Master Electronique Énergie Électrique Automatique.

Parcours : ROBOT

Année 2024-2025

De l'université de Montpellier

Rapport sur le tp1-AI

Principal Component Analysis and Support Vector Machines

par

[Zeyu YIN]

Tuteur : [Chenji LI]

Le problématique

Dans ce projet, nous nous concentrons sur l'application de deux techniques fondamentales d'apprentissage supervisé : **l'analyse en composantes principales (PCA)** et **les machines à vecteurs de support (SVM)**. Ces deux méthodes jouent un rôle clé dans la résolution du problème de classification des fleurs, en exploitant leurs caractéristiques morphologiques.

D'une part, la **PCA** est utilisée pour réduire la dimensionnalité des données, ce qui permet de simplifier leur représentation tout en conservant la majorité de l'information utile. Cette étape est essentielle pour visualiser les données dans un espace bidimensionnel et pour limiter la complexité du modèle d'apprentissage. La réduction de dimension permet également d'éliminer les corrélations redondantes entre les caractéristiques et de rendre les données plus adaptées au traitement par des algorithmes d'apprentissage.

D'autre part, les **SVM** sont utilisés pour construire un modèle de classification linéaire efficace. Ces algorithmes cherchent à maximiser la marge entre les différentes classes, ce qui garantit une généralisation optimale du modèle. En combinant la **PCA** pour réduire la dimensionnalité et les **SVM** pour effectuer la classification, nous pouvons non seulement identifier les frontières de décision entre les espèces florales, mais également prédire avec précision si un nouvel échantillon appartient à l'espèce *interior* ou non.

Ainsi, ce projet illustre l'importance de l'intégration de techniques de réduction de dimension et d'algorithmes de classification dans la résolution de problèmes complexes de classification.

Les données que je vais traiter

Dans cette étude, nous cherchons à classer des données botaniques associées à différentes espèces de fleurs à l'aide de modèles d'apprentissage supervisé. Les données proviennent du fichier **flowerTrain.data**, qui contient les mesures morphologiques de trois espèces de fleurs : interior, versicolor et convoluta. Chaque espèce est représentée par 48 échantillons. Pour chaque échantillon, quatre caractéristiques morphologiques ont été mesurées :la longueur des sépales, la largeur des sépales, la longueur des pétales, la largeur des pétales.

Toutes ces mesures sont exprimées en centimètres. Ces caractéristiques permettent de capturer les variations morphologiques des fleurs, comme illustré dans la figure ci-dessous.
L'objectif final de cette étude est de développer un modèle discriminant linéaire capable de prédire si un nouvel échantillon correspond à une fleur de l'espèce ***interior*** ou non (c'est-à-dire une des deux autres espèces : ***versicolor*** ou ***convoluta***). Pour atteindre cet objectif, nous avons suivi une approche méthodique en combinant l'utilisation de PCA pour réduire la dimensionnalité des données et de SVM pour construire un classifieur efficace.

Description du travail réalisé

Partie 1

L'objectif de cette première partie est d'explorer les données fournies et de comprendre leur structure à travers une visualisation simple. Cette étape permet de mieux appréhender les caractéristiques des données avant d'appliquer des algorithmes d'apprentissage supervisé.

Les données proviennent du fichier ***flowerTrain.data***, qui contient les mesures morphologiques de trois espèces de fleurs (*interior*, *versicolor*, et *convoluta*). Chaque échantillon est décrit par quatre caractéristiques exprimées en centimètres :

- la longueur et la largeur des sépales
- la longueur et la largeur des pétales

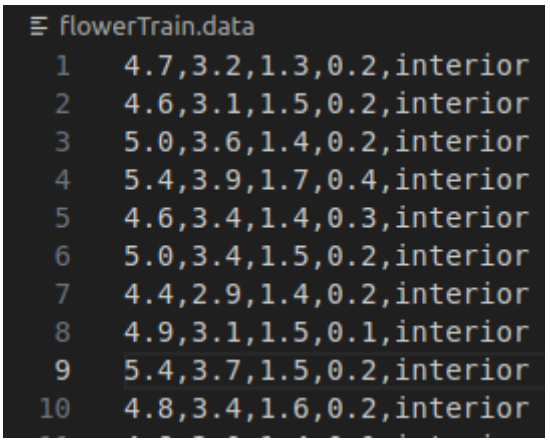

Fig1. 144 Flowerdatas

Pour explorer les données, un diagramme de dispersion en quatre dimensions a été réalisé à l'aide de matplotlib. Voici les conventions utilisées :

- Les abscisses et les ordonnées représentent respectivement la longueur et la largeur des sépales.
- La taille des points est proportionnelle à la longueur des pétales.
- La couleur des points est proportionnelle à la largeur des pétales.

