Université de Montréal

Coefficient de Pearson à décalage simple

Par
Alexandre Chartrand,
Lauranne Deaudelin,
Amélie Lacombe Robillard, et
Stéphane Verville-Vohl

B. Sc. Informatique
Département d'informatique et de recherche opérationnelle

Travail présenté à Alain Tapp
Dans le cadre du cours IFT 3700
Sciences des données

Novembre 2018

Introduction

Ce travail consiste à proposer une notion de similarité originale augmentant potentiellement la performance de divers algorithmes d'analyse de données sur l'ensemble de données MNIST. De ce fait, la distance permettant la comparaison entre la nouvelle notion de similarité développée est la distance euclidienne. Cette distance sera la distance contrôle tout au long de l'analyse. La notion de similarité développée pour l'analyse présente a été nommée la distance corrélation de Pearson avec décalage simple. La corrélation de Pearson en soi est intéressante dans le cas de MNIST, car elle est un indice de proportionnalité tandis que la distance euclidienne est un indice spatial; plus d'information sur le sujet suivra plus loin.

Pour différentes méthodes d'analyse d'information, soient la classification, le partitionnement et la réduction de dimensions, l'analyse a été séparée en plusieurs phases. Premièrement, il est nécessaire d'implémenter les différents algorithmes d'analyse permettant de comparer les différentes notions de similarité entre elles. Ensuite, des tests et des métriques de performance ont été mises en place afin d'obtenir des notions de comparaisons, sur lesquelles les analystes peuvent baser leur jugement. Afin de trouver une notion de similarité offrant potentiellement une amélioration des résultats, plusieurs métriques ont été testées. Au final, la décision a été d'implémenter une méthode qui prend en compte le décalage des chiffres dans l'espace combinée avec la corrélation de Pearson. Les résultats obtenus ont ensuite été analysés afin d'observer les forces et faiblesses de la notion de distance choisie. Ce rapport traite respectivement des catégories de méthodes utilisées pour faire l'évaluation de la nouvelle distance, soit des méthodes de partitionnement (k-médoïde, partitionnement binaire), des méthodes de réduction de dimensionnalité (PCOA, Isomap), et d'une méthode classification (KNN). Chaque sous-section parle de l'implémentation, des métriques d'évaluation utilisées, des résultats obtenus et conclusions pouvant être tirées des résultats obtenus.

Manipulation des données

L'ensemble de données MNIST comprend 70000 images de 28 pixels par 28 pixels, encodées sous forme de vecteurs de 784 entrées représentants la teinte d'un pixel. L'ensemble de données est divisé en 2 sous-ensembles : le premier de 60000 images constituant l'ensemble d'entraînement, et le deuxième de 10000 images étant utilisée comme ensemble de test. Cependant, dû au temps requis pour utiliser l'ensemble d'entraînement en son entier (aggravé par la complexité algorithmique de notre mesure de similarité), seulement 2000 données ont été considérées pour cette analyse. L'ensemble de données utilisé pour la prédiction correspond à 15% de la taille de l'ensemble d'entraînement.

La stratégie principale adoptée pour analyser efficacement les différents algorithmes est de calculer initialement la distance entre chaque point, en fonction de la notion de similarité ou de la distance choisie, et de conserver chaque distance dans une matrice de dissimilarité, pouvant être ensuite passée en argument à chaque méthode d'analyse. De cette manière, les distances pour l'entraînement peuvent n'être calculées qu'une seule fois. Dû à la dimensionnalité très élevée des données, le calcul de la matrice de dissimilarité doit faire n^2 produit scalaires, n étant le nombre de données utilisées. Cependant, le coefficient de Pearson avec décalage simple, ainsi que la distance euclidienne étant des notions de similarité symétriques, il suffit de calculer la moitié des combinaisons de 1 à 1 point possible et d'assigner la valeur du calcul faite à l'élément transposé de la matrice en même temps.

Toutefois, il aurait pu être intéressant, afin de réduire le temps de calcul des matrices de dissimilarités, d'appliquer un prétraitement aux données de l'ensemble MNIST. Premièrement, en considérant que chaque pixel des images ne peut être que blanc ou noir, en fonction d'une valeur frontière pour les teintes de gris, et de convertir les pixels en noir ou blanc (maintenant de valeurs 0 ou 1). Par la suite, chaque vecteur de 784 dimensions aurait été converti en un vecteur potentiellement plus petit, contenant seulement l'index de chaque pixel noir. Ainsi, la comparaison pixel à pixel aurait prise moins de temps, au coût d'une baisse de précision. Cependant, une contrainte de temps nous oblige à ne pas utiliser ce prétraitement, qui nous aurait potentiellement permis d'utiliser une plus grande partie de l'ensemble d'entraînement.

