APPRENTISSAGE STATISTIQUE - MS ENSAI
TP2 : SVM-SVR. 2025-2026

Objectifs du TP

Nous allons mettre en application les algorithmes relatifs aux machines à vecteurs de support (en classification et régression), et les comparer aux autres méthodes d'apprentissage supervisé (k-ppv, arbres de décision, régression logistique).

Comme dans les TP précédents, pour chaque jeu de données, il s'agit d'effectuer rapidement une première exploration descriptive des données et une analyse statistique simple des variables d'intérêt. Il faut ensuite utiliser les classifieurs SVM ou régresseurs SVR, tester les différentes fonctions et leurs options, puis comparer les performances avec toutes les méthodes vues précédemment.

N'hésiter pas à aller chercher de la documentation en ligne et à utiliser Chat-GPT ou autre IA pour générer ou corriger des parties de votre code.

1 Classification binaire sur des données linéairement séparables

1°. À l'aide de la fonction `make_blobs`, construire un échantillon d'une centaine de points en deux dimensions, séparés en deux groupes.

2°. Utiliser la fonction `svm.SVC` pour créer un classifieur SVM et fixer provisoirement le coefficient de régularisation C à 1000. Entraîner le classifieur sur les données et afficher le nuage de points à l'aide de la fonction `scatter`. Afficher également les frontières des zones de décision (fonction `DecisionBoundaryDisplay`) et les vecteurs du support (attribut `.support_vectors`). Prendre le temps d'étudier les paramètres et attributs de ces fonctions. Faire varier C.

3°. Reprendre le jeu de données iris de Fisher, afficher les trois classes, puis, comme dans les TP précédents, fusionner deux classes. Afficher le résultat avec `scatter` en sélectionnant deux variables parmi les quatre.

4°. Construire un classifieur SVM, l'entraîner sur le jeu de données, puis l'afficher, ainsi que les hyperplans séparateurs et les vecteurs de support.

2 Classification non binaire ou non linéairement séparables

5°. Avec le jeu de données iris de Fisher originel (et ses 3 classes), définir des classifieurs SVM en utilisant les fonctions `svm.SVC` et `svm.LinearSVC`. Choisir des noyaux linéaire, RBF et polynomial pour classer les données. Entraîner ces modèles, puis afficher les nuages de points ainsi que les zones de décision. Comparer les performances en fonction des noyaux.

6°. Comparer les résultats précédents avec un arbre de décision (fonction `DecisionTreeClassifier`), une régression logistique et la méthode des k-ppv. Afficher les zones de décision correspondantes.

Reprendre la base de données de chiffres manuscrits tirée de l' « UC Irvine Machine Learning Repository ». Pour rappel, c'est un jeu de données regroupant 5620 images de chiffres manuscrits compris entre 0 et 9. Les images de ces chiffres ont été discrétisées en matrices de pixels de 8 lignes sur 8 colonnes, dont chaque coefficient est un entier compris entre 0 et 16 représentant le niveau de gris du pixel. Les données sont disponibles à l'adresse https://archive.ics.uci.edu/ml/datasets, mais on peut les charger directement à partir d'une bibliothèque Python. Nous n'utilisons que la partie test du jeu de données qui comprend 1797 images.

7°. Charger les données via la commande `load_digits` de la bibliothèque `sklearn.datasets`. Définir comme échantillon d'entrainement les données dont les indices vont de 1 à 1438, puis le reste comme données de test. Définir un classifieur SVM, l'entraîner sur la première partie des données, puis évaluer les performances sur la seconde partie (fonctions `accuracy_score`, `classification_report`, `confusion_matrix`).

8° Comparer avec la méthode des k-ppv utilisée dans le TP1.

9°. À l'aide d'une boucle ou de la fonction `GridSearchCV`, comparer les performances du classifieur SVM pour différentes valeurs de C (de 10^{-4} à 10^4 en $\times$ par 10 d'une valeur à l'autre). Tracer la courbe des performances en fonction de C. Pour un choix de noyau gaussien RBF, comparer les performances en fonction du paramètre γ.

