# Projet STT 3795

Fondements théoriques en science des données


**Étudiant : Mathieu Lemire**

Professeur : Guy Wolf

28 avril 2025

# Table des matières

# Introduction

Dans ce projet, on testera deux techniques de classification vues en classe pour voir comment elles performent sur le jeux de données MNIST. On commencera par utiliser naïvement les algorithmes sur les données brutes sans étape de prétraitement, ce qui leur attribuera un score initiale. Ensuite on tentera quelques techniques de prétraitement et on étudiera leur impact sur la performance des algorithmes.

L'accent de ce projet porte sur l'analyse des données MNIST et sur des techniques transformation des données pour extraire de l'information afin d'obtenir un meilleur taux de classification ou, tout simplement, une meilleure compréhension des données.

On attribuera un système de pointage et de précision selon cette logique :

$$p_i = \mathbb{1}\{\text{Bonne prédiction}\}$$

Le score finale par chiffres seraient donc calculer comme suit :

$$s_i = \sum_{i=1}^{10000} p_i$$

Les techniques de classifications utilisées seront :

- KNN
- SVM

## Réduction de dimensionnalité sur les données originales

En appliquant directement la réduction de dimensionnalité sur les données, mon Google Colab plante à cause de manque de mémoire RAM. Il faut donc utiliser un sous-échantillon pour les données d'entraînement, utilisons 2000 d'entre eux.

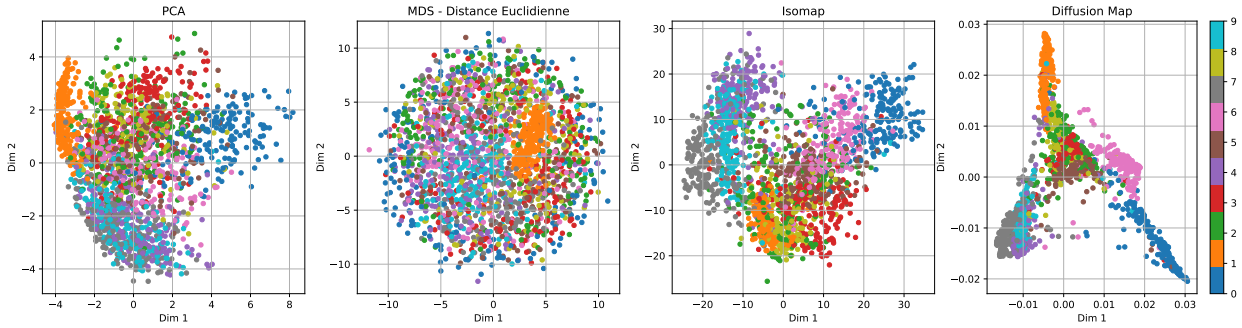


**Figure 1** – Réduction de dimensionnalité 2000 données originales en 2D

On voit que la méthode MDS ne semble pas donner beaucoup d'information. Elle a été calculer avec la distance euclidienne mais puisque les pixels ressemble à des cases, il semble plus naturel d'utiliser la distance de Manhattan pour MDS. Voyons la différence entre les MDS calculé avec la distance euclidienne et la distance de Manhattan

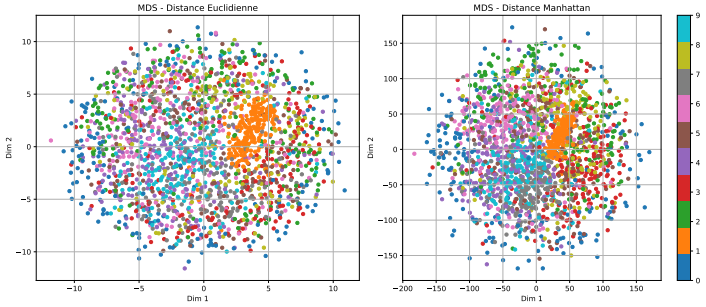


**Figure 2** – MDS Manhattan vs Euclidienne

La distance de Manhattan semble être plus fiable pour MDS et on l'utilisera pour les prochains graphiques.

# Objectifs

L'objectif principale est de trouver des techniques de prétraitement de données qui augmenteraient significativement le taux de classification des algorithmes utilisés.