Explication de la notion de similarité et son implémentation.

La notion de similarité développée est appelée la distance de corrélation de Pearson avec décalage simple. La corrélation de Pearson est aussi connue sous le nom de corrélation linéaire. Comme l'indique son nom, il s'agit d'une mesure de la relation linéaire entre deux données. Dans le cas de MNIST, elle indique si le vecteur d'une image est proportionnel au vecteur d'une autre image. La corrélation entre deux données peut prendre une valeur entre de -1 à 1; par exemple, -1 indiquant une forte dissimilarité linéaire entre deux images et 1 indiquant le contraire, une forte similarité linéaire. Elle est une bonne mesure pour MNIST par le fait qu'elle normalise les deux vecteurs images et qu'elle s'assure d'obtenir vraiment une statistique définitive entre la ressemblance et la différence des deux vecteurs par son quotient. Si on la compare avec la distance euclidienne, la corrélation est une valeur beaucoup plus significative, expressive et facile d'interprétation pour la similarité. Cependant, prendre la corrélation elle-même ne fait pas d'elle une distance et cela rentre en direct conflit avec les algorithmes requis pour ce travail. C'est pourquoi l'implémentation offerte par les librairies de python soustrait la corrélation à 1, respectant ainsi les propriétés d'une distance.

Si on prend cette mesure sans modification, elle est légèrement meilleure ou équivalente à celle la distance euclidienne. On explique cette meilleure performance par les points relevés plus haut,, mais elle ne la surpasse pas pour la même raison que celle-ci n'est pas tout à fait adaptée à MNIST : elle ne tient pas compte que le chiffre des images peut être légèrement décalé du centre de l'image. Bien qu'on puisse faire l'hypothèse que les chiffres sont en général bien centrés, il n'en reste pas moins qu'une variation existe et peut engendrer de grosses différences dans la classification et le clustering. Pour remédier à cette potentielle faiblesse, on considère tester des translations entre les deux images pour prendre la plus petite distance entre elles. Dans la présente implémentation, on investigue chaque translation d'un pixel de haut en bas et de droite à gauche et on cherche s'il est possible d'avoir une meilleure similarité grâce à une de ces différentes superpositions d'images. Ceci est également utile si nos deux mêmes chiffres sont écrits de manière légèrement différente (déformation, variation de taille). Il a été considéré de prendre un plus grand nombre de pixels pour la translation, mais il était possible de voir avec un sous ensemble des données qu'augmenter le nombre de pixels ne permettait pas d'améliorer les performances et même cela pouvait les dégrader. On peut supposer qu'à chercher un trop gros décalage, on tombe sur une superposition incongrue (par exemple, d'un 1 avec la barre verticale d'un 7 ou d'un 9) et que cela affecte la similitude entre ces chiffres. Ce choix engendre également une perte d'efficacité due au nombre supérieur de distances à calculer (plus élevé plus le nombre de pixels de translation considéré est élevé). Pour une translation d'un pixel dans quatre directions, on effectue 4 calculs de distance supplémentaire (pour un total de 5). En revanche, le gain en performance excuse cette perte. Il est supposé que cette mesure serait utilisée sur des systèmes suffisamment puissant pour amortir la baisse d'efficacité. Les translations diagonales ont également été initialement

considérées mais ont été rejetés car le gain de performance n'était pas significatif pour le coût en efficacité.

Méthodes de partitionnement et performances (k-Medoid, Partitionnement binaire hiérarchique)

k-Médoide

L'idée de base de l'implémentation consiste à utiliser les fonctions offertes dans la librairie kmedoids de pyclustering pour Python afin de pouvoir entraîner le modèle aisément. Une méthode de plus a été développée cependant afin de pouvoir prédire les classes d'une partie de l'ensemble de données de test.

Il a été décidé d'utiliser des méthodes supervisées pour faire l'évaluation de différentes propriétés des partitions produites en fonction de la notion de similarité utilisée, pour tirer avantage du fait que nous ayons accès aux vraies classes de chaque image. Ainsi, les métriques utilisées afin de comparer la corrélation de Pearson avec décalage simple à la distance euclidienne pour l'algorithme de k-médoïde sont le pourcentage de prédictions correctes, l'index Rand corrigé, l'index d'homogénéité et l'index de complétude des partitions.