Ce graphique met en évidence la complexité de la classification des échantillons dans cet espace à quatre dimensions, comme le montre la figure ci-dessous :

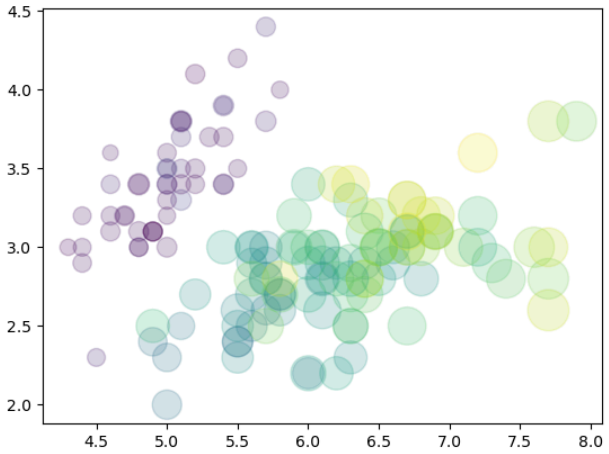

Fig2. quatre dimensions

Cette première visualisation démontre que les données sont bien réparties et qu'une séparation nette entre les classes n'est pas évidente dans l'espace original. Cela justifie l'utilisation de techniques comme l'analyse en composantes principales (PCA) pour réduire la dimensionnalité des données.

In the next Section, I will project the 4-D data to 2-D by selecting the Principal Components, to simplify classification.

Partie 2.1

L'objectif de cette section est de réduire la dimensionnalité des données initiales à quatre caractéristiques en deux dimensions principales, tout en conservant la majorité de l'information. Cela permet de faciliter la visualisation et de préparer les données pour l'entraînement du modèle de classification.

Dans cette section, j'ai défini une fonction **PCA** pour réaliser l'analyse en composantes principales (PCA). Le fichier fournit deux ensembles de données bidimensionnelles qui permettent de vérifier la validité de la fonction. Ces ensembles ont un format où **les lignes représentent les caractéristiques et les colonnes représentent les échantillons**. Ce

format diffère de celui du jeu de données **flowerdata**, où **les lignes représentent les échantillons et les colonnes représentent les caractéristiques**.

Cette différence est cruciale, car elle impacte directement la logique de l'implémentation de la fonction PCA, notamment lors du calcul de la moyenne, de la centralisation des données et de la matrice de covariance. Afin de rendre la fonction flexible et adaptable, j'ai ajouté un paramètre **axis** dans les entrées de la fonction **PCA**. Ce paramètre permet de sélectionner automatiquement les formules adaptées en fonction de la structure des données :

- Si **axis=0**, cela signifie que **les colonnes représentent les caractéristiques** ; cette configuration correspond au jeu de données **flowerdata** ;
- Si **`axis=1`**, cela signifie que **les lignes représentent les caractéristiques**, comme dans les ensembles de données de validation.
-

```
def PCA(X, reducedDim, axis):
```

Fig3. Function PCA

Grâce à cette approche, la fonction PCA est capable de s'adapter à des formats de données variés, garantissant ainsi la justesse et la cohérence des opérations de réduction de dimensionnalité. Cette conception améliore également la réutilisabilité du code pour traiter d'autres types de jeux de données.

```
# 2. Compute the covariance matrix (manual formula)
if axis == 0:  # Compute by columns
    covariance_matrix = np.dot(X_centered.T, X_centered) / (X_centered.shape[0] - 1)
else:  # Compute by rows
    covariance_matrix = np.dot(X_centered, X_centered.T) / (X_centered.shape[1] - 1)
```

Fig4. Choisir mode en axis

Le PCA comporte 6 étapes:
1.On commence par calculer la moyenne pour chaque caractéristique (ou échantillon selon l'orientation des données). Cela permet de recentrer les données autour de zéro.

$$\bar{\mathbf{x}} = \frac{1}{m}\sum_{1}^{m}\mathbf{x}_i \in \mathbb{R}^n.$$

Fig 5. Compute the mean of all measures

2.Chaque valeur des données est ajustée en soustrayant la moyenne correspondante. Cette étape garantit que les données sont centrées autour de l'origine dans l'espace multidimensionnel.

