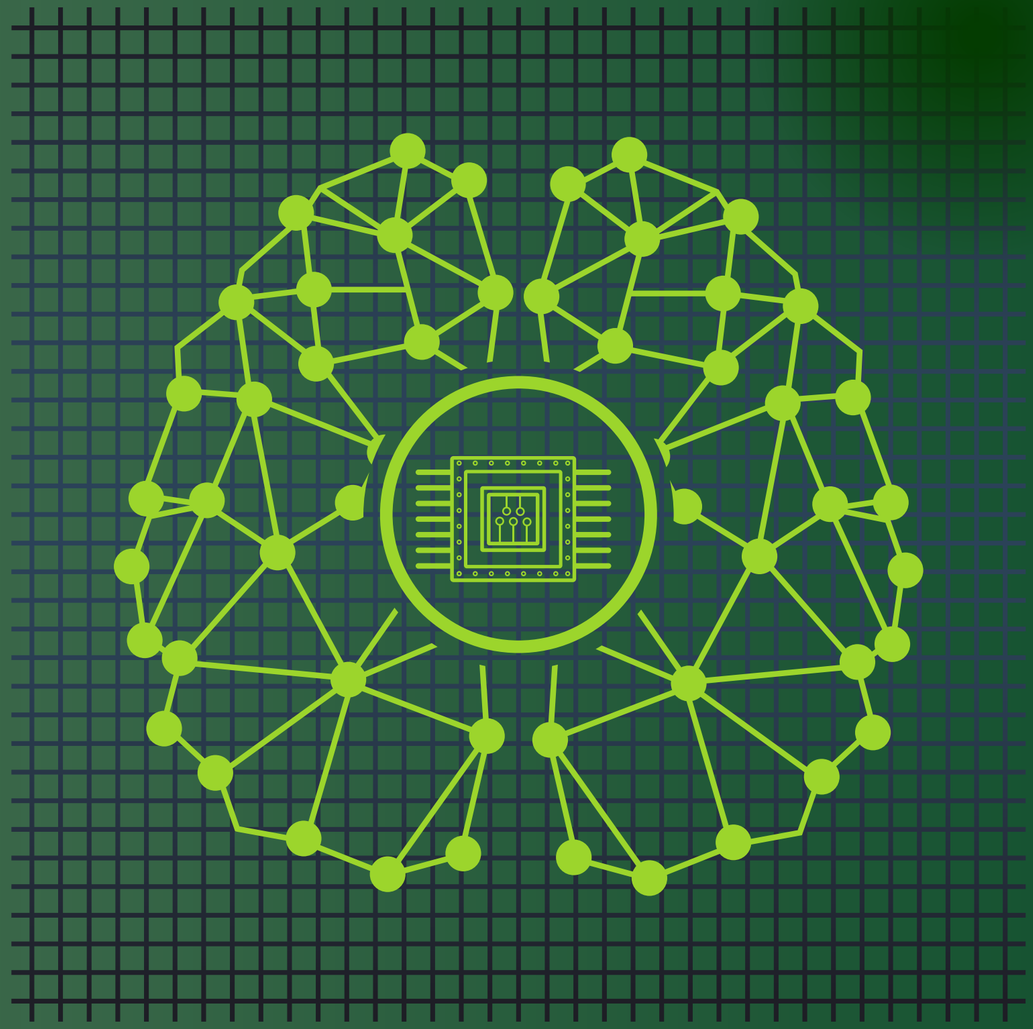
MACHINE LEARNING

BENCHIHA ETAN, COSTANTIN PERLINE, HAJLAOUI KHAOULA



INTRODUCTION

Le Machine Learning permet à un systéme d’apprendre à partir des données afin d’ffectuer des prédictions sur de nouvelles données.

01 Nettoyage et exploration des données

On prépare les données en supprimant les erreurs, en complétant les valeurs manquantes et en les mettant sous une forme exploitable.

02 Choix du modéle

Une fois les données prêtes, on sélectionne un ou plusieurs algorithmes adaptés au problème : régression, classification, ou clustering selon le cas d’usage.

03 Entrainement du modéle

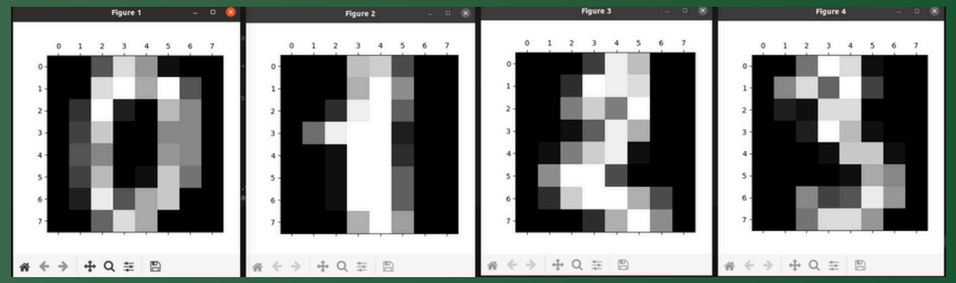
On divise les données en jeu d’entraînement et de test, puis on entraîne le modèle en ajustant ses paramètres pour apprendre à prédire sur la base des exemples fournis.

04 Amélioration

On optimise les performances du modèle en ajustant les hyperparamètres, en testant d’autres modèles, ou en réentraînant avec de nouvelles données. On veille aussi à éviter le surapprentissage.

EXPLORATION

Après normalisation,

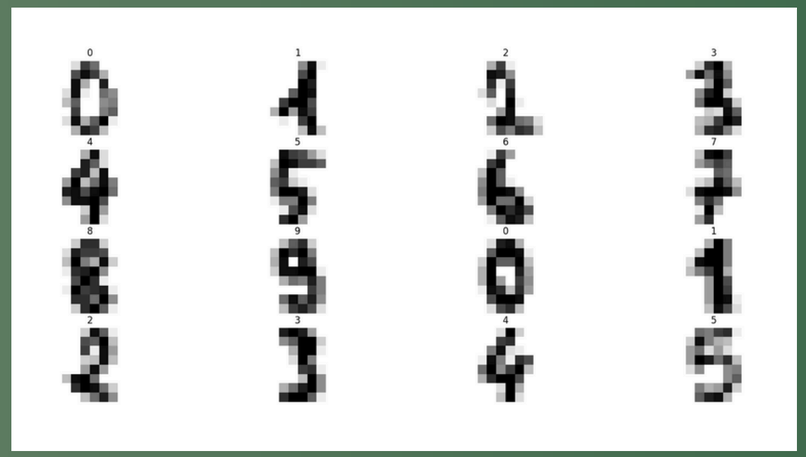


01 Affichage d'exemples :

visualisation des 4 premières images pour se familiariser avec les chiffres manuscrits.