## Score sur les données originales

On obtient déjà un bon taux de classification en appliquant les deux techniques sur les données originales. Les scores seront toujours afficher par la même type de graphique et on compara les différences. On utilisera que les techniques de réduction de dimensionnalité PCA et les cartes de diffusion (50 dimensions pour les deux). Le problème pour MDS et Isomap est le nombre de RAM disponible sur Google Colab, les deux techniques utilisent davantage de RAM que la capacité sur Colab. J'aurais bien aimé ajouter Isomap car elle semblait une bonne technique de réduction pour les données.

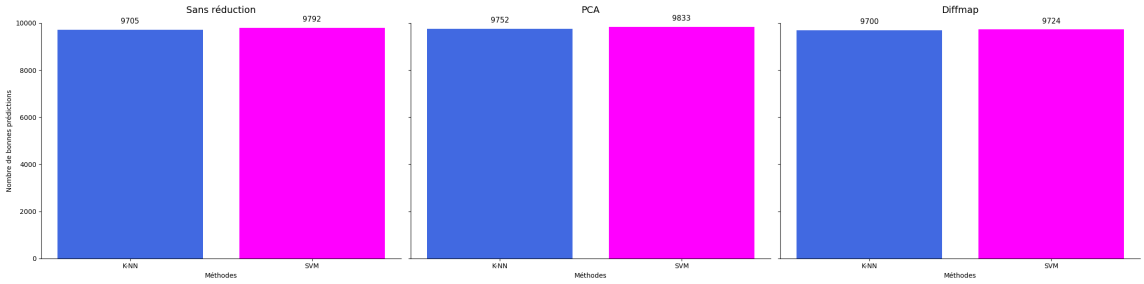


**Figure 3** – Score sur les données originales

Le score initiale pour KNN est 9705 et pour SVM est 9792. On voit que PCA augmente la précision pour KNN de 47 points et de 41 points pour SVM, ce qui était attendu. Par contre, les cartes de diffusion diminue le score de 5 pour KNN et de 64 pour SVM. Peut-être est-ce à cause du nombre de dimension trop élevé ? Je ne peux pas étudier toute les hypothèses dû au manque de temps.

# Prétraitement de Données

Dans cette section, on essaiera d'extraire le plus d'information des données en les transformant tout en minimisant la perte d'information lié à ces transformations. On appliquera successivement les méthodes suivantes aux chiffres d'entraînement et de test puis on testera les deux méthodes de classification dessus :

1. Centrer les pixels
2. Faire une rotation des pixels dans autour des axes de cadrages
3. Cadrer les pixels et faire une transformation linéaire vers 27x27

On affichera les réductions de dimensionnalités 2D des données transformées à chaque étapes ainsi que leur score de classification. On n'affichera pas les réductions 3D par manque de place, ils sont tous sur le notebook

## Analyse de densité

Tout d'abord, regardons comment la répartition des chiffres dans le jeux de données, le nombre de chacun des chiffres est répartit comme suit

| Chiffres | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| Nombre de chiffre d'entraînement | 5923 | 6742 | 5958 | 6131 | 5842 | 5421 | 5981 | 6265 | 5851 | 5949 |
| Nombre de chiffre de test | 980 | 1135 | 1032 | 1010 | 982 | 892 | 958 | 1028 | 974 | 1009 |

TABLE 1 – Répartition des chiffres d'entraînement et de test

Les chiffres semblent bien répartit. Voici quelques exemples de chiffres

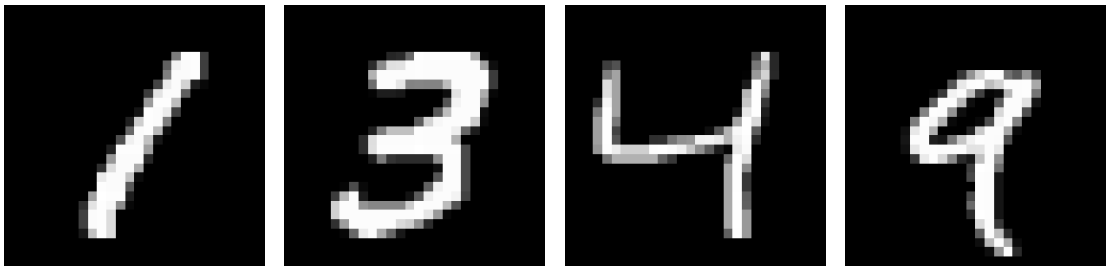

**Figure 4** – Exemples de chiffres

Comme on peut le voir, les images de pixels $28 \times 28$ possèdent une grande quantité de 0 qui donne peu d'information sur les chiffres. En fait, des 47040000 pixels des chiffres d'entraînement($28 \times 28 \times 60000$), 8994156 des pixels. Donc $\frac{38045844}{47040000} \approx 0.81\%$ des pixels sont des zéros.

