Version Space Learning

Notes de cours

Nadjib Lazaar
Université Paris-Saclay
lazaar@lisn.fr

Introduction

Le Version Space Learning est une approche d'apprentissage supervisé proposée par *Tom Mitchell*. Il repose sur le principe du filtrage des hypothèses en fonction des exemples d'apprentissage pour restreindre l'espace de recherche à un ensemble de solutions compatibles. Cette méthode est particulièrement utile dans des contextes où les connaissances a priori sont limitées et où l'objectif est d'apprendre une représentation conceptuelle précise.

1 Contexte et Historique

L'apprentissage automatique (*Machine Learning - ML*) est aujourd'hui dominé par les approches **connexionnistes**, en particulier les réseaux de neurones profonds. Ces modèles, inspirés du fonctionnement du cerveau humain, ajustent des milliers de paramètres afin de capturer des régularités dans les données. Cependant, cette approche n'est pas la seule existante en apprentissage automatique.

Avant l'essor des réseaux de neurones, l'apprentissage était souvent vu comme un **processus symbolique**, basé sur des règles logiques et la manipulation de concepts explicites. Parmi ces approches, le **Version Space Learning** (*VSL*), proposé par **Tom Mitchell en 1978**, occupe une place importante en **apprentissage supervisé**. Ce modèle repose sur l'idée d'affiner un ensemble d'hypothèses candidates au fil des observations, en maintenant une **frontière entre les hypothèses les plus générales et les plus spécifiques**.

Le *VSL* appartient aux méthodes d'**apprentissage conceptuel**, où l'objectif est d'apprendre une règle ou une catégorie à partir d'exemples positifs et négatifs. Contrairement aux réseaux de neurones, qui apprennent des représentations implicites sous forme de poids, le *VSL* manipule directement des concepts exprimables en logique propositionnelle.

2 Un Apprentissage Symbolique

L'**apprentissage symbolique** repose sur la manipulation de **structures logiques** et la déduction de **règles explicites**. Contrairement aux méthodes connexionnistes, qui s'appuient sur l'optimisation numérique, les modèles symboliques cherchent à **extraire des concepts interprétables** à partir des données.

Le **Version Space Learning** appartient à cette tradition **symbolique**, où l'objectif est de manipuler un **ensemble d'hypothèses** et de les raffiner en fonction des données observées. Cette approche permet d'avoir des modèles **compréhensibles et interprétables**, un avantage majeur dans les domaines où l'explicabilité est cruciale (santé, droit, finance).

3 Apprentissage de Concepts (Concept Learning)

L'apprentissage de concepts est une approche fondamentale en apprentissage automatique, où l'objectif est d'identifier une règle de décision permettant de distinguer les instances positives des instances négatives. Il s'agit d'un cas particulier de la classification binaire.

3.1 Définition du problème

Un problème de classification binaire est également appelé un **problème d'apprentissage de concepts** (*concept learning problem*). L'objectif est d'apprendre une fonction cible :

$$f : X \to \{0, 1\} \tag{1}$$

où :

— X représente l'espace des instances, c'est-à-dire l'ensemble des objets ou exemples considérés.

— $f(x) = 1$ signifie que l'instance $x \in X$ appartient à la classe positive.

— $f(x) = 0$ signifie que x appartient à la classe négative.

L'ensemble des instances positives, noté C, est défini comme :

$$C = f^{-1}(1) = \{x \in X \mid f(x) = 1\} \tag{2}$$

Autrement dit, un **concept** est simplement un sous-ensemble $C \subseteq X$ contenant toutes les instances positives. Son complémentaire, $X \setminus C$, représente les instances négatives.