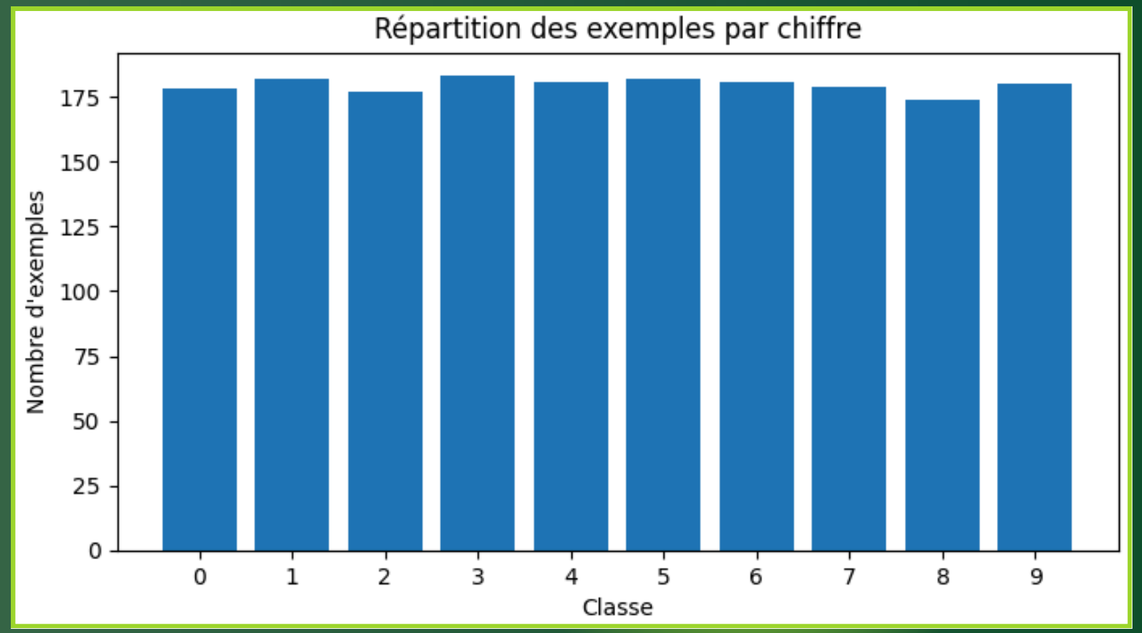
02 Exploration par classe :

on affiche 16 images aléatoires (une par classe de chiffre).



03 Analyse statistique :

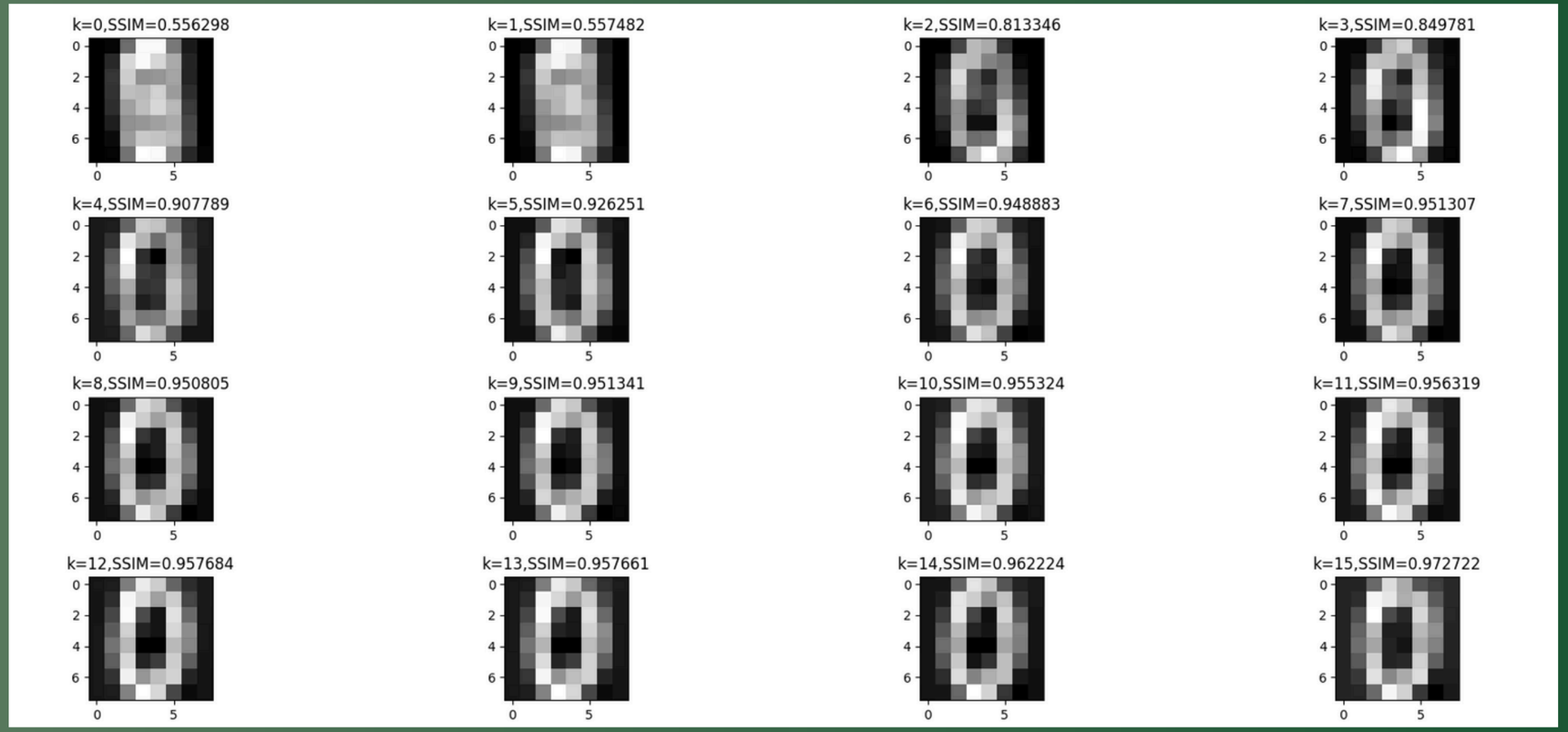
on calcule le nombre de données par classe, la moyenne, les valeurs extrêmes.