Bien que le pouvoir de prédiction ne soit pas la tâche principale des méthodes de partitionnement, il reste intéressant de voir si le changement de la notion de similarité change de façon drastique la faculté de prédiction d'un algorithme de partitionnement de données. L'index ajusté de Rand, quant à lui est une métrique ajustée du pouvoir de prédiction. Elle est en effet spécialisée pour les méthodes de partitionnement de données pour évaluer la similarité entre les valeurs prédites et les valeurs cibles en fonction du partitionnement fait. Cette mesure indiquera ainsi si la distance développée donne des meilleurs résultats. De plus, il est intéressant de voir la différence des résultats entre la méthode naïve de calcul de pouvoir de prédiction et entre l'index ajusté de Rand. Les deux autres métriques sont plus spécifiques: l'index d'homogénéité vérifie si les points de chaque partition sont le plus de la même classe possible, tandis que la mesure de complétude vérifie si tous les points d'une classe sont contenus dans les mêmes partitions.

Cependant, ces méthodes utilisent les étiquettes de prédictions des données et les comparent avec les vraies étiquettes pour ainsi calculer leurs index. Dans l'algorithme de k-medoid présent, rien ne garantit que l'étiquette donnée soit bien le chiffre que l'image représente; il a donc fallu mettre en place un procédé pour permettre des prédictions, en cherchant la correspondance entre les points d'une partition et leurs étiquettes respectives. En résumé, pour chaque donnée test, on prédit son chiffre en donnant la valeur *target* du centroïde qui est le plus près de la donnée.

Résultat Euclidien

Prediction score	0.5333333333333333
Adjusted Rand Index	0.31044616617686344
Homogeneity	0.5236123191238723
Cluster Completeness	0.7715090312822468

Résultat Distance Créée

Prediction score	0.4666666666666667
Adjusted Rand Index	0.4025667004390409
Homogeneity	0.617930727319
Cluster Completeness	0.7483084431196855

Pour toutes ces métriques, l'objectif est que leur valeur soit la plus proche possible de 1. Une valeur près de 1 représente que la caractéristique observée est particulièrement forte. Ainsi, il est possible d'observer que l'index de Rand ajustée obtenue pour la corrélation de Pearson avec décalage simple est plus élevée que l'indice de Rand obtenue en fonction de la distance euclidienne. Toutefois, le pouvoir de prédiction naïvement implémenté semble avoir des résultats favorisant la distance euclidienne comme notion de similarité. L'indice de Rand ajustée étant habituellement choisie pour quantifier la similarité de partitions, il est juste de penser que la notion directe de pouvoir de prédiction n'aurait pas sa place pour l'évaluation de performance d'algorithmes de partitionnement de données, ou qu'un détail ait été oublié d'être pris en considération lors de l'implémentation de la fonction d'évaluation du pouvoir de prédiction. En ce qui concerne les indices d'homogénéité et de complétude, la corrélation de Pearson avec décalage simple semble avoir partitionné les données de façon plus homogènes, tandis que la distance euclidienne semble avoir partitionné les données de manière plus complète. Un rappel de la notion d'homogénéité est que les partitions comprennent le plus possible une seule classe. La notion de complétude, quant à elle, indique si les points d'une classe font partie de la même partition.

Arbres de partitionnement binaires

La classe utilisée pour l'algorithme de partitionnement hiérarchique binaire est AgglomerativeClustering de SciKit-Learn pour Python. Pour cet algorithme, l'équipe a décidé de n'utiliser que des métriques d'évaluation non-supervisées, ne nécessitant aucune prédiction. Après-tout, c'est le partitionnement lui-même et non le pouvoir de prédiction qui est important pour les algorithmes de partitionnement. Or, la métrique utilisée est le coefficient de silhouette. Le coefficient de silhouette indique comment le *cluster* est bien définie, c'est-à-dire comment une donnée est similaire à son propre *cluster* en comparaison avec les autres *clusters.* Elle est obtenue par la cohésion de chaque donnée et de la séparation de celle-ci aux autres clusters. Plus le coefficient est près de 1 ou -1, plus il indique si les données du *cluster* sont respectivement effectivement dans le bon *cluster* ou pas. Puis, un coefficient qui est près de 0 est signe de l'ambiguïté que les données ont; elles peuvent être autant dans un *cluster* ou un autre.