On pourrait donc se débarrasser des 38045844 pixels et ne prendre en comptes que les pixels non négatives. Si on prend en compte que les 8994156 non négatives et on leur attribues une position relative par rapport à l'images on se retrouve avec une matrice $8994156 \times 3$, car on a pour chaque pixel :

- Une coordonnée horizontale
- Une coordonnée verticale
- Une intensité relative du pixel

Regardons à présent des boxplots qui montre les répartions des intensité relatives de chaque pixel plus grande qu'un $\epsilon$

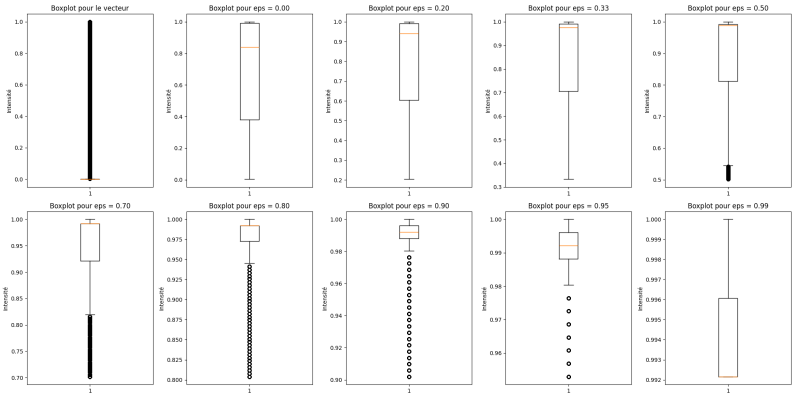


**Figure 5** – Boxplots qui réprésentent les pixels d'entraînements avec un seuil d'intensité de eps

On voit que toutes les pixels non négatives semble être des outliers malgré que toutes l'information utile se trouve dans ces pixels. De plus, il semble intéressant de prendre en compte que les pixels qui atteint un certain seuil ou de change toute les pixels qui atteint un certain seuil à 1, ce qui changerait les dimensions du vecteur des pixels d'entraînement à $|\text{intensité} > \text{eps}| \times 2$

Examinons les distributions de densité sur l'un de ces chiffres. Il serait peut-être intéressé de garder que les pixels atteignant un certain seuil dans l'analyse. On pourrait aussi transformé les pixel qui atteignent un certain seuil à 1, pour montrer la confiance du traçage.

