Introduction aux Algorithmes d'Apprentissage

L'*intelligence artificielle* est un champ mal délimité, dans lequel peut entrer beaucoup, ou très peu, d'algorithmes et d'applications.

Il sera plus adéquat dans le cadre du programme de parler d'algorithmes d'*apprentissage automatique*.

❦ Définition.

Les *algorithmes d'apprentissage automatique* sont des algorithmes cherchant à apporter une réponse *générale* à un problème algorithme en exploitant un ensemble préalable de données *spécifiques*.

C'est la faculté de *généralisation à partir d'une somme finie d'expérience* qui donne le nom d'*apprentissage* à ces méthodes. Il ne s'agit pas d'affirmer ici que cela correspond au sens commun d'*apprentissage*, en particulier utilisé dans le cadre de l'espèce humaine.

Les problèmes algorithmiques peuvent être considérés et résolus de manière exacte, mais le plus souvent, les algorithmes d'apprentissage ne permettent que de résoudre des versions *approchées* de problème algorithme.

On distingue trois catégories d'algorithmes d'apprentissages :

1. les algorithmes d'apprentissage *supervisés* basés sur des données *étiquetées*.
2. les algorithmes d'apprentissage *non supervisés* basés sur des données non étiquetées.
3. les algorithmes d'apprentissage *par renforcement* basés sur une mesure et les retours d'un environnement. Dans ce cadre, les données sur lesquelles ont s'appuie sont construites à chaque boucle d'apprentissage avec la mesure des résultats préalablement obtenus. Cette catégorie n'est pas au programme.

I Algorithmes d'apprentissage supervisé

❦ Définition.

Un algorithme d'apprentissage *supervisé* est un algorithme qui s'appuie sur un ensemble de données *étiquetées*, *i.e.* des entrées du problème à résoudre dont on connaît déjà la sortie (la solution).

1. Les problèmes de *régression* cherchent à attribuer une valeur *quantitative* aux entrées.
2. Les problèmes de *classification* cherchent à attribuer une valeur *qualitative* aux entrées, une classe ou catégorie, *i.e* une valeur dans un ensemble fini.

1.1 Traitement préalables des données

Souvent, les données doivent être préalablement traitées et validées avant de pouvoir être utilisée par un algorithme d'apprentissage supervisé.

On sélectionne ensuite dans ces données deux échantillons :

1. Un échantillon d'*entraînement* utilisé par l'algorithme pour construire un *modèle* qui lui permettra de répondre.
2. Un échantillon de *test* utilisé par l'algorithme pour valider et corriger ses réponses.

L'échantillon de test est utile lors de la conception (par un être humain) de l'algorithme pour valider ou non le modèle produit par l'entraînement.

Il est essentiel de ne pas réutiliser les données d'entraînement dans l'échantillon de test. Au risque de valider un modèle qui répond spécifiquement aux données d'entraînement sans avoir réussi à généraliser ses réponses.

On choisit classiquement 80% des données étiquetées pour l'entraînement et 20% pour le test.

1.2 Taux d'erreur et Matrice de confusion

Une fois l'échantillon de test isolé, on peut estimer l'erreur du modèle.

On se place dans un cadre de *régression*, avec n valeurs de l'échantillon de test, étiquetées par n réels $\hat{y}_i$, $0 \leq i < n$.

❦ Définition.

L'erreur quadratique moyenne est la moyenne des différences des écarts au carré, entre les sorties de l'algorithme et les étiquettes, sur l'échantillon de test. *i.e.* si on note y_i, $0 \leq i < n$ les sorties de l'algorithme sur les n entrées de l'échantillon de test, l'erreur quadratique moyenne est :

$$\frac{1}{n}\sum_{i=0}^{n}(\hat{y}_i - y_i)^2$$

On se place maintenant dans un cadre de *classification* avec n valeurs de l'échantillon de test, étiquetées à l'aide de m étiquettes numérotée $\{0, \cdots, m-1\}$.

❦ Définition.

La *matrice de confusion* est la matrice $m \times m$ telle que le coefficient $y_{i,j}$ correspondent au nombre de données de l'échantillon de test, étiquetées par i sur lesquelles l'algorithme d'apprentissage a produit comme sortie l'étiquette j.

❦ Définition.

Le *taux d'erreur empirique* est la somme des coefficients non diagonaux de la matrice de confusion divisée par n, *i.e.* le nombre d'entrées de l'échantillon de test sur lequel l'algorithme d'apprentissage s'est trompé, divisé par le nombre d'entrée total dans l'échantillon.

1.3 Exemple - Algorithme des K-voisins

L'algorithme des k plus proche voisin (k nearest neighbors, *KNN*) est un algorithme supervisé le plus souvent utilisé dans le cadre de la classification, mais qui peut aussi être utilisé pour de la régression.

Il suppose qu'on dispose d'une distance sur l'ensemble des entrées et utilise les données de l'échantillon d'entraînement pour déterminer l'étiquette d'une entrée, e, à l'aide des étiquettes des k données les plus proches de e dans l'échantillon d'entraînement.

On se réduira à des entrées dans $\mathbb{R}^p$ muni de la distance euclidienne.

Il existe plusieurs stratégies de détermination de l'étiquette de l'entrée en fonction de celle de ses voisins.

Pour une *régression*, si chaque entrée de l'échantillon d'entraînement est étiquetée par un réel, on pourra considérer classiquement la *moyenne*, pondérée ou non, des étiquettes de ses plus proches voisins.