Ainsi, pour 2000 données, les résultats obtenues sont d'environ 0.064 pour la distance euclidienne et -0.0378 pour la mesure de similarité créée. Clairement, il n'est pas possible de conclure autant pour la distance euclidienne ou l'autre distance que l'algorithme d'arbres de partitionnement binaires a offert de bons *clusters;* c'est deux résultats sont trop près de zéro. Toutefois, on peut remarquer qu'il semble y avoir une meilleure performance pour la distance euclidienne, mais la différence est trop mince (environ 0.1018) pour être significatif et permettre d'affirmer que ce partitionnement est mieux. Pour tenter d'améliorer les scores, il aurait été bénéfique d'augmenter la taille de l'ensemble d'entraînement pour permettre une meilleure définition des *clusters*. De plus, le coefficient de silhouette n'est peut-être pas le mieux adapter pour comparer deux métriques différentes, car il est souvent utilisé pour comparer la même distance avec différents nombres de clusters; il est donc possible qu'il ne soit pas cohérent d'utiliser deux métriques de distances différentes pour comparer des résultats de coefficient de silhouette. Une différente mesure de performance aurait pu être appliqué comme tenter d'utiliser des méthodes supervisées comme avec k-médoïde.

Euclidean silhouette score:	0.06350857914892953
Custom silhouette score:	-0.03771464839835632

À noter pour le partitionnement: certains conflits sont arrivés avec la définition du chiffre de groupe vu la différence qu'on peut écrire le même nombre, par exemple le 7 (avec ou sans barre), car la notion de similarité et la distance euclidienne ne couvre pas la possibilité de différence entre ces chiffres. Alors, pour avoir une meilleure précision, il a fallu augmenter le nombre de groupe, mais une notion de similarité mieux adaptée aurait pu faire en sorte que le nombre de groupe soit plus intuitif. De plus, une autre piste qui aurait pu être considérée pour

k-médoïde est de réduire le seuil de tolérance sur la différence entre deux partitionnements; ce qu'on utilise généralement comme critère d'arrêt pour cet algorithme, augmentant par conséquent la précision des résultats obtenus, au coût d'un temps d'exécution drastiquement plus élevé.

Méthodes de réductions de dimensionnalité

PCoA

Pour l'algorithme PCoA, la classe MDS (Multidimensional Scaling) de la librairie Scikit-Learn pour Python a été utilisée. La comparaison des notions de similarité, pour les méthodes de réduction de dimensionnalité, a tendance à être difficile à quantifier. Ainsi, une partie de l'interprétation est faite de manière qualitative, plutôt que quantitative. De plus, l'analyse qualitative peut se faire car les résultats sont réduits à deux dimensions, contrairement aux 784 dimension des données pour la classification et pour les méthodes de partitionnement. Cependant, une métrique quantifiable a pu être trouvée et utilisée pour comparer l'expressivité des réductions de dimensionnalité,

En effet, il est possible de comparer l'algorithme PCoA qui utilise la distance euclidienne avec celui qui utilise la distance de similarité créée grâce à la métrique nommé stress. La mesure stress est utilisée pour exprimer à quel point les données du jeu d'apprentissage particulier est bien représenté par le modèle choisi.

$$stress = \frac{\sum (d_{ij} - d_{predicted})^2}{\sum d_{ij}^2}$$

La valeur de stress est calculée en effectuant la somme de la différence entre chacune des distances de la matrice de dissimilarité données à l'algorithme MDS et la distance prédite après l'application des transformations engendrées par MDS. Cette somme des erreurs de moindre carrées est ensuite divisée par la somme des distances de la matrice de dissimilarité, et on y retient finalement la racine carrée du résultat.

Avec ce calcul, il est possible de déterminer si le modèle de PCoA avec MDS représente bien les données qu'on lui a fourni.

Plus le résultat du stress est proche de 0, plus le modèle ressemble aux données présentées.

Le chapitre 435, de NCSS Statistical Software[1] présente la table de valeur suivante pour qualifier la ressemblance des données.