FEATURES

01 ACP

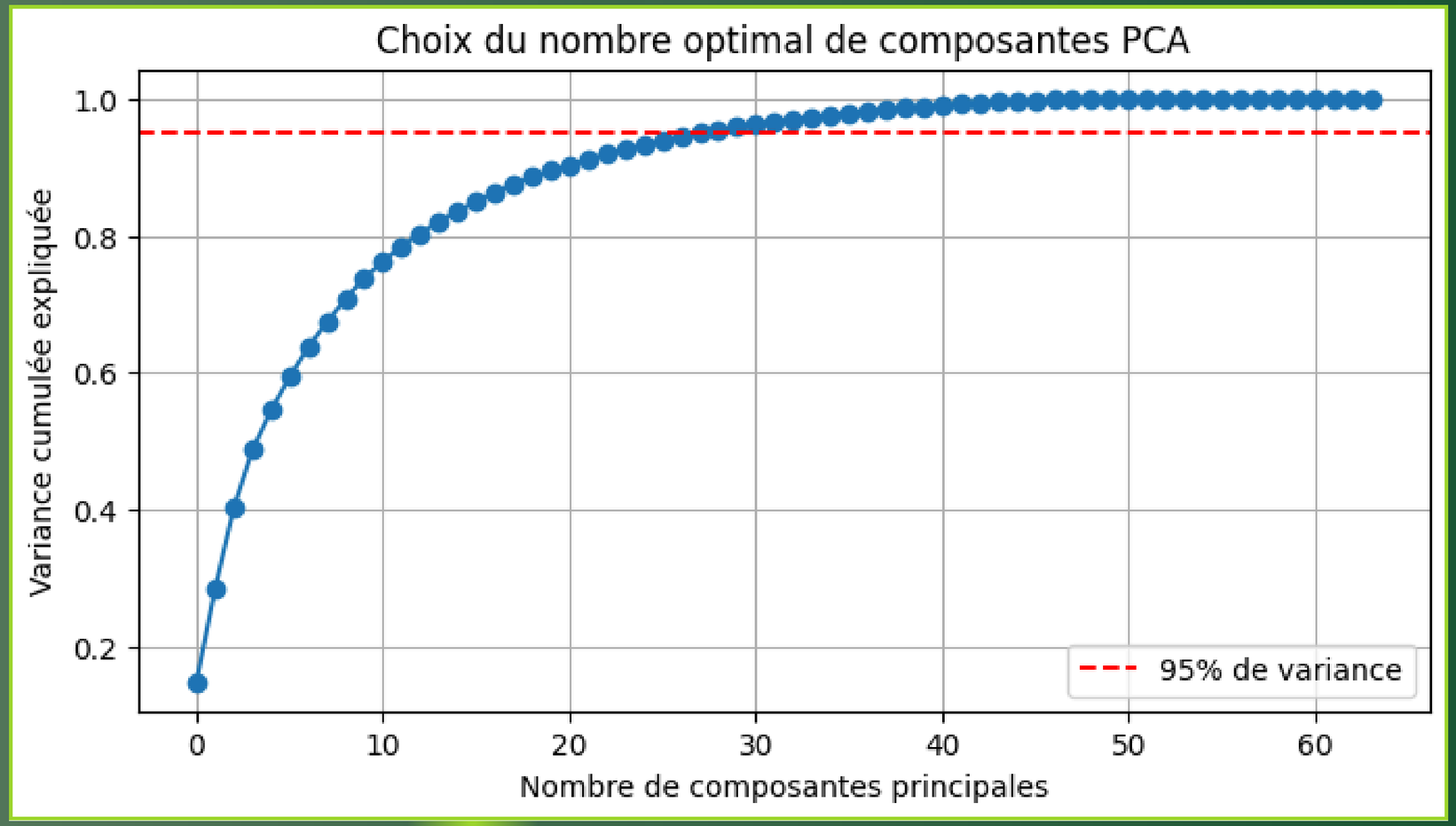
On utilise l'ACP (analyse en composantes principales) pour ne garder que l'essentiel de l'information tout en réduisant la taille des données. On teste l'ACP avec différentes dimensions :



FEATURES

01 ACP

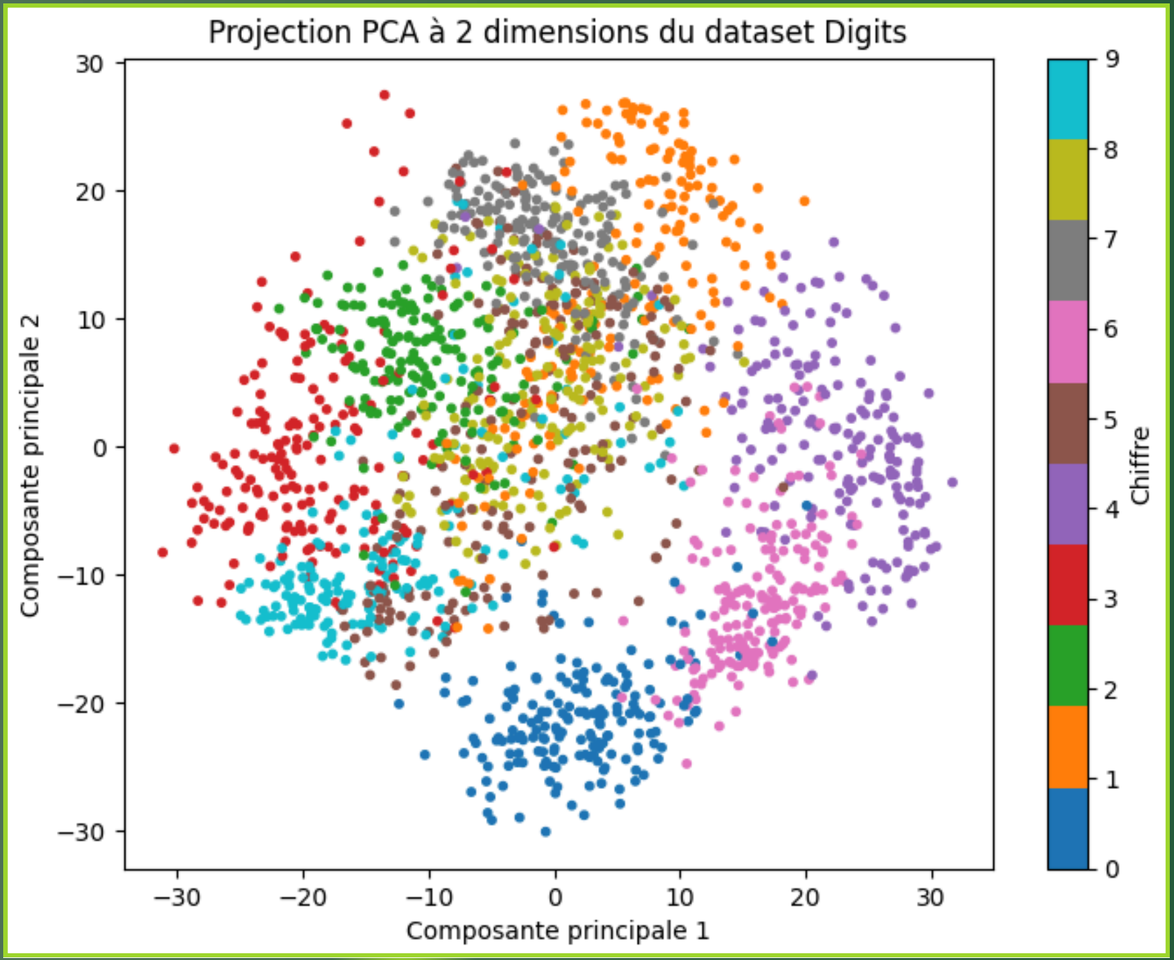
Nous voulons une variance explicatée cumulée de plus de 95 % → On doit prendre 28 composantes pour le PCA



FEATURES

01 ACP-Clusters

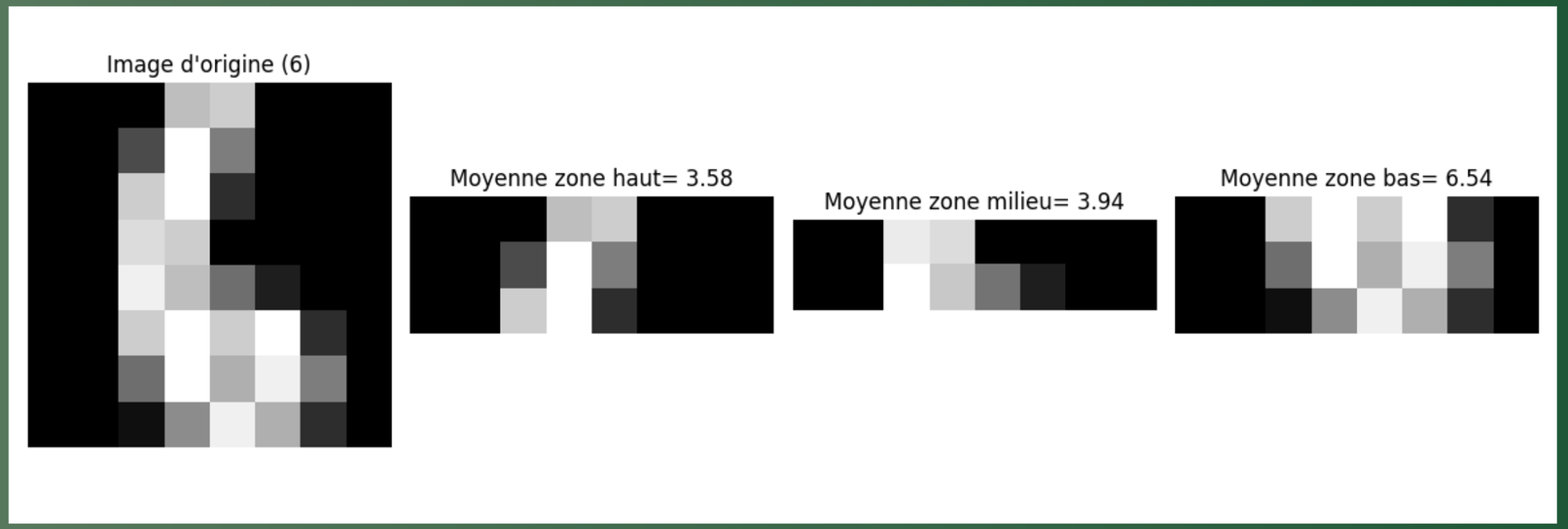
Visualisation des clusters via la PCA à 2 dimensions