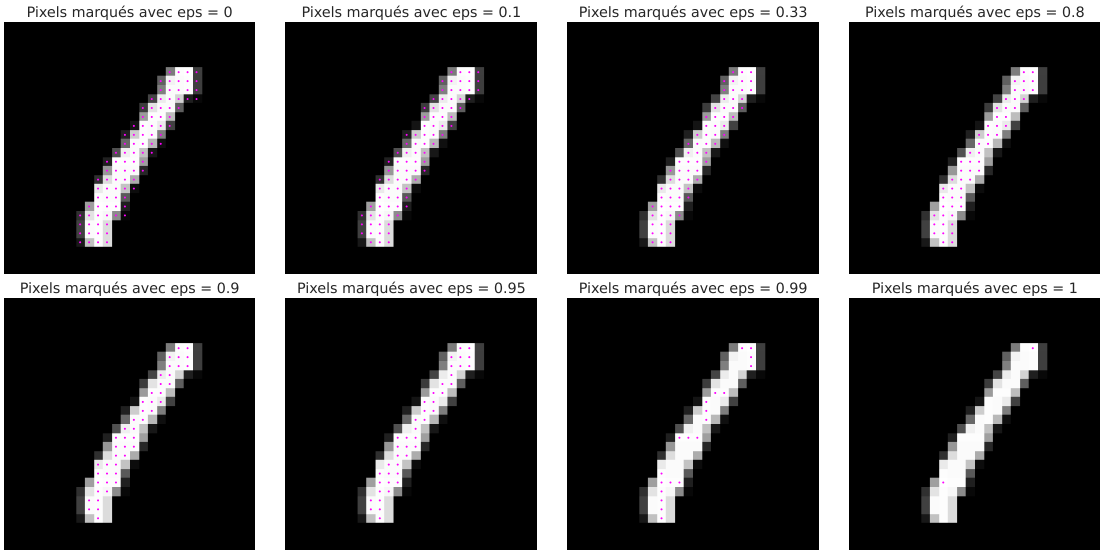


**Figure 6 –** Exemple de 1 avec des points magenta sur les pixels plus grand ou égale à eps

On remarque que seulement 2 pixels possèdent un intensité relative de 1 par rapport à 255. Avec une intensité relative de 0.9, on garde une grande partie de la structure du chiffre. Il faudrait analyser davantage de chiffre pour trouver un bon seuil.

## Analyse des formes et structures

Les chiffres possèdent des structures uniques et d'autres communes. Il serait intéressant de pouvoir trouver avec des données test certaines de ces structures. Énonçons certaines de ces structures :

- Les chiffres 1, 4, 7 et 9 possèdent en général des traits verticales
- Les chiffres 2, 3, 4, 5, 6, 7 et 9 possèdent des traits horizontales
- Les chiffres 0, 2, 3, 5, 6, 8 et 9 possèdent des courbes
- Les chiffres chiffres 0, 6, 9 et certains 2 possèdent en général un "trou" et les 8 en possèdent deux
- Les chiffres 1, 4 et 7 peuvent être écrit de 2 manières différentes

D'ailleurs, la personne qui à tracer le chiffre peut l'avoir tellement mal tracer qu'il ressemble davantage à un autre.

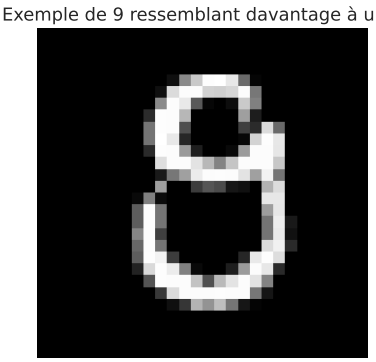


**Figure 7** – Exemple de 9

## Centrage et axes de cadrage

Il est important que les pixels des images soient centrés uniformément par rapport au centre de la matrice de pixels. On doit donc les centrer.

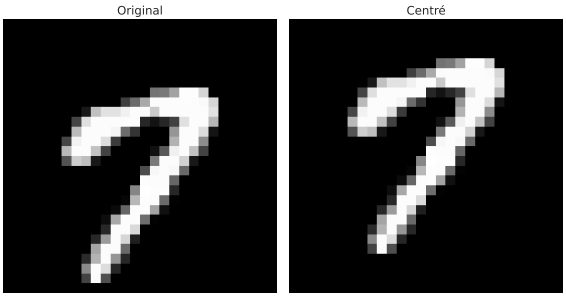


**Figure 8** – Exemple de centrage

Ce centrage est obtenue en rendant la différences entre les lignes de zéros du haut et celui des lignes de zéros du bas à au plus 1. Ce qui équilibre les lignes de zéros de la matrice de pixel.

Analysons les réduction de dimensionnalité sur ces données centrées.


**Figure 9 –** Réduction de dimensionnalité sur les données centré

Les points changent beaucoup par rapport aux données originales et on dirait que l'on perd de l'information. Voyons à présent le score de classification sur les données centrées.