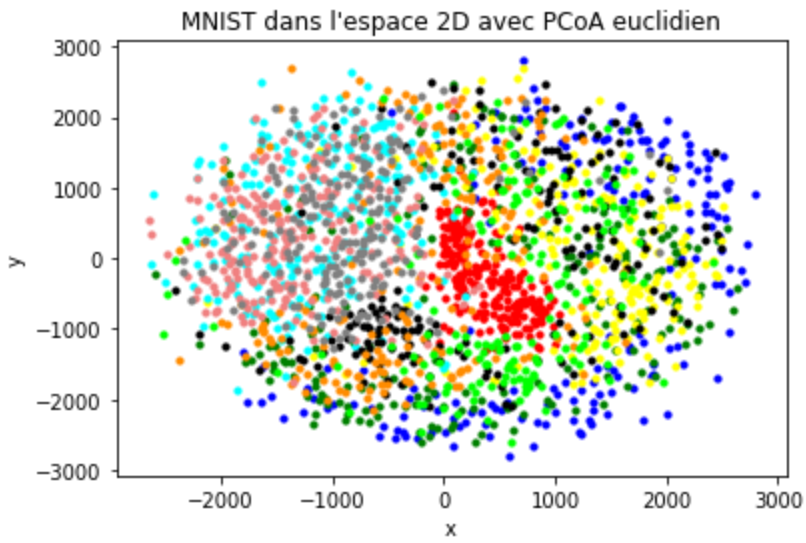
Stress	Goodness-of-fit
0.200	poor
0.100	fair
0.050	good
0.025	excellent
0.000	perfect

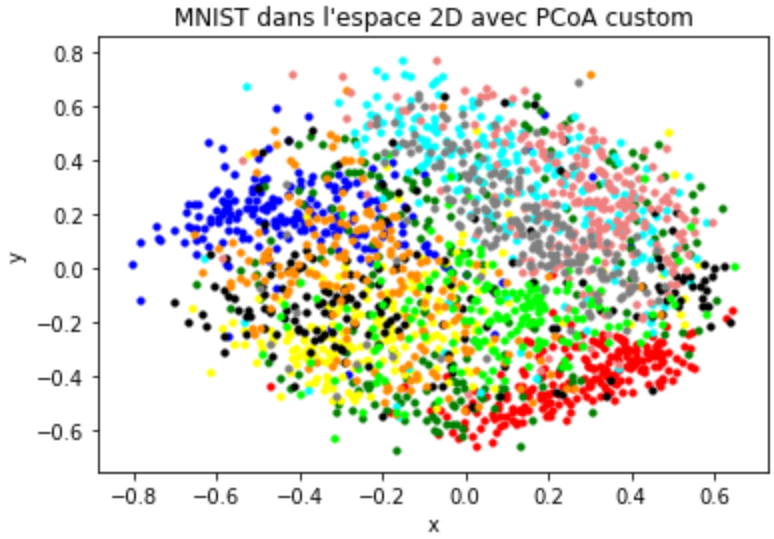
L'algorithme PCoA, avec 2000 données, a généré un stress d'environ 0.25 en utilisant la distance euclidienne et autour de 0.229 en utilisant la mesure de similarité créée. On voit donc que, malgré que l'algorithme PCoA ne puisse pas être considéré comme un très bon modèle pour représenter ces 2000 données, il n'en demeure pas moins que le calcul avec la mesure de similarité représente mieux les données que le calcul utilisant la distance euclidienne.

Résultats obtenus

Goodness-of-Fit for PCoA with euclidean distance	0.25362705293132665
Goodness-of-Fit for PCoA with custom similarity measure	0.22974606738757558

[1] https://ncss-wpengine.netdna-ssl.com/wp-content/themes/ncss/pdf/Procedures/NCSS/Multidimensional_Scaling.pdf





Légende
0 : blue
1 : red
2 : green
3 : yellow
4 : cyan
5 : black
6 : darkorange
7 : lightcoral
8 : lime
9 : grey

Analyse des graphiques

On peut voir que l'algorithme PCoA avec la mesure de similarité implantée regroupe mieux certaines données entre elles que le même algorithme, mais qui utilise la distance euclidienne. Par exemple, le groupement des chiffres 7 (lightcoral) sont éparpillés en 2 sous-groupes dans PCoA custom plutôt qu'éparpillé partout dans moitié gauche du graphique (PCoA euclidien). On peut voir le même phénomène avec les chiffres 3 (jaune), qui sont beaucoup mieux regroupés ensembles que dans le graphique avec la distance euclidienne. Il n'en demeure pas moins que que les deux graphiques se ressemblent beaucoup.