$$\bar{\mathbf{X}} = \left[\mathbf{x}_1 - \bar{\mathbf{x}} \ldots \mathbf{x}_m - \bar{\mathbf{x}}\right].$$

Fig 6. Centralisation des données

3.La matrice de covariance mesure les relations linéaires entre les différentes caractéristiques ou échantillons. Elle reflète la variance de chaque dimension ainsi que les corrélations croisées.

$$\mathbf{C} = \frac{1}{m-1}\bar{\mathbf{X}}\bar{\mathbf{X}}^\top.$$

Fig 7. Calcul de la matrice de covariance

4.La matrice de covariance est décomposée pour extraire ses valeurs propres (indiquant la quantité de variance expliquée) et ses vecteurs propres (représentant les directions principales des données).

$$\mathbf{C} = \mathbf{U}\mathbf{\Lambda}\mathbf{V}^\top.$$

Fig 8. Décomposition en valeurs propres et vecteurs propres

5.Les valeurs propres sont triées par ordre décroissant. Les vecteurs propres associés aux plus grandes valeurs propres sont retenus, car ils capturent la majorité de la variance des données.

$$\mathbf{U}_p$$

Fig 9. Tri des composantes principales

6.Les données originales sont projetées sur les composantes principales sélectionnées. Cela permet de réduire la dimensionnalité tout en conservant la structure informative.

$$\mathbf{s} = \mathbf{U}_p^\top \left(\mathbf{x} - \bar{\mathbf{x}}\right).$$

Fig 10. Projection des données

Grâce à ces six étapes, j'ai pu définir une fonction PCA qui permet d'obtenir les données réduites ainsi que les vecteurs propres correspondant aux principales composantes.

```python
def PCA(X, reducedDim, axis=0):
    if isinstance(X, pd.DataFrame):
        X = X.to_numpy()

    # 1. Compute the mean
    mean = np.mean(X, axis=axis, keepdims=True)
    X_centered = X - mean  # Center the data

    # 2. Compute the covariance matrix (manual formula)
    if axis == 0:  # Compute by columns
        covariance_matrix = np.dot(X_centered.T, X_centered) / (X_centered.shape[0] - 1)
    else:  # Compute by rows
        covariance_matrix = np.dot(X_centered, X_centered.T) / (X_centered.shape[1] - 1)

    # 3. Perform eigen decomposition
    eigenvalues, eigenvectors = np.linalg.eigh(covariance_matrix)

    # 4. Sort by descending eigenvalues
    sorted_indices = np.argsort(eigenvalues)[::-1]
    eigenvectors = eigenvectors[:, sorted_indices[:reducedDim]]
    eigenvalues = eigenvalues[sorted_indices[:reducedDim]]

    # 5. Project data onto principal components
    if axis == 0:
        reduced_X = np.dot(X_centered, eigenvectors)
    else:
        reduced_X = np.dot(eigenvectors.T, X_centered)

    return reduced_X, eigenvectors, eigenvalues
```

Fig 11. Function PCA

Je l'ai donc testé avec les deux chiffres fournis ci-dessous:

$$\mathbf{X} = \begin{bmatrix} 1 & 5 & 3 & 3 \\ 4 & 4 & 3 & 5 \end{bmatrix} \in \mathbb{R}^{2\times 4}.$$

Fig 12. Test Data 1

$$\mathbf{X} = \begin{bmatrix} 1.268 & 4.732 & 3.5 & 2.5 \\ 3 & 5 & 3.134 & 4.866 \end{bmatrix} \in \mathbb{R}^{2\times 4},$$

Fig 13. Test Data 2

Nous pouvons alors obtenir leurs vecteurs propres:

