

MASTER MLDS

**APPRENTISSAGE SUPERVISE**

**Exploration, Visualisation et Classification du Jeu de Données Fashion-MNIST**

Exploration, Visualisation et Classification du Jeu de Données Fashion-MNIST

Auteur :

Auteurs : Jamal Aourraz

Zakaria Hajji

TABLE DE MATIÈRES

[1. Introduction 3](#_Toc533181500)

[2. Préparation des données 3](#_Toc533181501)

[3. Analyse en Composantes Principales 4](#_Toc533181502)

[4. T-SNE (t-distributed Stochastic Neighbor Embedding) 6](#_Toc533181503)

[5. Deep Autoencoder 8](#_Toc533181504)

[6. Clustering dans un espace latent de petite dimension 9](#_Toc533181505)

[7. Conclusion 11](#_Toc533181506)

TABLE DE FIGURES

[Figure 1: Affichage des 4 premières composantes de l'ACP 4](#_Toc533181513)

[Figure 2: Résultats de l'ACP 5](#_Toc533181514)

[Figure 3: Affichage des 5 premières images du jeu de données 5](#_Toc533181515)

[Figure 4: Affichage de 5 images reconstruites après ACP (nc=11) 6](#_Toc533181516)

[Figure 5: Affichage de 5 images reconstruites après ACP (nc=2) 6](#_Toc533181517)

[Figure 6: Résultats de la t-SNE 7](#_Toc533181518)

[Figure 7: Résultats de la t-SNE selon les vraies classes 7](#_Toc533181519)

[Figure 8: Modèle de l'auto-encodeur 8](#_Toc533181520)

[Figure 9: Images d'origine et images résultantes de l'auto-encodeur 8](#_Toc533181521)

[Figure 10: Distribution de la perte d'infromation sur le processus d'apprentissage 9](#_Toc533181522)

[Figure 11: Résultats du Kmeans sur les données encodées 10](#_Toc533181523)

[Figure 12: Résultas de la t-SNE sur les données encodées 10](#_Toc533181524)

# Introduction

# 

# Préparation des données

Le jeu de données Fashion-MNIST est largement utilisé et disponible via plusieurs sources et dans plusieurs packages. On utilisera le module datasets de Keras pour l’importer.

Le jeu de données comporte des données train (x\_train, y\_train) d’une longueur de 60,000 et des données test d’une longueur de 10,000 (x\_test, y\_test). Les images sont codées sur des tableaux de 28\*28 digits, chaque digit correspond au niveau de gris d’un pixel de l’image.

Pour utiliser les données codées en tableaux de 28\*28, on va les transformer en lignes de 748. On passera d’une forme de (60000,28,28) à une forme de (60000, 748) en aplatissant chaque tableau ; pour ceci on utilisera la fonction reshape( ) de la librairie numpy.

Les données sont désormais sous la forme d’un tableau de (60000,748), une autre étape importante du pré-processing et nécessaire avant l’application d’une Analyse en Composantes Principales est de standardiser les données. Cette étape trouve justification dans le fait que les données sont généralement d’une variance instable ce qui influence la qualité du résultat de l’ACP.

Pour ce faire, on utilise StandardScaler( ) du module Preprocessing de Scikit Learn.

# Analyse en Composantes Principales

# Conclusion

Ce projet s’est fixé comme objectif d’appliquer des techniques de réduction de dimension sur un jeu de données de grandes tailles et comparer les résultats.

On a utilisé en premier l’algorithme de l’Analyse en Composantes Principales et puis l’algorithme de t-SNE ((t-distributed Stochastic Neighbor Embedding) et on a conclu que ces deux algorithmes ne donnent pas de bons résultats. Et même quand ils réussissent à réduire la dimension tout en gardant un bon taux d’information, ils ne nous donnent pas une idée sur les classes des données.

En deuxième partie on a utilisé les auto-encodeurs et on a refait l’application de ces deux algorithmes avec l’algorithme du Kmeans. On peut conclure que l’application de ces algorithmes après le passage des données par l’auto-encodeur donne de meilleurs résultats.