-Nous aurions pu faire un clustering après avoir dim-redux

Un correctif à l'implémentation a dû s'appliquer et cela est survenu lorsque l'algorithme PCoA a été implémenté. Pour le bon fonctionnement de cet algorithme, il est nécessaire que la matrice de dissimilarité soit symétrique et à cette étape, une erreur d'exécution s'affichait mentionnant que la matrice de dissimilarité de la distance de corrélation de Pearson avec décalage simple ne l'était pas. Cela était dû à la façon dont les translations étaient appliquées aux images, qui ne produisait pas des résultats symétriques. En effet, les translations étaient appliquées sur une seule image de la paire (x,y) alors que l'autre image demeurait fixe. Lorsque l'on arrivait à la paire (y,x), les translations étaient appliquées à l'autre image, et cette recherche produisait ainsi des valeurs différentes. Pour corriger ce problème, les translations sont appliquées successivement aux deux images selon une direction donnée (alors que l'autre image reste fixe). Cette implémentation nécessite le même nombre de comparaisons et produit toujours des résultats symétriques.

Isomap

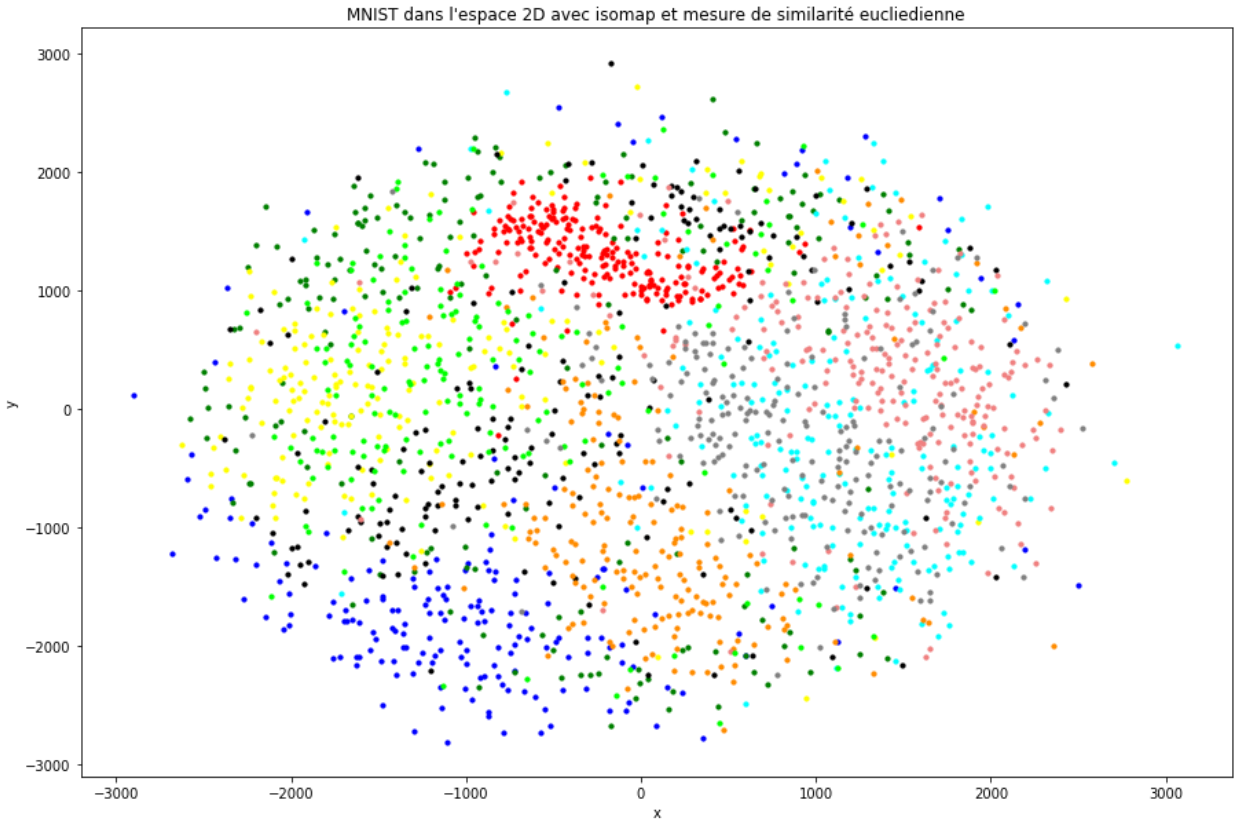
L'implémentation d'isomap dans la librairie Scikit-Learn ne permet pas de fournir en argument ni la matrice de dissimilarité, ni la métrique de distance elle-même à l'algorithme. Il a donc fallu implémenter isomap par nous-même afin de pouvoir tester la nouvelle métrique développée. Pour ce faire, une méthode permettant de ne considérer que les distances dans un certain rayon a dû être implémenté. De plus, il a fallu utiliser l'algorithme de floyd_warshall afin de déterminer une matrice de similarité qu'il a ensuite été possible de donner en argument à l'algorithme MDS. L'implémentation d'isomap fût très difficile, car le très peu de documentation en ligne à rendu le travail très archaïque. De plus, comme la librairie Scikit-Learn possède des méthodes de calcul pour isomap, il a été à tort de croire que c'était la bonne librairie à utiliser. Beaucoup de temps ont été passé à surcharger les fonctions de la librairie sans succès.