10°. Après avoir récupéré les meilleures valeurs des paramètres, entraîner le classifieur et donner ses performances (fonctions `classification_report` et `confusion_matrix`).

11°. charger le jeu de données de visages d'hommes politiques (fonction `fetch_lfw_people`) et afficher quelques images. Vous pouvez éventuellement normaliser les données (fonction `StandardScaler`). Séparer le jeu de données en deux groupes en réservant 40% des données pour les tests. Construire et entraîner un classifieur SVM, après avoir réduit les données via une analyse en composantes principales (fonction `RandomizedPCA`). Pourquoi est-il nécessaire d'utiliser une ACP en amont du classifieur?

12°. Évaluer les performances du classifieur.

13°. Visualiser un échantillon de visages bien classés et quelques visages mal classés. Visualiser les

premiers vecteurs propres obtenus par l'ACP.

3 Régression avec SVR

14°. Créer une centaine de points (en dimension deux) situés sur une sinusoïde et ajouter du bruit sur ces données. À l'aide de la fonction `SVR`, effectuer une régression pour estimer la forme de la fonction initiale. Utiliser les noyaux linéaire, polynomial et RBF et comparer les résultats.

15°. En profiter pour étudier les paramètres et les attributs de SVR et les tester sur le jeu de données précédent.

Le jeu de données « California Housing Dataset » est une base de données très courante utilisée comme benchmark en apprentissage statistique. Il contient des données issues d'un ancien recensement effectué en Californie, relatives au prix des logements en fonction de la position géographique, du nombre de pièces, de l'âge de la construction, de la population du district où le logement est construit, etc. La variable cible est le prix médian du logement. Ce jeu de données est inclus dans la librairie `sklearn.datasets`.

16°. À l'aide de `fetch_california_housing`, importer ce jeu de données et effectuer des statistiques descriptives permettant de visualiser la base de données et d'étudier ses caractéristiques. En utilisant `scatterplot`, visualiser le nuage de points représentant la position géographique des logements.

17°. Effectuer une régression linéaire pour trouver les déterminants du prix médian d'un logement (vous aurez au préalable étudié les corrélations entre variables). Évaluer les performances de cette régression.

18°. Effectuer une régression à l'aide d'un modèle de SVM, en utilisant plusieurs types de noyaux. Comparer les résultats avec la régression linéaire.

4 Étude d'un jeu de données de marketing

Nous allons travailler dans cette dernière partie sur le jeu de données « IBM Watson Marketing Customer Value Data » qui comprend des informations sur 9134 clients d'IBM. La base de données est souvent utilisée pour modéliser le comportement des clients en termes de segmentation, valeur vie client (VVC ou CLV pour Customer Life Value) ou autre comportement d'achat. Le but initial du jeu de données est la modélisation de la VVC, mais il peut être utilisé pour effectuer de la classification binaire en utilisant l'une des variables binaires disponibles. Parmi les variables, on trouve : l'âge, le genre, le lieu d'habitation, le niveau d'éducation, le revenu, le statut marital, la fréquence d'interaction avec les campagnes marketing (clics, réponses aux courriels), l'historique d'achat, la VVC, etc.

19°. Charger la base de données via ce lien (elle est également disponible sur le site du cours sous le nom `customer`). Il s'agit d'un fichier .csv qu'il convient de stocker dans un dataframe Pandas.

20°. Effectuer une étude descriptive du jeu de données, en vous intéressant en particulier à la variable `Response`.

21°. Effectuer une régression logistique permettant de prédire la valeur de `Response`.

22°. Créer, puis entraîner un modèle de machine à vecteurs de support pour prédire la valeur de `Response` et comparer ce modèle avec la régression logistique.

23°. Déterminer une régression linéaire, puis une régression à l'aide d'une machine à vecteurs de support pour prédire la valeur de la VVC.