Pour une *classification*, on pourra considérer classiquement l'étiquette *majoritaire* parmi celle de ses voisins, avec pondération ou non. Les cas d'égalités peuvent être traitées de différentes manières le cas d'égalité.

k est donc un paramètre de l'algorithme à faire varier pour obtenir les meilleurs résultats. Prendre trop peu de voisin peu mener par exemple à un trop gros poids des cas particulier. Alors qu'en prendre trop ne permettra pas d'identifier les spécificités de la zone.

L'algorithme en pseudo-code s'écrit donc :

Algorithme KNN

entrées : k un entier naturel, E un échantillon d'entraînement, e une entrée

1. P ← l'ensemble des k plus proches voisins de e dans E.
2. **renvoyer** résolution de P

L'ensemble d'entraînement doit avoir une structure de tableau associatif : à une entrée, on associe son étiquette. Les entrées sont des points de l'espace, ils seront donc représentés par des tuples de valeurs numériques. Un dictionnaire ou une liste de couple est la structure adéquate.

Pour récupérer les k plus proches voisins de e dans E, on peut au choix :

1. Trier E par ordre de distance à e et prendre les k premiers éléments en $O(|E| \log_2(|E|))$.
2. Adapter un tri pour ne trouver que les k premiers éléments. Par exemple un tri par sélection ou insertion ce qui donne un $O(k|E|)$.

Dans le cadre on k reste petit (*e.g.* inférieur à 20) et E contient beaucoup de données (*e.g.* plus de un million), la deuxième solution est plus efficace. De manière générale, c'est le cas, dès que $|E| > 2^k$.

Voici une manière d'adapter un tri par insertion pour obtenir les k plus proches éléments. On peut

aussi voir ça comme une généralisation de l'algorithme de second minimum.

Algorithme calcul des k plus proches

entrées : k un entier naturel, E un échantillon d'entraînement, e une entrée

```
1.  P ← [(None, +∞)] * k
2.  pour tout x une clé de E faire
3.      d ← distance de e à x
4.      si  d < P[k − 1][1]  alors
5.          i ← k − 1
6.          tant que i > 0 et P[i − 1][1] > d  faire
7.              P[i] ← P[i − 1]
8.              i ← i − 1
9.          fin tant que
10.         P[i] ← (x, d)
11.     fin si
12. fin pour
13. renvoyer  P
```

La résolution se fait ensuite par un algorithme de moyenne ou de maximum, éventuellement pondéré, sur P.

II Algorithmes d'apprentissage non supervisé

Dans le cadre de l'apprentissage non supervisé, l'algorithme exploite la structure même des données, leurs relations internes et leur organisation, ainsi que des mesures de qualités des résultats obtenus pour produire le modèle.

La validation du modèle par l'humain s'obtient lui aussi *in fine* à l'aide d'une mesure de qualité.

2.1 Algorithme des k moyennes.

L'algorithme des k moyennes (ou \$k\$-*means*) est un algorithme de classification. Il s'agit de regrouper les données en k catégories déterminées par leur proximité dans un espace vectoriel réel de dimension fini arbitraire, muni d'une distance euclidienne.

L'algorithme produit k centroïdes dans l'espace de telle manière à ce que :

1. une entrée est dans la catégorie i si et seulement si le centroïde le plus proche est le centroïde i.
2. les k centroïde choisi minimisent localement l'erreur quadratique moyenne (*c.f.* définition dans la section apprentissage supervisé) dans chaque catégorie construite sur l'ensemble des données exploitées.

Pour construire les k centroïdes finaux, l'algorithme construit une suite de centroïdes en cherchant à diminuer à chaque étape l'erreur quadratique moyenne. Pour ce faire, étant donné k centroïdes :

1. on répartit les données dans les k catégorie correspondantes,
2. on calcule le barycentre de l'ensemble des données dans une même catégorie,
3. On définit les nouveaux centroïdes comme étant égaux aux barycentres calculés.

On réitère ce processus qui converge vers un minimum local, autant de fois que nécessaire. Étant donné la précision finie des flottants, on peut répéter le processus jusqu'à ce que les coordonnées des centroïdes n'évoluent plus.

Différentes méthodes existent pour initialiser les premiers centroïdes. On en citera deux :

1. la méthode de *Forgy* consiste à choisir aléatoirement k données différentes de l'ensemble des données à exploiter.
2. la méthode de *partition aléatoire* répartie aléatoirement les données à exploiter dans k catégories différentes et non vides, puis calculer les barycentres de ces catégories.

Ces méthodes aléatoires, mais exploitant tout de même la forme des données initiales, permettent d'éviter une initialisation dégénérée qui convergerait vers des minimums locaux très éloignés de la structure des données.

Le k est un paramètre ou bien connu si le nombre de catégories est imposé, ou bien que l'être humain doit adapter pour que le nombre de catégories correspondent au mieux à la structure des données, tout en permettant tout de même une généralisation.

Voici un pseudo-code de l'algorithme des k-moyennes qui calcule les centroïdes à utiliser pour catégoriser les entrées.

Algorithme K-means

ENTRÉES : k un entier naturel, E un ensemble de données (points de l'espace)

```
C ← la liste des k centroïdes initiaux
C' ← [None] * k
tant que C ≠ C' faire
    C' ← C
    CAT ← [∅] * k
    pour x ∈ E faire
        i ← l'indice du centroïde de C le plus proche de x
        ajouter x à CAT[i]
    fin pour
    C ← la liste des barycentres de CAT[i]
fin tant que
```