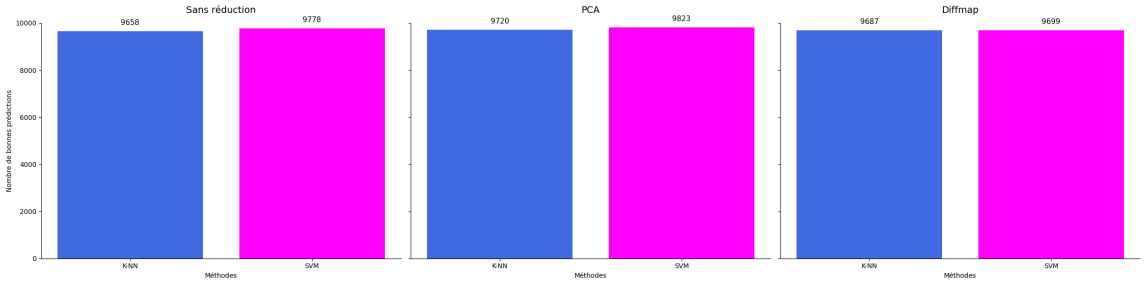


**Figure 10 –** Score de classification pour les données centrées

On voit que l'on perd des points de classification dans tout les méthodes de classifications, avec et sans réductions. Donc, centrer les données de la manière employée et appliquer les méthodes de classification ne semble pas très efficace.

Voyons à présent, ce à quoi ressemble des exemples d'axes de cadrage d'une matrice de pixel pour des matrices de pixel carrées de taille différente.

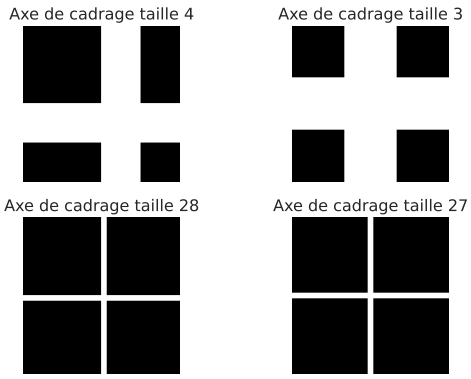


**Figure 11 –** Exemple d'axe de cadrage

On voit que les blocs de matrices sont symétriques si et seulement si la matrice de pixel est de taille impaire. C'est le même phénomène que pour trouver la médiane d'un jeu de données pairs mais dans notre cas il semble déraisonnable de scinder des pixels en deux lorsque la matrice de pixel est pair.

**Définition 1: Axes de cadrage d'une matrice de pixel**

Les axes de cadrage d'une matrice de pixel sont les pixels centrales verticales et horizontales d'une matrice carrée impaire.

Donc, après avoir centrer la matrice, il faut retirer une ligne et une colonne de 0 pour obtenir une matrice 27x27. On utilisera les axes de cadrages pour tenter d'avoir une meilleur rotation dans la section suivante.

## Rotation des pixels par rapport aux axes de cadrage

L'idée ici est de trouver les deux composantes principales d'une matrice et faire une rotation pour que celles-ci soit aligné avec l'axe de cadrage.

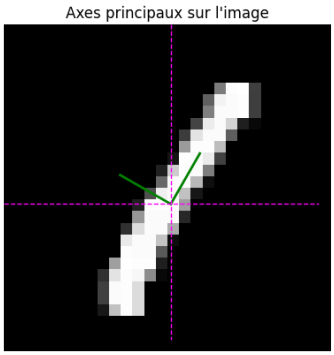


FIGURE 12 – Exemple de chiffre avec 2 composantes principales

On calcule le plus petit angle entre les composantes principales et l'axe de cadrage (On utilise que des vecteurs unitaire de type [0,1], [-1,0] etc). Si cette angle minimale est au-dessus de 25 degrés, on n'applique pas la rotation au risque de faire une mauvaise rotation. (On pourrait améliorer l'algorithme de rotation pour minimiser ces mauvaises rotations)

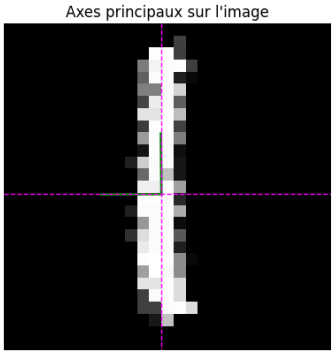


**Figure 13** – Exemple de rotation

On voit que la rotation provoque une perte d'information sur les pixels, il existe peut-être une meilleure technique de rotation que celle utilisé. Malgré que cette technique aligne bien la plupart des chiffres, elle peut faire des rotations de chiffres qui semblait correct et les mettre dans un sens incorrecte (J'ai peut-être utiliser une mauvaise formule mathématique pour la rotation). Donc, on devrait améliorer le taux de classification de certains chiffres et réduire le taux de classification de d'autres chiffres. Des ajustements dans le code pourrait être fait pour minimiser ces cas.