Ensuite, les mêmes métriques que pour PCoA ont été utilisées afin de pouvoir comparer la performance d'isomap en fonction des distances utilisées, soient l'évaluation qualitative (observation des *scatter plots* et interprétation subjective), et par la métrique de stress. Il a aussi été possible d'utiliser cette métrique pour isomap, car MDS est aussi utilisé pour calculer isomap, tout comme avec PCoA. Il a été possible d'observer un stress d'environ 0.25 pour l'algorithme Isomap utilisant la distance euclidienne et 0.229 pour le calcul avec la mesure de similarité développée. Donc, encore une fois, la même conclusion que pour PCoA s'applique : l'algorithme avec la mesure de similarité conçue performe mieux.

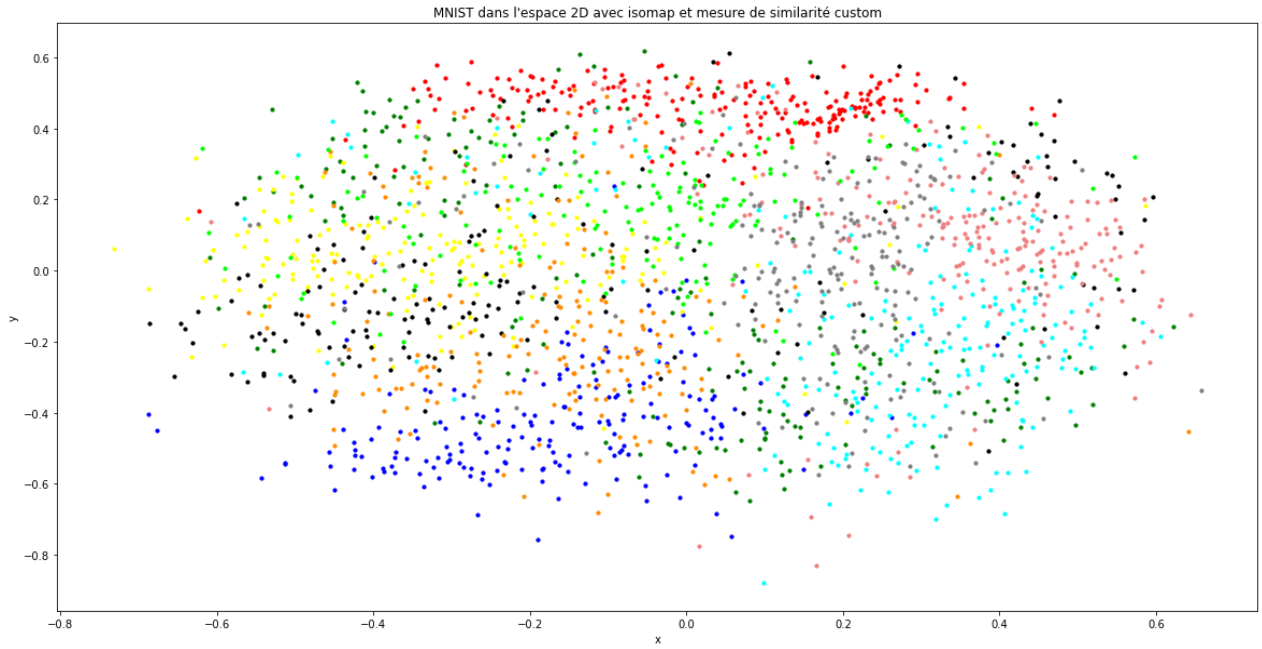
Un autre fait intéressant à relever est le fait qu'en général, l'algorithme PCoA performe aussi mal que l'algorithme Isomap, autant avec la mesure développée que la distance euclidienne.