FEATURES

02 Positionnement en zones

On divise l’image en 3 bandes horizontales, puis on calcule la moyenne des intensités de chaque bande pour résumer l’intensité de chaque sous-partie.
Cela permet de résumer la répartition verticale du chiffre, ce qui est souvent suffisant pour différencier un 1 vertical d’un 3 arrondi.



FEATURES

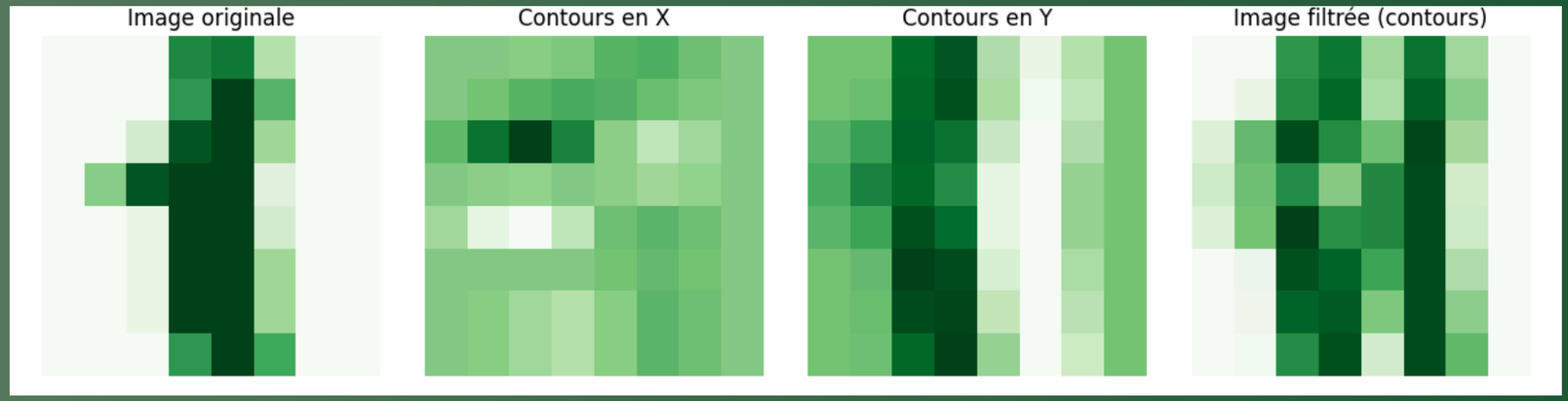
03 Detection des contours - Filtre de Sobel

Le filtre de Sobel permet de détecter les contours dans l'image, en capturant les zones où l'intensité change brusquement. Cela met en évidence la forme globale du chiffre, indépendamment de sa position exacte dans la case.

$$G = \sqrt{G_x^2 + G_y^2}.$$

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \qquad G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

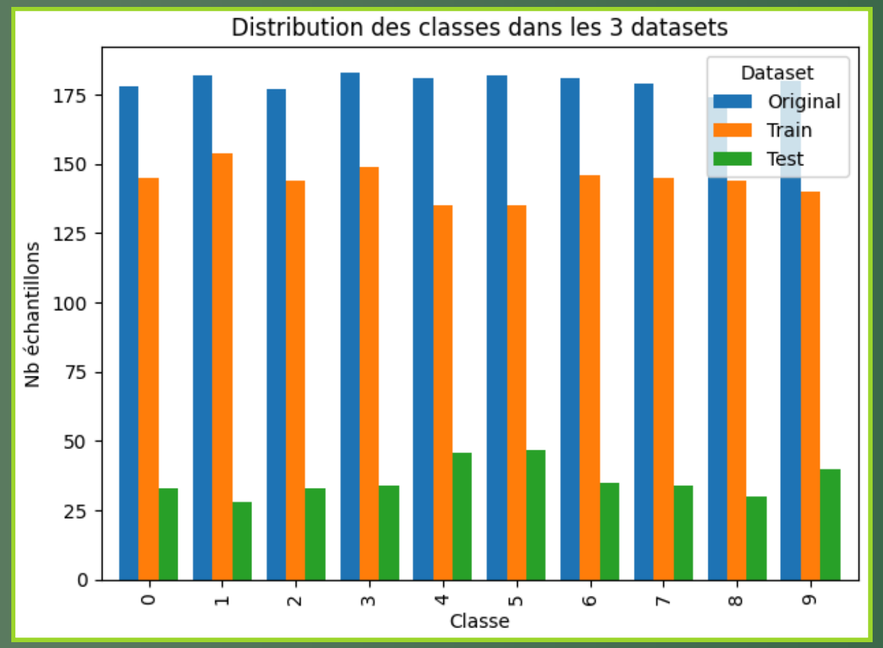
```python
for image in images:
    A_x = ndimage.sobel(image, axis=0, mode='reflect')
    A_y = ndimage.sobel(image, axis=1, mode='reflect')
    A = np.sqrt(A_x**2 + A_y**2)
    m = np.mean(A)
```



DIVISION TRAIN/TEST

01 SÉPARATION DES DONNÉES EN DEUX ENSEMBLES

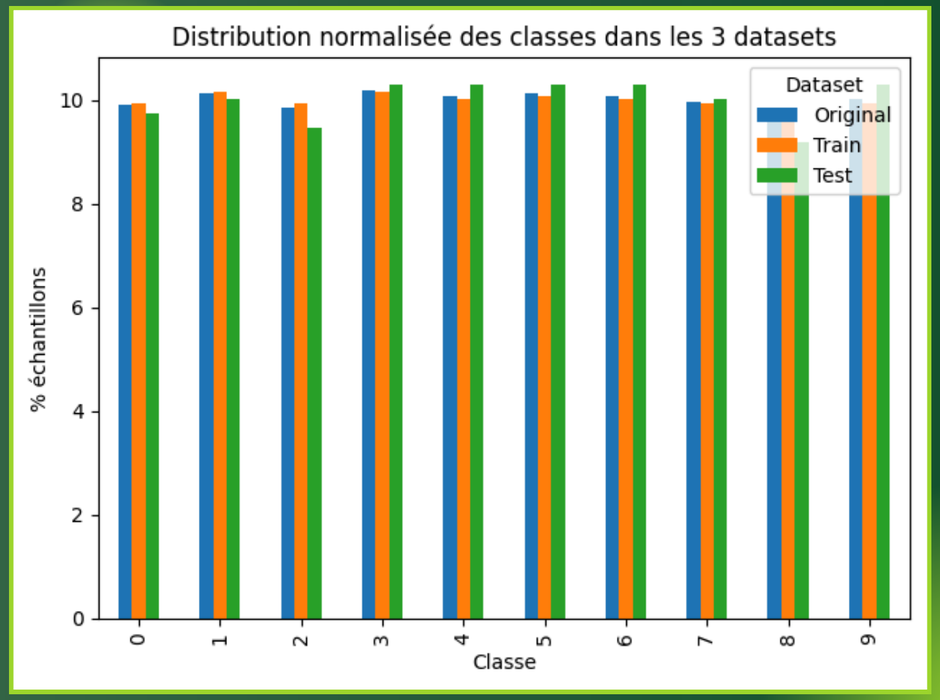
Ensemble d'entraînement (train) : utilisé pour apprendre → 80 %
Ensemble de test (test) : utilisé pour évaluer les performances du modèle sur des données jamais vues → 20%
Cela permet de vérifier si le modèle généralise bien et n'a pas simplement « mémorisé » les données.



