

TP : CLASSIFICATION

SVM Linéaire

Dans ce TP, vous allez découvrir le fonctionnement d'un SVM linéaire en travaillant sur un dataset réel simplifié (Iris).

Vous devrez visualiser l'hyperplan, les marges, les vecteurs de support et comprendre l'impact du paramètre C.

Le fichier utilisé est : iris_svm_lineaire_pret.csv

PARTIE 1 — Chargement et Visualisation des Données

1. Charger le fichier iris_svm_lineaire_pret.csv.
2. Afficher les premières lignes.
3. Tracer un graphique 2D montrant les deux classes.

PARTIE 2 — Séparation Entraînement/Test

1. Séparer les données en train et test

Fonction autorisée :

```
train_test_split(test_size=0.2, random_state=42)
```

PARTIE 3 — Entraîner un SVM Linéaire

1. Créer un modèle SVM linéaire avec $C = 1$.
2. L'entraîner sur les données d'entraînement.

Fonctions autorisées :

```
SVC(kernel="linear", C=1)
```

```
modèle.fit(X_train, y_train)
```

PARTIE 4 — Impact du Paramètre C

1. Tester plusieurs valeurs de C : [0.01, 0.1, 1, 10, 100]
2. Pour chaque C, mesurer :
 1. Accuracy train
 2. Accuracy test
 3. Nombre de vecteurs de support
3. Que se passe-t-il quand C est très petit ?
4. Que se passe-t-il quand C est très grand ?
5. Quelle valeur de C semble la plus adaptée ? Pourquoi ?
 - 1.

Fonctions autorisées :

`SVC(C=...)`

`accuracy_score()`

Nombre de vecteurs de support= `len(svm_C.support_)`

PARTIE 5 — Accuracy, Precision, Recall

1. Prédire les étiquettes test.
2. Calculer accuracy, precision et recall.
3. Quelle métrique privilégier si la classe 1 correspond à un cas critique (maladie, fraude...) ?
4. Pourquoi accuracy seule peut-elle être trompeuse ?

Fonctions autorisées :

`modèle.predict()`

`accuracy_score()`, `precision_score()`, `recall_score()`