Résultats obtenus

Goodness-of-Fit for ISOMAP with eucliedian distance	0.25418521948841627
Goodness-of-Fit for ISOMAP with custom similarity mesure	0.22955102821003437



MNIST dans l'espace 2D avec isomap et mesure de similarité euclidienne



MNIST dans l'espace 2D avec isomap et mesure de similarité custom

Légende
0 : blue
1 : red
2 : green
3 : yellow
4 : cyan
5 : black
6 : darkorange
7 : lightcoral
8 : lime
9 : grey

Analyse des graphiques

On peut apercevoir que certains regroupements sont plus évasés dans l'espace avec mesure de similarité custom, comme on peut le constater avec les chiffres 1. Mais on peut aussi voir que certains groupes sont beaucoup plus distinguables dans l'algorithme avec la mesure de similarité custom. Par exemple, le groupe des chiffres 4 (cyan) est moins mélangé avec les chiffres 7 (lightcoral) que dans Isomap avec mesure euclidienne.

Méthodes de classifications et performances

Pour k-Nearest Neighbors, la classe KNeighborsClassifier dans la librairie SciKit a été suffisante pour entraîner le classificateur avec la notion de similarité et la distance euclidienne par son initialisation et les méthodes qui l'accompagnent. Puis, Python offre une grande sélection de mesure de performance pour les classificateurs et la mesure de performance qu'il a été utilisé dans ce travail fut la précision des classements (*accuracy score*), qui évalue le nombre de points ayant bien été classés sur le nombre total de point de l'ensemble de test, et détermine le pourcentage de prédictions réussies.

Euclidean prediction score:	0.9
Custom prediction score:	0.9666666666666667

La seule métrique observée pour l'algorithme des k-plus-proches voisin a été le pourcentage de prédiction correctes faites sur des données que l'algorithme n'a jamais vu lors de l'entraînement. Ainsi, avec 2000 données d'entraînement il a été possible de voir une augmentation de 6.66% de pouvoir de prédiction en utilisant la corrélation de Pearson avec décalage simple plutôt que la distance euclidienne, ce qui semble être une augmentation considérable du pouvoir de prédiction grâce à cette nouvelle notion de similarité. Cependant, il serait possible qu'une augmentation de la taille de l'ensemble de données d'entraînement puisse permettre à l'algorithme des k-plus-proches voisins utilisant la distance euclidienne d'obtenir des résultats similaires à ceux de la notion de similarité étudiée, indiquant que le pouvoir de prédiction, en ayant un ensemble de données de bonne taille, ne serait pas affecté par le changement de la notion de similarité. Il serait cependant possible de dire, d'après les informations données, que l'utilisation de la corrélation de Pearson avec décalage simple pour l'algorithme des k-plus-proches voisins donnerait des résultats de prédictions plus favorables avec moins de données.

Conclusion

Pour conclure, le travail aura été de mettre en place les algorithmes demandés pour le partitionnement, la réduction de dimensionnalité et la classifications et puis, d'obtenir des mesures pour évaluer la performance de ceux-ci avec la notion de similarité développée. Par les résultats obtenues, il a été possible d'améliorer la quasi-totalité des algorithmes présentées avec la mesure créée, bien qu'habituellement de peu. Une faiblesse qui a été remarqué de la part de la distance de corrélation de Pearson avec décalage simple était principalement lors des algorithmes de partitionnement non-supervisé où elle semblait être peu performante. Elle a donné de relativement bonne performance dans les autres algorithmes en comparaison avec la distance euclidienne si on fait abstraction des résultats non concluants de la partition binaire. De plus, il est certain que le fait de tester les algorithmes avec plus de données aurait permis de tirer des conclusions encore plus précises. Mais sommes toutes, avec 2000 données, il a été possible de tirer des conclusions assez précises. Au final, certaines améliorations pourraient être apportées à nos implémentations afin de les rendre encore plus efficaces et conviviaux. Par exemple, implémenter isomap avec l'algorithme dijkstra aurait pu être testé afin de déterminer si les résultats avec cette méthode sont plus concluant. De plus, différentes mesures de performances auraient pu être utilisés afin d'offrir des statistiques plus expressives.

K-médoide	Pearson avec décalage simple
Partition binaire	Non concluant
PCoA	Pearson avec décalage simple
Isomap	Pearson avec décalage simple
KNN	Pearson avec décalage simple