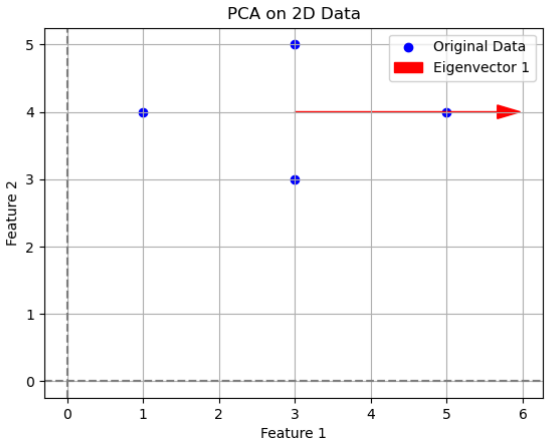

Fig 14. Eigenvector of Test Data 1

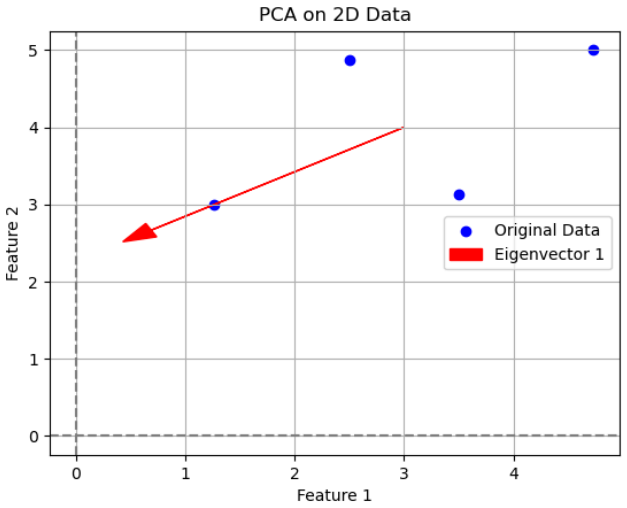

Fig 15. Eigenvector of Test Data 2

Partie 2.2

Dans cette partie, nous utilisons la fonction PCA définie précédemment pour traiter le jeu de données **flowerdata**, en transformant les données initiales à quatre dimensions en un espace bidimensionnel, puis en les représentant graphiquement.

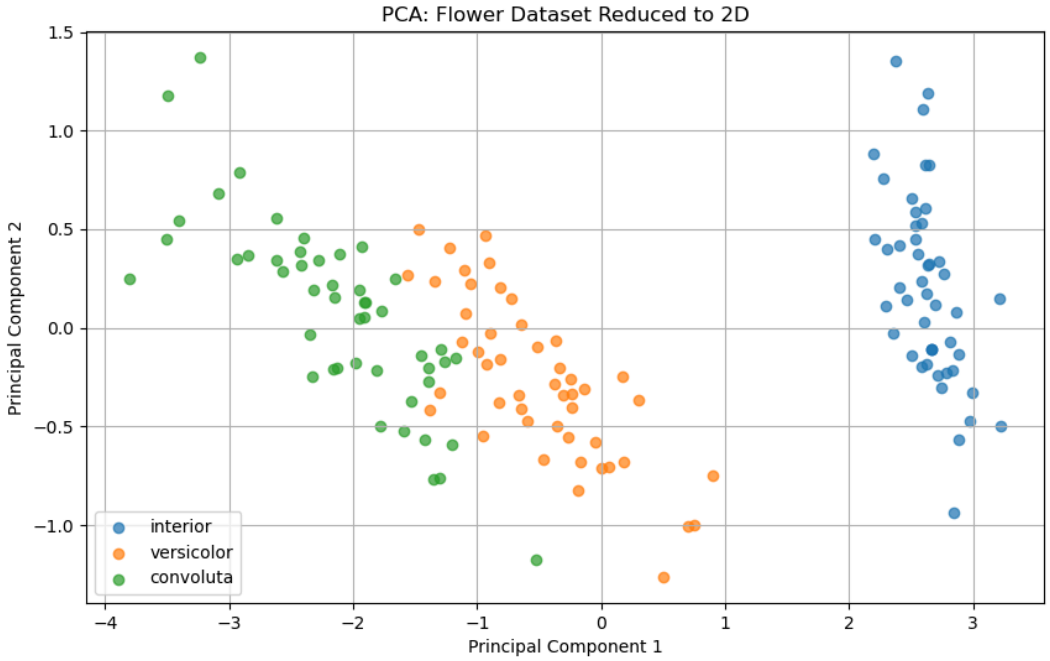

Fig 16. Composantes principales de l'ensemble de données sur les fleurs

Parallèlement, afin de vérifier le succès de la réduction de dimensionnalité, nous affichons les dimensions avant et après la transformation, ainsi que les données projetées, comme illustré dans la figure.

```
Original data shape: (144, 4)
Reduced data shape: (144, 2)
```

Fig 17. Forme d'origine et forme réduite

Partie 3

Dans cette partie, l'objectif principal est d'utiliser une machine à vecteurs de support (SVM) pour classifier les données florales réduites en deux dimensions à l'aide de PCA. Le but est de déterminer si un échantillon appartient à l'espèce *interior* ou aux deux autres espèces (*versicolor* et *convoluta*). Pour cela, nous avons converti les étiquettes d'origine en un format binaire : les échantillons appartenant à l'espèce *interior* ont été étiquetés comme 1, tandis

que les autres ont reçu l'étiquette -1. Cette étape permet de transformer le problème en une tâche de classification binaire adaptée à l'approche SVM.