Caractéristique	Apprentissage Connexionniste	Apprentissage Symbolique
Représentation	Réseaux de neurones, poids synaptiques	Règles logiques, structures symboliques
Interprétabilité	Boîte noire, difficile à expliquer	Explicable, concepts clairs
Données nécessaires	Gros volumes, besoin d'annotations	Peut fonctionner avec peu d'exemples
Approche	Approximative, probabiliste	Déductive, basée sur des règles
Capacité d'abstraction	Apprend des patterns statistiques	Peut généraliser avec peu d'exemples
Besoin de connaissances de fond	Non	Oui, souvent

TABLE 1 – Comparaison entre l'apprentissage connexionniste et symbolique

3.2 Hypothèses et Contraintes

L'apprentissage de concepts repose généralement sur certaines hypothèses :

— **Hypothèse du bruit** : On suppose que les exemples d'entraînement sont **exacts**, c'est-à-dire qu'il n'y a pas d'erreurs ou d'ambiguïtés dans l'étiquetage des données.

— **Hypothèse du concept bien défini** : Il est supposé que le concept cible est bien défini et peut être appris à partir d'un ensemble fini d'exemples.

— **Hypothèse du monde fermé** : Tout ce qui n'est pas spécifiquement mentionné comme positif est supposé être négatif.

Cependant, ces hypothèses sont restrictives dans des contextes réels où les données peuvent contenir du bruit (exemples mal étiquetés, capteurs imprécis, etc.), et où le concept cible peut être plus complexe que prévu.

3.3 Illustration et Exemple

Exemple 1 : Reconnaissance des formes

Supposons que l'espace des instances X soit un ensemble de formes géométriques, et que le concept cible soit « les cercles ». Dans ce cas :

— Le concept C est défini comme :

$$C = \{x \in X \mid x \text{ est un cercle}\} \tag{3}$$

— Toutes les autres formes (carrés, triangles, etc.) appartiennent à l'ensemble négatif $X \setminus C$.

Exemple 2 : Classification des emails

Prenons un autre exemple où l'objectif est de classifier des emails comme *spam* ou *non-spam* (ham). Ici :

— L'espace des instances X représente tous les emails reçus.

— Le concept C correspond aux emails considérés comme **spam**.

— Le complément $X \setminus C$ contient tous les emails non-spam.

Dans un cas idéal, un algorithme d'apprentissage de concepts apprendrait une fonction f qui prédit correctement si un email est spam ou non, en fonction de caractéristiques comme la présence de certains mots-clés, la structure du message, ou l'expéditeur.

3.4 Lien avec les Hypothèses d'Apprentissage

L'apprentissage de concepts s'appuie sur des hypothèses fondamentales permettant de guider la généralisation :

— **Hypothèse de stationnarité** : Les distributions des instances positives et négatives restent stables dans le temps.

— **Hypothèse de représentativité** : Les exemples d'entraînement sont représentatifs du problème global.

— **Hypothèse d'absence d'ambiguïté** : Chaque instance appartient soit à la classe positive, soit à la classe négative, mais jamais aux deux en même temps.

4 Treillis des Espaces de Version

Dans l'apprentissage de concepts, l'espace des hypothèses forme généralement une structure ordonnée que l'on peut représenter sous forme de treillis. Cette organisation permet de mieux comprendre la manière dont un algorithme explore l'espace des hypothèses pour identifier le concept cible.

4.1 Définition de l'Espace de Version

L'**espace de version** (*version space*) est l'ensemble des hypothèses qui sont compatibles avec l'ensemble des exemples d'apprentissage observés jusqu'à présent.

$$VS = \{h \in H \mid h(x) = f(x) \text{ pour tous les exemples } x \text{ observés}\} \tag{4}$$

où :
— H est l'ensemble des hypothèses possibles.
— $h(x)$ est la prédiction de l'hypothèse h pour une instance x.
— $f(x)$ est l'étiquette réelle de l'instance x.
L'espace de version est donc constitué de toutes les hypothèses qui sont cohérentes avec les exemples d'apprentissage.