Normalisation →

02 VISUALISER LES DISTRIBUTIONS DES CLASSES

Il est crucial de s'assurer que la répartition des classes (ex : nombre de 0, 1, 2. dans le cas des chiffres manuscrits) est équilibrée dans les deux ensembles :



PIPELINE

SVC WITH LINEAR KERNEL

Extraction des features (PCA, zones, bords), normalisation des features avec MinMaxScaler, classification avec un SVC (kernel linéaire)

```
Nb features computed:  32
Accuracy of the SVC on the test set:  0.95
Accuracy of the SVC on the train set:  0.9582463465553236
```

SVC WITH RBF KERNEL

Même pipeline, mais avec un noyau RBF.
Permet de capturer des frontières de décision non linéaires.

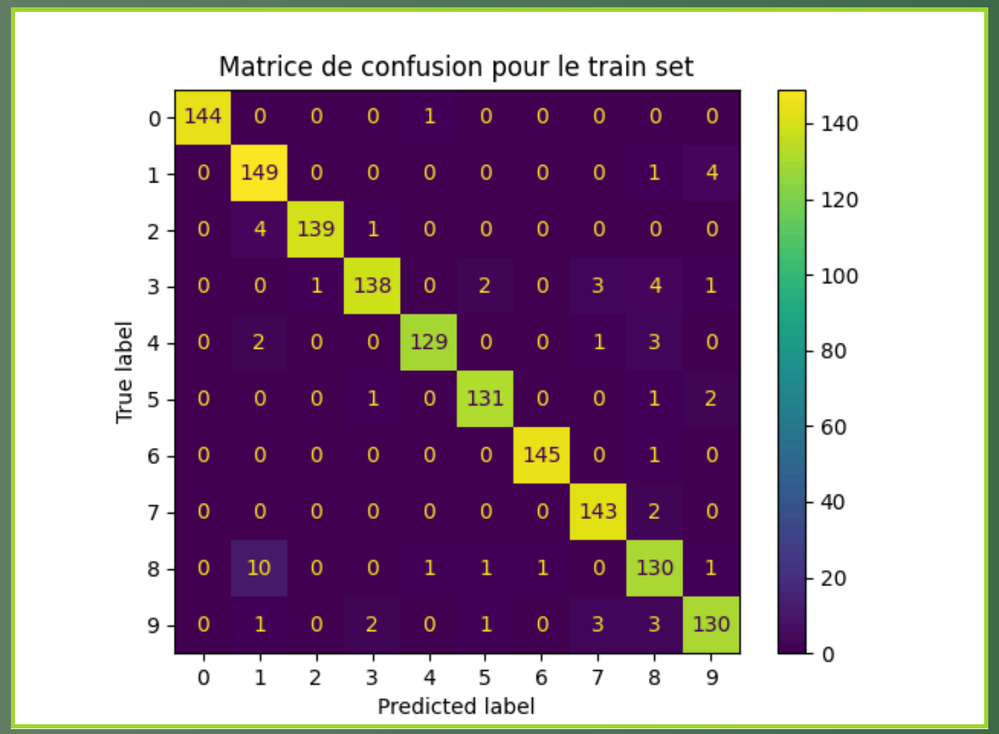
HYPERPARAMETER TUNING

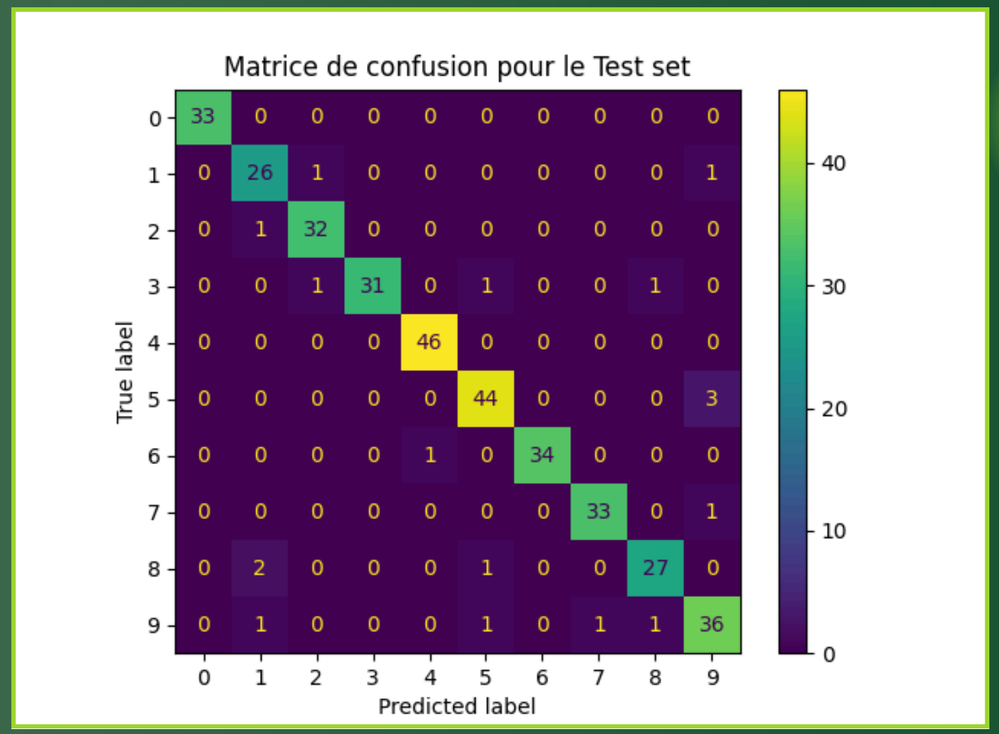
L'optimisation des hyperparamètres permet de trouver la meilleure combinaison de réglages du modèle (comme C, gamma ou le nombre de composantes de la PCA) pour améliorer ses performances.
Grâce à GridSearchCV avec validation croisée, on ajuste ces paramètres.

```
Best parameters: {'features__pca__n_components': 20, 'svc__C': 10, 'svc__gamma': 0.01}
Best cross-validation score: 0.9888695315524585
Accuracy on test set: 0.9861111111111112
```

MATRICE DE CONFUSION-NOYAU LINÉAIRE

La matrice de confusion permet de visualiser les performances détaillées du modèle, classe par classe.
Elle montre combien d'images ont été bien ou mal classées, pour chaque chiffre de 0 à 9.

