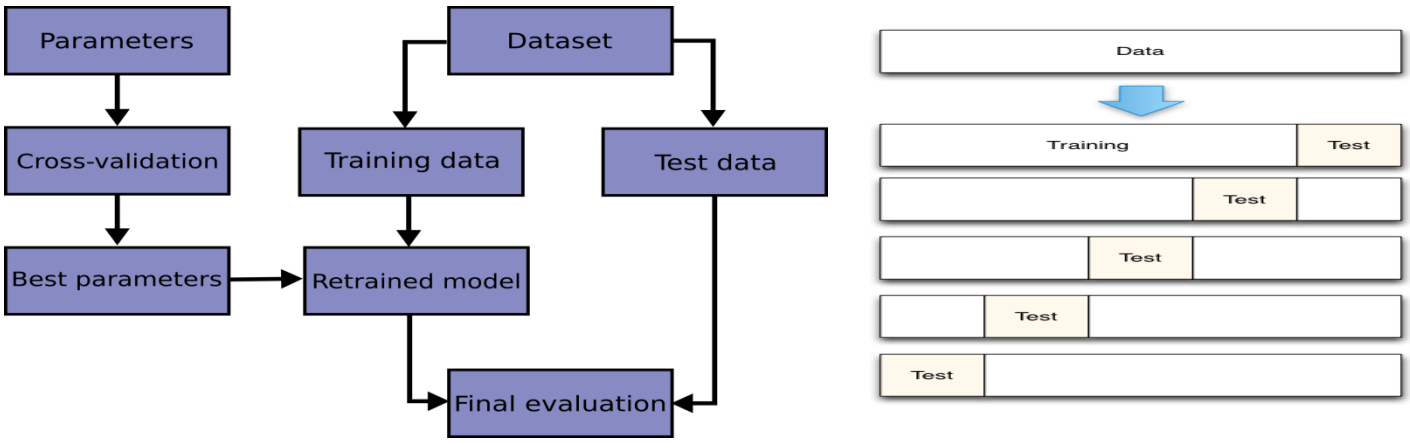


Figure 14: Stratégie de développement des modèles statistiques

Ci-dessous les différents résultats obtenus :

Méthode	***Paramètre***	***score de validation croisée (5 folder)***
SVM linéaire	NA	98,6%
SVM RBF	C varie dans [0.1,1,2,5,10,100]	98,2%
SVM polynomial	Degré dans [1,2,3,5,10]	98,3%

Suite à ces différents tests, nous allons retenir le modèle SVM linéaire avec lequel on obtient le plus grand score via les tests de validations

3.2 Modèle Random Forest

La méthode des forêts aléatoires (Randomforest) consiste à sélectionner de façon aléatoire un certain nombre d'arbres de décision pour les rendre plus indépendants en réduisant la variance de l'estimateur.

En testant plusieurs nombre d'estimateurs, nous obtenons les résultats suivants :

Paramètre	Score
n_estimators=5	***93,63%***
n_estimators=10	***94,15%***
n_estimators=20	***94,93%***
n_estimators=30	***94,89%***
n_estimators=40	***94,97%***
n_estimators=50	***95,09%***
n_estimators=100	***95,08%***

Nous avons un score de 95% qui maximum avec n_estimators=50.

3.3 Modèle Réseaux de neurones

Les réseaux de neurones souvent considérés comme des boites noires (à cause de la difficulté d'interprétation). Ils permettent d'effectuer de la classification mais aussi de la régression.
La fonction d'activation utilisée par librairie python est la fonction ReLU.
Nous allons modifier par la suite le nombre de couches qui représentent le paramètre à optimiser.
Ci-dessous les différents scores obtenus en fonction du nombre des couches :

Paramètre	Score
n_hidden_layer_sizes=5	0.8793766302038897
n_hidden_layer_sizes=10	0.95496498803918
n_hidden_layer_sizes=20	0.970227147774749
n_hidden_layer_sizes=30	0.9711510071408801
n_hidden_layer_sizes=40	0.9778045539849852
n_hidden_layer_sizes=50	0.9771140449006401
n_hidden_layer_sizes=100	0.9779396688656433
n_hidden_layer_sizes=200	0.9805044232654291

Sans surprise, le modèle avec le nombre le plus grand des couches obtient les meilleures performances. Nous allons le garder pour le comparer avec les autres modèles.

3.4 Bilan des trois méthodes

En utilisant les données de tests gardées auparavant, nous avons conclu que le modèle SVM permet de donner légèrement des meilleurs résultats que les réseaux de neurones. Les RF sont classés loin derrière.

Modèle	***Score***
Random Forest	69%
SVM linéaire	99%
Réseaux de neurones	98,7%

Nous remarquons que les modèles SVM et réseaux de neurones ont très peu de variances contrairement au modèle RF qui passer de 95% pour le jeu d'apprentissage à 69% pour le jeu de données de test.

4. Conclusion

A travers ce projet, nous avons travaillé sur un jeu de données avec la particularité qu'il est issu de plusieurs tableaux.
Nous avons utilisé les différentes techniques apprises lors de notre cours pour comprendre le jeu de données à travers ses différentes façades et résumer l'information à travers des représentations factorielles.
Tout ce travail nous a été utile dans le travail prédictif, pour le choix de la méthode et aussi le choix des paramètres.
En effet, le fait que l'inertie soit dispersée à travers un large nombre de composantes comme il l'a été démontré à travers l'analyse AFM, nous a poussés à penser directement à la méthode SVM, qui est une méthode très performante dans un contexte de très grande dimension.
Cependant, comparer des méthodes nécessite de prendre en considération d'autres paramètres outre que le taux de bon classement tel que la rapidité, la robustesse et l'interprétabilité.

Une suite intéressante au projet pourrait être de creuser les valeurs mal classées dans notre jeu d'apprentissage et essayer de retravailler la phase de prétraitement pour améliorer notre score de prédiction.
On pourrait aussi utiliser un modèle basé sur « Ensemble learning» qui permettrait de combiner plusieurs algorithmes et utiliser par la suite un vote majoritaire.