Une fois les données préparées, nous avons entraîné un modèle SVM avec un noyau linéaire, en utilisant les deux composantes principales (**PC1** et **PC2**) comme caractéristiques d'entrée. Nous avons également défini un paramètre de régularisation **C=1E10** pour éviter une marge excessive tout en maintenant une bonne capacité de généralisation. Le modèle SVM a été entraîné pour trouver une frontière de décision optimale qui maximise la marge entre les deux classes et minimise les erreurs de classification.

```
data['binary_label'] = data['species'].apply(lambda x: 1 if x == 'interior' else -1)

# Extract the reduced features and labels
X = data[['PC1', 'PC2']].to_numpy()  # Feature matrix
y = data['binary_label'].to_numpy()  # Labels
def svm_model(X, y, C=1E10, plot_decision_boundary=False):
```

Fig18. SVM model

Ensuite, nous avons visualisé les résultats obtenus. La frontière de décision, représentée par une ligne rouge continue, sépare les échantillons appartenant à l'espèce *interior* de ceux des autres espèces. Les marges de support, correspondant aux étiquettes +1 et -1, sont indiquées par des lignes rouges pointillées. Les vecteurs de support, c'est-à-dire les points critiques qui définissent la marge, sont marqués par des cercles vides. Enfin, les données réduites sont affichées en fonction de leurs deux composantes principales, avec des couleurs différentes pour distinguer les trois espèces.

Les résultats montrent que le modèle SVM parvient à tracer une frontière de décision claire, bien que certaines zones présentent un chevauchement entre les espèces. Les vecteurs de support jouent un rôle clé dans la définition des marges, et leur visualisation permet de mieux comprendre la manière dont le modèle effectue la séparation entre les classes.

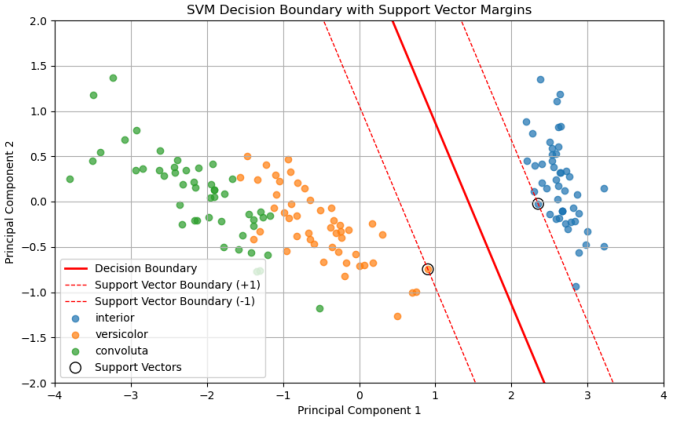

Fig 19. Composantes principales avec un frontiere

Partie 4

Dans cette partie, l'objectif principal est de tester le modèle SVM entraîné précédemment sur de nouveaux échantillons et de visualiser leur position dans l'espace réduit en deux dimensions par PCA. Ces échantillons représentent des fleurs hypothétiques dont les mesures morphologiques sont fournies. L'objectif est de vérifier si ces nouveaux points appartiennent à la classe *interior* ou aux deux autres espèces.

Tout d'abord, nous avons défini six nouveaux échantillons avec leurs quatre caractéristiques (longueur et largeur des sépales, longueur et largeur des pétales). Ces échantillons ont été soumis à la fonction PCA déjà définie, afin de réduire leurs dimensions à deux composantes principales (PC1 et PC2). Les dimensions initiales et réduites des données ont été affichées pour confirmer que la réduction a été effectuée correctement.

Table 1: New samples to be classified.

Sepal Length	Sepal Width	Petal Length	Petal Width
5.1	3.5	1.4	0.2
7.0	3.2	4.7	1.4
6.4	3.2	4.5	1.5
6.3	3.3	6.0	2.5
5.8	2.7	5.1	1.9
4.9	3.0	1.4	0.2

Fig 20. six nouveaux échantillons

Ensuite, ces nouveaux points ont été ajoutés au graphique existant qui présente la frontière de décision du modèle SVM, les marges de support, ainsi que les données d'entraînement d'origine. Les nouveaux échantillons sont représentés par des étoiles orange distinctes pour les différencier des données d'entraînement. Leur position par rapport à la frontière de décision permet de déterminer visuellement s'ils appartiennent ou non à la classe *interior*.