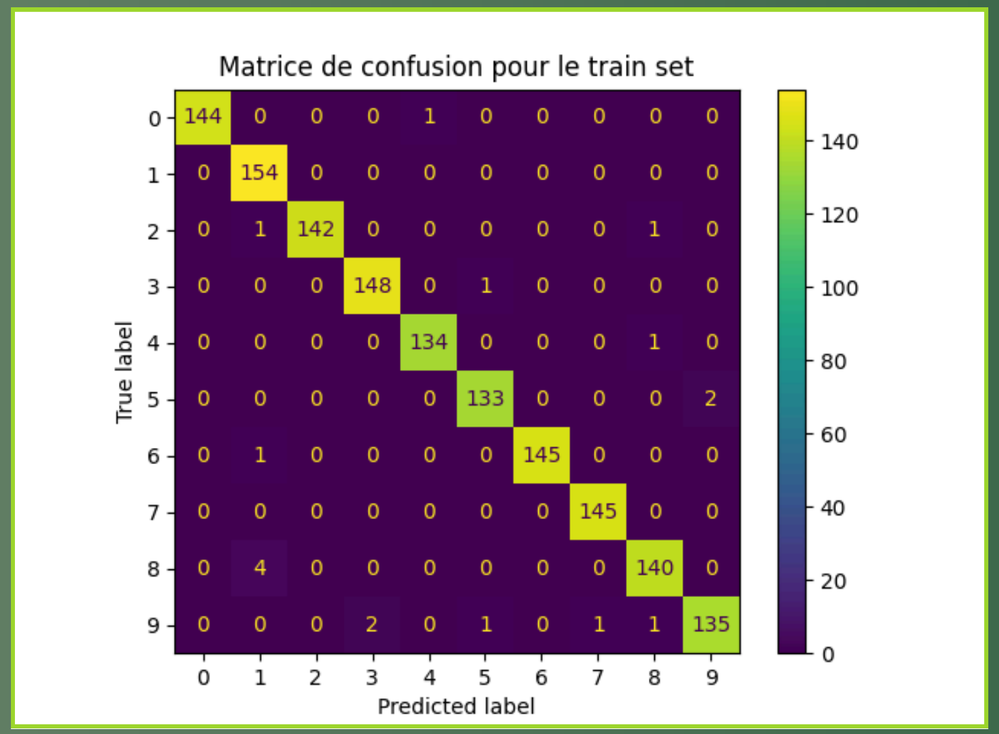


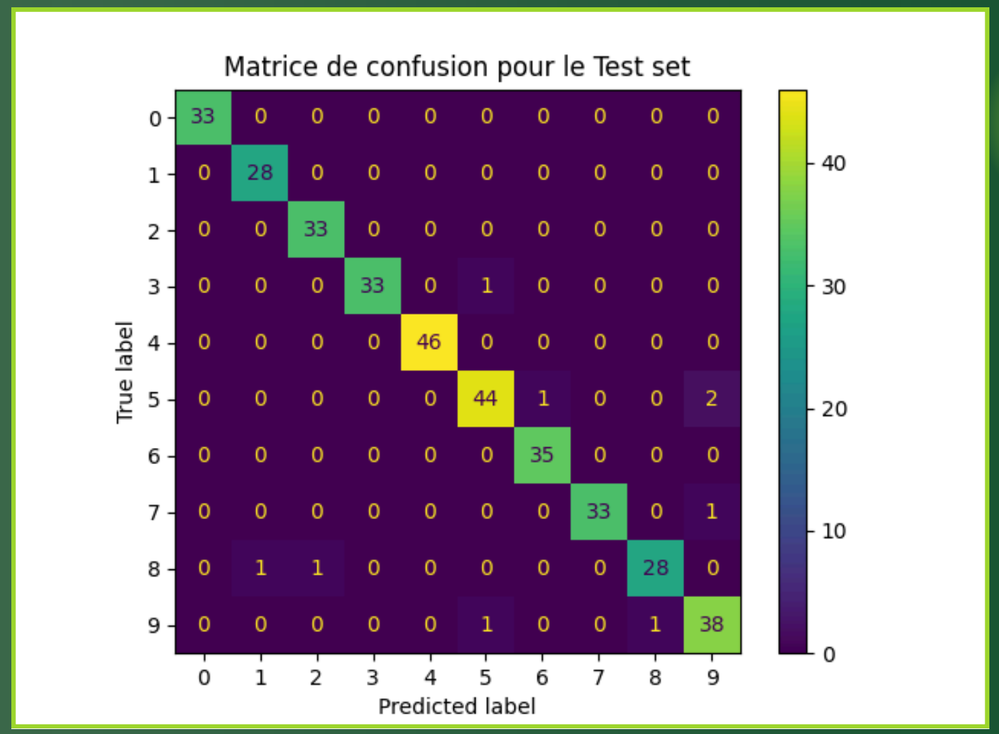


Les bonnes prédictions sont sur la diagonale.
Les erreurs apparaissent en dehors de la diagonale.
Cela permet de voir quelles classes sont le plus souvent confondues entre elles.

Sur 360 données, on a 18 erreurs
Scode de précision : 0.9874

MATRICE DE CONFUSION - NOYEAU RBF





Meilleures prédictions sur le test set avec un noyau RBF

Sur 360 données, on a 9 erreurs

Score de précision : 0.9882

K-FOLD CROSS-VALIDATION

La validation croisée K-fold consiste à diviser l'ensemble d'entraînement en K sous-ensembles (folds).
Le modèle est entraîné sur K-1 folds et testé sur le fold restant. Ce processus est répété K fois, avec un fold différent à chaque fois.

Intérêt

Permet d'évaluer les performances d'un modèle de façon plus fiable en le testant sur plusieurs sous-ensembles des données.
Réduit l'impact d'un mauvais découpage train/test et améliore la robustesse du choix des hyperparamètres.

Résultats

K = 5 donne le meilleur score sur cet ensemble.
Les paramètres optimaux restent globalement stables (C=10, gamma=0.01).
K plus grand nous donne une évaluation plus fine, mais aussi plus coûteuse en temps de calcul.

```
Running GridSearchCV with K=3-fold cross-validation:
- Best CV score with K=3: 0.9861
- Best parameters with K=3: {'features__pca__n_components': 30, 'svc__C': 10, 'svc__gamma': 0.01}

Running GridSearchCV with K=5-fold cross-validation:
- Best CV score with K=5: 0.9889
- Best parameters with K=5: {'features__pca__n_components': 20, 'svc__C': 10, 'svc__gamma': 0.01}

Running GridSearchCV with K=10-fold cross-validation:
- Best CV score with K=10: 0.9882
- Best parameters with K=10: {'features__pca__n_components': 30, 'svc__C': 10, 'svc__gamma': 0.01}
```

OVO & OVR

01 OVR : ONE VS REST

Entraîne un classifieur par classe, contre toutes les autres.
Plus rapide à entraîner, surtout pour peu de classes.

```
OVR :
  Accuracy train : 0.9408
  Accuracy test  : 0.9361
  Temps d'entraînement : 0.3722 s
```

02 OVO : ONE VS ONE

Entraîne un classifieur pour chaque paire de classes.
Peut être plus précis mais plus lent à entraîner.

```
OVO :
  Accuracy train : 0.9569
  Accuracy test  : 0.9500
  Temps d'entraînement : 0.4714 s
```