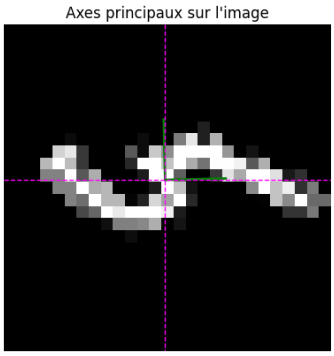


**Figure 14** – Exemple de mauvaise rotation

Analysons les réductions de dimensionnalité sur les données après la rotation.


**Figure 15 –** Réduction de dimensionnalité après rotation

Il semble que PCA soit peu affecté par la rotation, on utilise les 2 composantes principales pour faire cette rotation, alors c'est peut-être l'explication. On dirait que la rotation à légèrement amélioré MDS, les chiffres semblent en moyenne plus regroupé. Pour Isomap, on dirait que la rotation n'a que changé le positionnement des chiffres sur les dimensions. Ce qui semble concorder avec la théorie puisque cette réduction cherche la structure des données. Diffusion Map a peu changé.

Voyons à présent les scores de classification.

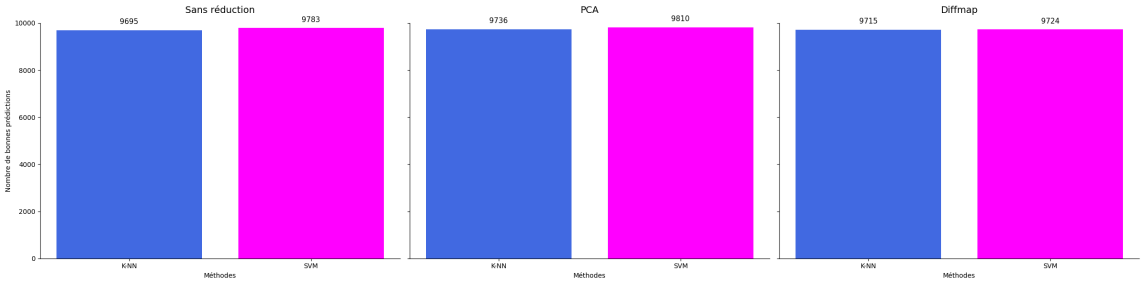


**Figure 16 –** Score de classification après la rotation

On obtient une perte d'environ 10 points pour les deux techniques sans la réduction. On a une plus grande perte avec PCA, environ 20 points en moins pour les deux méthodes. Le seul gain de score obtenue est sur les cartes de diffusion avec KNN. On a un gain de 15 points. Je crois que la technique de rotation utilisé n'est pas optimal et devrait être amélioré.

## Cadrage et transformation linéaire

Le cadrage est obtenue simplement en retirant toutes les lignes et colonnes nuls de la matrice de pixel. Ce qui nous donne une matrice rectangulaire dans la plupart des cas. Ensuite, on applique une interpolation bilinéaire afin de redimensionner la matrice en 27x27 et provoque un effet de zoom.

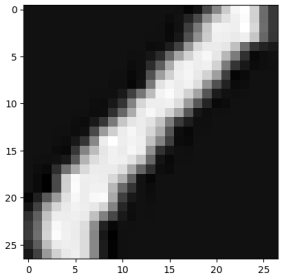

FIGURE 17 – Exemple de zoom

Voyons à présent les réductions de dimensionnalités après avoir appliqué successivement toute les techniques.