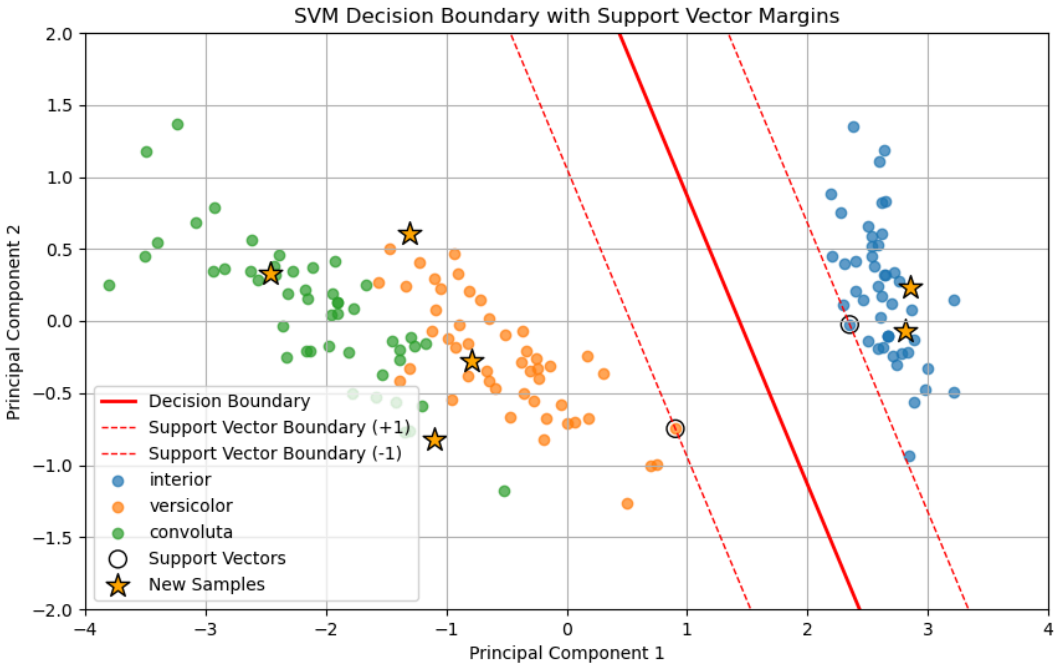

Fig 21.nouveaux points ajoutés

Les résultats montrent que les nouveaux échantillons se distribuent de part et d'autre de la frontière de décision. Certains se situent clairement dans la classe *interior*, tandis que d'autres sont associés aux deux autres espèces. Cette visualisation intuitive confirme l'efficacité du modèle SVM dans la classification des données réduites.

Conclusion

Ce travail pratique a permis d'explorer deux concepts fondamentaux de l'apprentissage supervisé : l'analyse en composantes principales (PCA) et les machines à vecteurs de support (SVM). À travers l'application de ces méthodes, nous avons développé un modèle capable de classifier des échantillons floraux en trois espèces, avec un focus particulier sur la distinction de l'espèce *interior*.

L'utilisation de la PCA a joué un rôle clé en réduisant la dimensionnalité des données de quatre à deux dimensions, tout en conservant l'essentiel de l'information. Cela a non seulement facilité la visualisation des données, mais aussi amélioré l'efficacité et la performance du modèle SVM. La combinaison de PCA et de SVM a permis de gérer efficacement un espace de caractéristiques complexe, offrant une séparation claire entre les classes dans un espace réduit.

Le modèle SVM, entraîné avec un noyau linéaire, a démontré sa capacité à tracer une frontière de décision optimale tout en identifiant des vecteurs de support critiques. La

visualisation des résultats, incluant les marges et les supports, a permis de mieux comprendre la manière dont le modèle réalise ses prédictions.

Enfin, l'ajout de nouveaux échantillons au modèle a confirmé sa robustesse et sa capacité à généraliser. Les nouveaux points ont été correctement classifiés en fonction de leur position par rapport à la frontière de décision. Cela valide l'efficacité de l'approche combinant PCA et SVM pour résoudre des problèmes de classification avec des données complexes et multidimensionnelles.