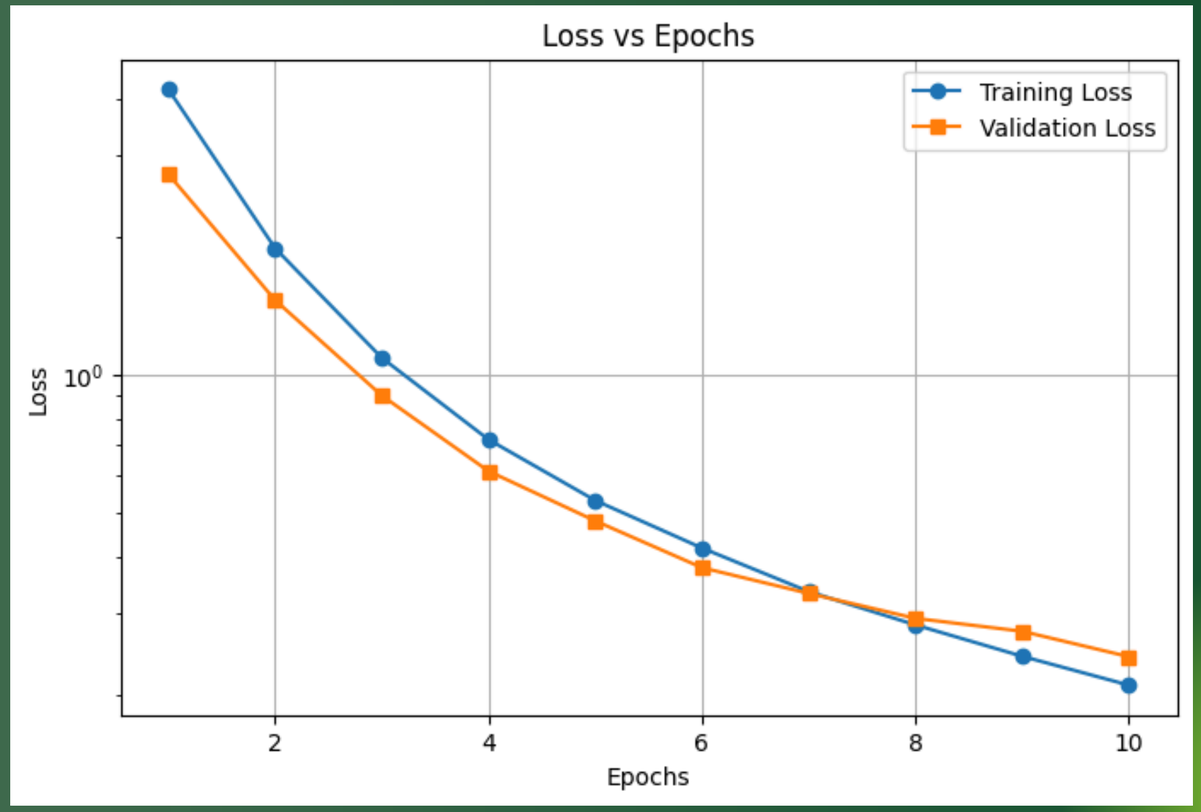
NEURAL NETWORKS

Un réseau de neurones est composé de plusieurs couches de nœuds : une couche d’entrée, des couches cachées et une couche de sortie. Chaque nœud transmet des informations à la couche suivante s’il dépasse un certain seuil, en fonction de ses poids et de sa valeur d’activation.

Etapes

Construction du réseau avec :
Une couche d’entrée (les pixels de l’image)
Une couche cachée avec 32 neurones
Une couche de sortie qui donne la classe du chiffre (0 à 9)

10 époques

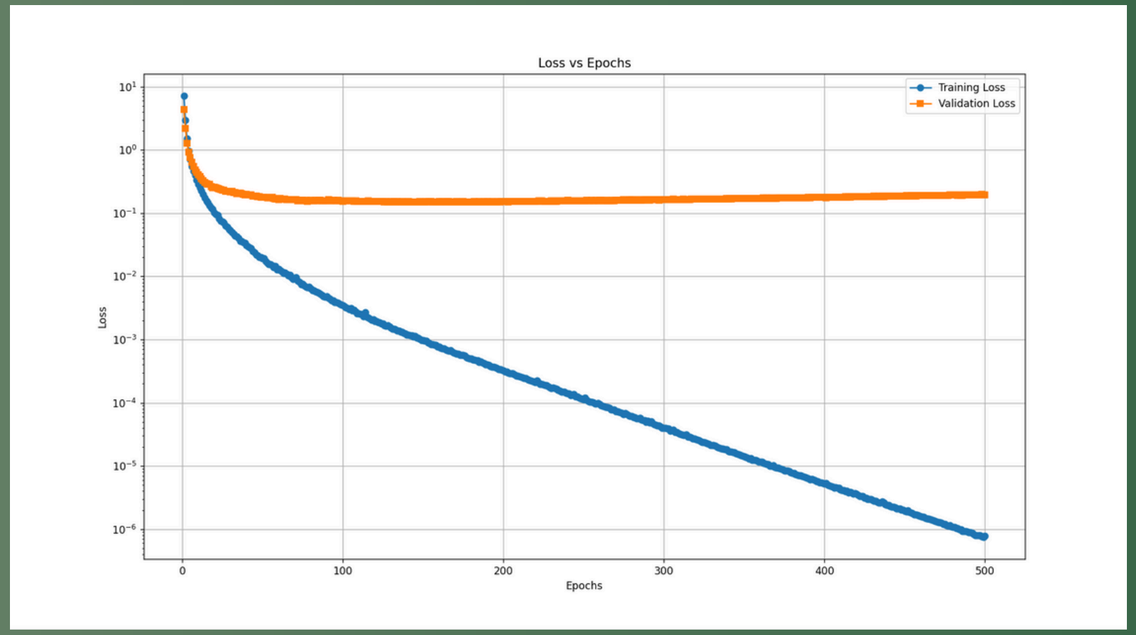


NEURAL NETWORKS

On change maintenant le nombre d’epochs pour comprendre l’évolution du modèle d’apprentissage

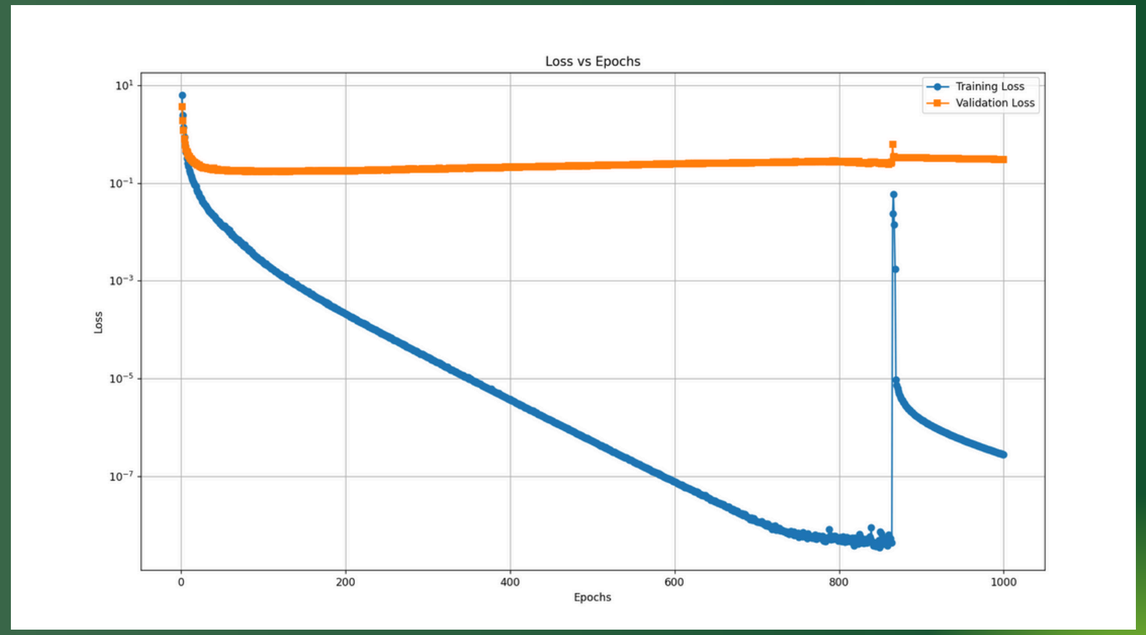
500 époques

Bon compromis entre apprentissage et généralisation.