4.2 Ordre Général-Spécifique et Treillis

Les hypothèses dans H peuvent être ordonnées selon un **ordre général-spécifique** (*general-to-specific ordering*). On dit qu'une hypothèse h_1 est plus générale qu'une hypothèse h_2, noté :

$$h_1 \geq h_2 \tag{5}$$

si et seulement si h_1 couvre au moins toutes les instances couvertes par h_2 :

$$\forall x \in X, \quad h_2(x) = 10, \Rightarrow h_1(x) = 1 \tag{6}$$

Cet ordre induit une structure de **treillis**, où :
— L'hypothèse la plus générale (notée $\top$) accepte toutes les instances.
— L'hypothèse la plus spécifique (notée $\bot$) rejette toutes les instances.
— Chaque hypothèse est positionnée dans le treillis en fonction de sa généralité par rapport aux autres.

4.3 Bornes du Treillis : Hypothèses Maximale et Minimale

Dans l'algorithme d'élimination de version (*Version Space Learning*), on manipule deux ensembles d'hypothèses :
— **G-set (Général)** : ensemble des hypothèses les plus générales encore compatibles avec les exemples.

— **S-set (Spécifique)** : ensemble des hypothèses les plus spécifiques compatibles avec les exemples.

L'espace de version peut être vu comme l'ensemble des hypothèses situées entre ces deux bornes :

$$S \subseteq VS \subseteq G \tag{7}$$

5 Fondements Formels du Version Space Learning

L'apprentissage par VSL est une approche d'apprentissage symbolique qui repose sur le maintien d'un ensemble d'hypothèses compatibles avec les données observées. Cette méthode permet de restreindre progressivement l'espace des hypothèses jusqu'à l'identification d'un modèle unique (ou d'un sous-ensemble restreint de modèles).

5.1 Définitions et Notations

— Soit X l'espace des exemples et H l'espace des hypothèses.
— Une hypothèse $h \in H$ est une fonction de classification $h : X \to \{0, 1\}$, où $h(x) = 1$ signifie que x est classé comme positif.
— Un ensemble d'exemples d'apprentissage est donné sous la forme $D = \{(x_i, y_i)\}_{i=1}^{N}$, où $x_i \in X$ et $y_i \in \{0, 1\}$ est l'étiquette associée.
— L'ensemble des hypothèses compatibles avec D est appelé **Version Space** et est noté $VS(D)$.

Avant d'introduire les frontières de l'espace des versions, il est essentiel de comprendre la relation d'inclusion entre les hypothèses. Une hypothèse h est un sous-ensemble d'une autre hypothèse h' (noté $h \subset h'$) si et seulement si h impose des conditions plus strictes sur la classification des exemples que h'. Autrement dit, toute instance acceptée par h est aussi acceptée par h', mais l'inverse n'est pas nécessairement vrai. Cette relation d'inclusion est fondamentale pour définir les frontières de l'espace des versions.

Définition 1 (Frontières de l'espace des versions) *L'espace des versions est caractérisé par deux ensembles appelés* ***frontières*** *:*

— *La* ***frontière spécifique*** S *contient les hypothèses* ***les plus spécifiques*** *qui classifient correctement tous les exemples positifs observés :*

$$S = \{h \in H \mid h(x) = 1, \quad \forall (x, 1) \in D \text{ et } \nexists h' \in H, h' \subset h\}.$$

— *La* ***frontière générale*** G *contient les hypothèses* ***les plus générales*** *qui ne classifient incorrectement aucun exemple négatif :*

$$G = \{h \in H \mid h(x) = 0, \quad \forall (x, 0) \in D \text{ et } \nexists h' \in H, h \subset h'\}.$$

6 Exemple d'Espace des Hypothèses

Considérons une tâche de classification où chaque exemple est décrit par trois attributs :

— **Couleur** : Rouge (`R`) ou Bleu (`B`)
— **Forme** : Cercle (`C`) ou Carré (`S`)
— **Taille** : Grande (`G`) ou Petite (`P`)

L'espace des hypothèses H comprend toutes les combinaisons possibles des valeurs de ces attributs, y compris des hypothèses générales et spécifiques.