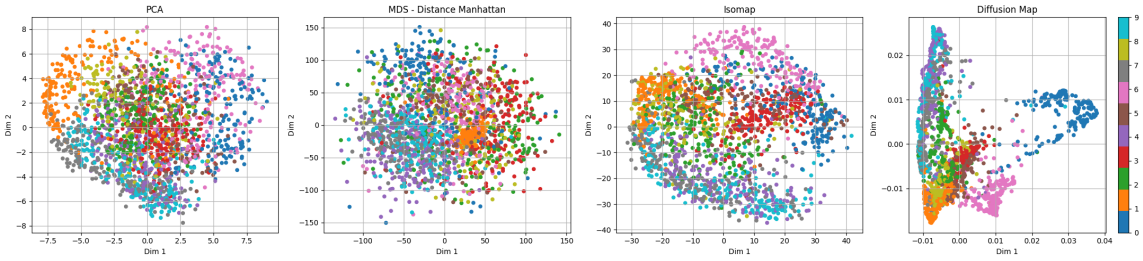


FIGURE 18 – Réduction de dimensionnalité avec toute les techniques

On a l'impression que toute les techniques ont une meilleur réduction.

Voyons les scores de classification.

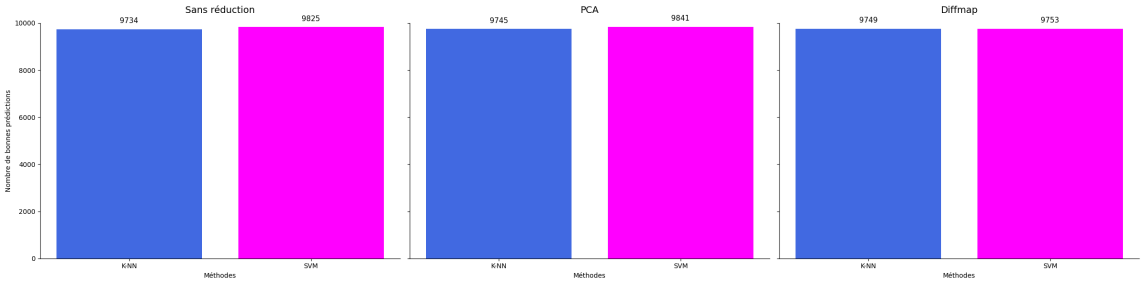


FIGURE 19 – Score de classification avec toute les techniques

La dernière technique semble la plus prometteur, le score sans réduction est augmenté de près de 30 malgré la perte de 10 points lors de la rotation. On obtient notre meilleur score avec toutes les techniques plus la réduction PCA et SVM, soit 9841. Ce qui nous donne une précision de 98.41%. Aussi, on a une augmentation du score avec les cartes de diffusion pour les deux méthodes.

# Conclusion

## Tentatives infructueuses et difficultés

J'avais l'intention d'utiliser une troisième technique de classification, les forêts aléatoires mais je ne l'ai pas utilisé puisqu'il peut donner deux résultats différents pour les mêmes données. Je pensais utiliser les réductions de dimensionnalité MDS et Isomap sur les données mais les 12 GB de RAM de Google Colab n'étaient pas suffisant pour les utiliser. De plus, je pensais faire une analyse des classifications par chiffres mais je n'ai pas eu le temps de les faire. Au final, je suis plutôt dessus du taux de classification de la méthode de rotation. J'ai clairement mal implémenté mon idée mais je trouve intéressant que la théorie du cours m'ai donné l'idée de faire la rotation avec les composantes principales. Les cartes de diffusion ont été décevante pour la classification mais a permis de faire des beau graphique 3D, de même que les autres techniques. (Je ne les ai pas affiché dans le rapport car on voit bien la repartions des points que dans les versions interactives Lien vers un notebook avec visualisation 3D interactifs sur les données originales

## Ce qui a bien fonctionné

Le cadrage et la redimenssion en matrice 27x27 a bien fonctionné. La réduction de dimension avec PCA a bien capté les tendances des chiffres. On a réussi à améliorer le taux de classification d'une cinquantaine de point pour SVM comparativement sans traitement. Ce n'est pas énorme mais ça reste une amélioration.

## Les directions futurs pouvant être intéressantes

- On pourrait améliorer la technique de rotation pour réduire le nombre de mauvaise rotation.
- On pourrait faire les méthodes MDS et Isomap sur un ordinateur possèdant assez de RAM pour faire le calcul.
- On pourrait ne prendre en considération que les pixels dont la densité relative atteint un certain seuil et/ou et les transformer en 1.
- On pourrait utiliser la technique de réduction de dimension t-SNE qui semble approprié pour le jeu de données.
- On pourrait utiliser un CNN qui traite super bien les images.