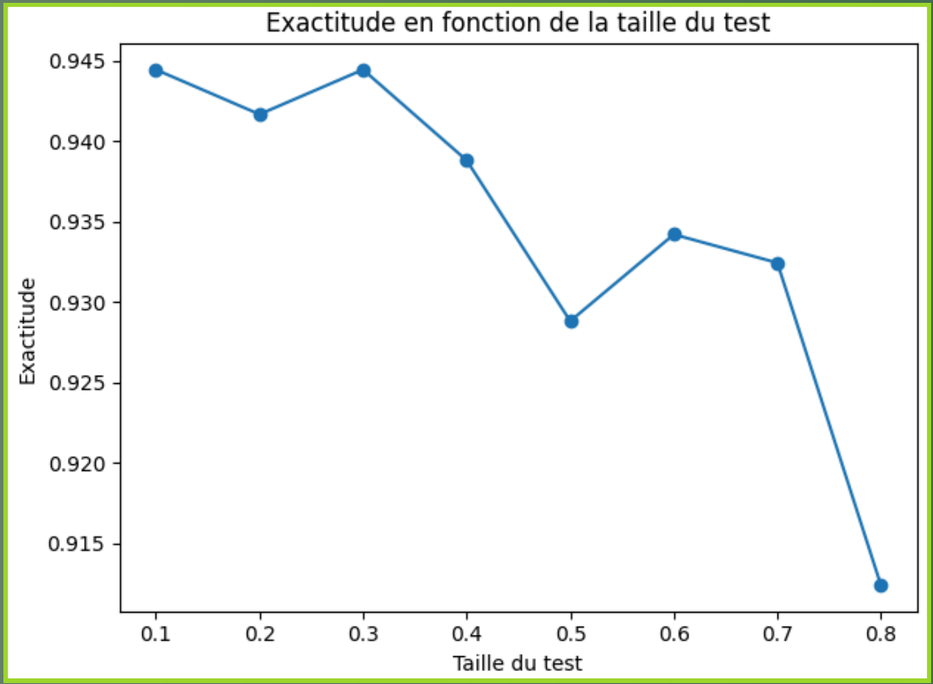
1000 époques

Le graphe montre une singularité, aux alentours des 850 époques (déscentes de gradient faible)



OPTIMIZATION

TEST SIZE



KERNEL

Linear kernel : rapide, efficace si les données sont bien séparées

RBF kernel : plus flexible, capte les frontières complexes

RBF plus GridSearchCV nous donne la meilleure précision (C=10, gamma=0.01)

CLASSIFICATION METHOD

SVC (linéaire) : Accuracy : 0.95

Rapide mais moins flexible, adapté aux données linéaires.

SVC (RBF) : Accuracy : 0.9861

Meilleur score grâce à des frontières non linéaires. Optimisé avec GridSearchCV.

K-NN (k=5) : Accuracy : 0.9722

Très bon score, simple à implémenter, mais plus lent en prédiction.

Conclusion : Le SVC RBF est le plus performant, tandis que K-NN est une alternative robuste et simple.

```
Précision (accuracy) sur le test set avec K-NN : 0.9722
```

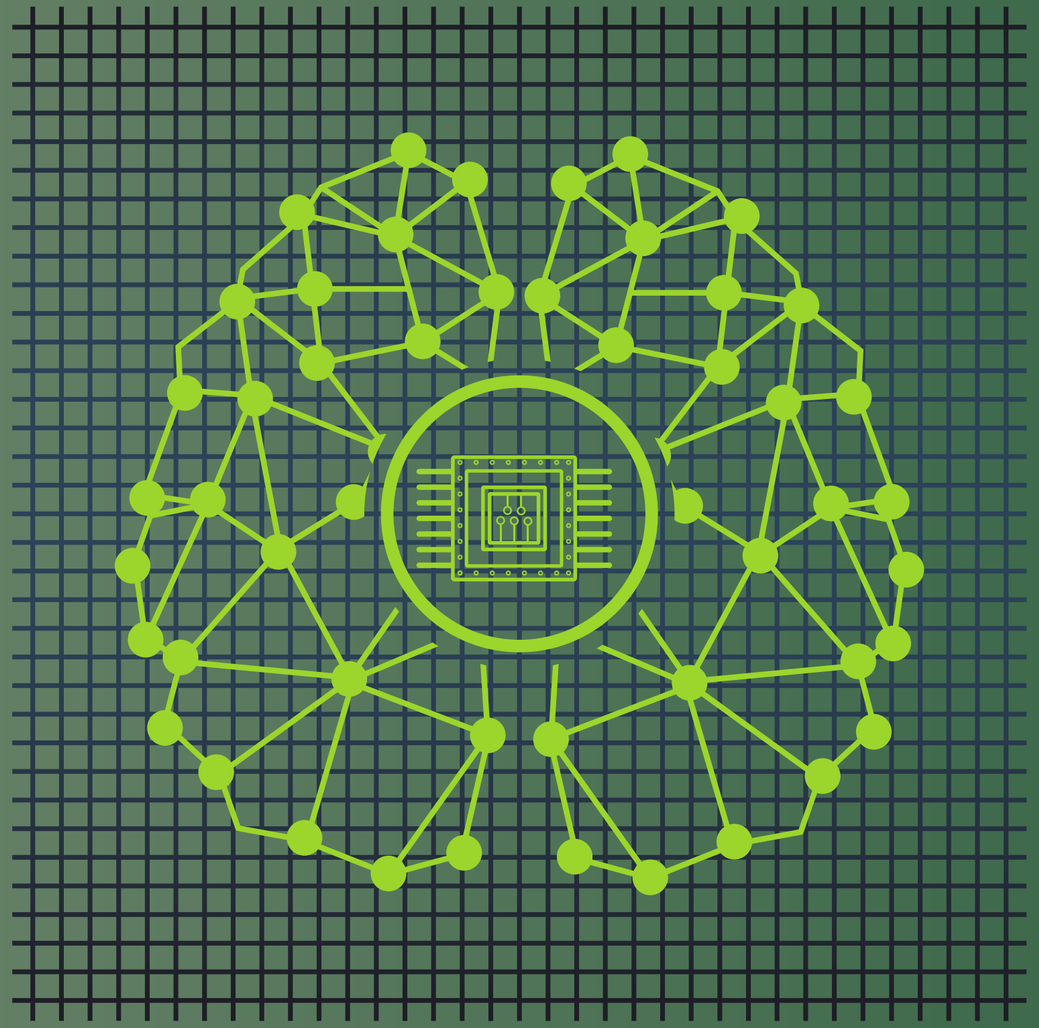
CONCLUSION

Le Machine Learning transforme aujourd’hui de nombreux domaines, en permettant aux machines d’apprendre à partir des données sans être explicitement programmées.

Grâce à des techniques variées (réseaux de neurones, SVM, validation croisée, extraction de features...), on peut créer des modèles capables de reconnaître, prédire et s’adapter à des situations complexes.

Apports personnels :

- Travail en équipe
- Manipulation de Data
- Introduction au réseau de neurones



MERCI POUR VOTRE ATTENTION !