Hypothèse	**Couleur**	**Forme**	**Taille**
$h_{\text{gén}}$ (Hypothèse générale)	?	?	?
h_1	R	?	?
h_2	?	C	?
h_3	R	C	?
h_4	R	C	G
$h_{\text{spéc}}$ (Hypothèse spécifique)	R	C	G

TABLE 2 – Exemple d'espace des hypothèses H

L'hypothèse $h_{\text{gén}}$ représente le cas le plus général où tous les exemples sont acceptés, tandis que $h_{\text{spéc}}$ est le plus restrictif, n'acceptant qu'une seule configuration d'attributs.

6.1 Algorithme d'Apprentissage par Version Space

L'algorithme suit un processus itératif où il met à jour les ensembles S et G en fonction des exemples observés.

Algorithme 1 : Algorithme Version Space Learning

Input : Un ensemble d'exemples d'entraînement D avec attributs et classes

Output : Les ensembles S et G définissant l'espace des versions

Initialisation :
$S \leftarrow$ hypothèse la plus spécifique (ex : un ensemble vide ou une instance spécifique)
$G \leftarrow$ hypothèse la plus générale (ex : $(?, ?, ..., ?)$)

foreach *exemple* $(X, y) \in D$ **do**
  if *y est positif (exemple accepté)* **then**
    foreach $g \in G$ **do**
      if *g ne couvre pas X* **then**
        Retirer g de G
      end
    end
    foreach $s \in S$ **do**
      if *s ne couvre pas X* **then**
        Généraliser s au minimum pour couvrir X
      end
    end
    Retirer de S les hypothèses plus générales que d'autres
  else if *y est négatif (exemple rejeté)* **then**
    foreach $s \in S$ **do**
      if *s couvre X* **then**
        Retirer s de S
      end
    end
    foreach $g \in G$ **do**
      if *g couvre X* **then**
        Spécialiser g pour exclure X
      end
    end
    Retirer de G les hypothèses plus spécifiques que d'autres
  end
end
return S, G

6.2 Convergence et Propriétés

— L'algorithme garantit la convergence vers une unique hypothèse h^* si et seulement si $S = G$.
— Si S et G ne convergent pas à une unique hypothèse, cela signifie que les données disponibles sont insuffisantes pour identifier un modèle unique.
— L'espace des versions peut être utilisé pour générer de nouvelles requêtes ou poser des questions stratégiques afin de distinguer les hypothèses can-

didates.

7 Pourquoi Étudier le Version Space Learning ?

Même si les réseaux de neurones dominent aujourd'hui l'apprentissage automatique, il est essentiel de comprendre les approches **symboliques** pour plusieurs raisons :

1. **Explicabilité et Interprétabilité** : Contrairement aux modèles neuronaux, les méthodes comme *VSL et ILP* permettent d'expliquer facilement les décisions prises.
2. **Apprentissage avec peu de données** : Les méthodes symboliques nécessitent moins de données pour apprendre des concepts généraux.
3. **Utilisation de connaissances de fond** : Elles peuvent être intégrées avec des bases de connaissances préexistantes.
4. **Complémentarité avec l'IA moderne** : Aujourd'hui, on assiste à une renaissance des approches **neuro-symboliques**, combinant **réseaux de neurones et modèles symboliques**.

8 Applications et défis

Le Version Space Learning est particulièrement utile dans des domaines où l'on dispose de peu de données et où la généralisation doit être bien contrôlée. Parmi ses applications, on trouve :

— L'apprentissage de concepts en classification supervisée.
— L'extraction de règles explicables à partir de données.
— La planification et la robotique, où les contraintes de généralisation sont cruciales.

Cependant, la méthode souffre de limitations, notamment sa sensibilité aux erreurs et son inefficacité lorsque l'espace des hypothèses devient trop large. C'est là que l'ILP (Inductive Logic Programming) et d'autres approches plus robustes peuvent être envisagées.