Principe de parcimonie (rasoir d'Ockham) en Statistique

Prise de notes pour l'exposé de Pierre W

9 avril 2017

1 Problème

1.1 Petites questions

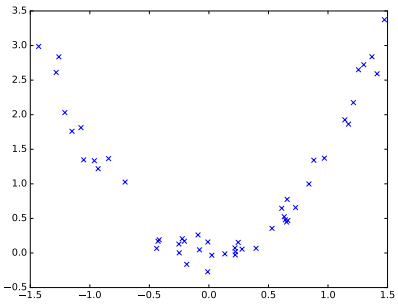
1. Problème de la suite logique:

   On se donne une suite, par exemple: $1, 1, 2, 3, 5, 8, \ldots$..Comment trouver le terme suivant ?

   On peut répondre 13 mais on peut aussi considérer le polynôme d'interpolation qui passe par tous les termes déjà considérés, ou bien considérer une suite périodique, etc.

2. Régression polynomiale:

   Étant donné un nuage de points on cherche le polynôme qui approxime le mieux ce nuage.

   Une question qui se pose est alors le degré du polynôme recherché. En effet un polynôme de haut degré passera par tous les points mais il est probablement préférable dans le cas à droite de prendre un polynôme de degré 2.

   De manière générale la problématique dans les deux problèmes précédents est la suivante:

   "Il y a quelque chose chose que l'on a envie de faire (trouver le modèle "simple") mais ce n'est pas évident de le décrire mathématiquement".

3. Compression de données.

   On considère une message, par exemple 001000000010000100 . . ..

   On observe qu'il y a "beaucoup de zéros". On peut alors "coder" les suites de zéros pour réduire la taille de la représentation.

4. Prédiction. À partir d'une suite de mesures, trouver la loi de prédiction qui a du sens.

On observe de la "régularité" et on cherche " l'essence " de la suite ou des données.

1.2 Régression Polynomiale

Considérons une ensemble de points (X_i, Y_i) avec $X_i \sim \mathcal{U}([0,1])$ et $Y_i = f(X_i) + \varepsilon_i$ où $f : [0,1] \to \mathbb{R}$ et $\varepsilon_i \sim \mathcal{N}(0, \sigma^2)$.

On cherche le polynôme P qui minimise

$$L(P) = \frac{1}{n} \sum_{i=1}^{n} \|P(X_i) - Y_i\|^2$$

On remarque que si on s'autorise tous les polynômes alors le minimum est 0 car on peut trouver un polynôme passant par tous les points.

Pour éviter ce problème on introduit une pénalité pour pénaliser les polynômes trop "complexes".

$$\tilde{L}(P_n) = l(P_n) + \text{pen}(P_n)$$

Par exemple:

- Pour éviter des polynômes de trop haut degré:

  $$\text{pen}(P_n) = \lambda \deg(P_n)$$

  avec $\lambda > 0$ un coefficient à choisir judicieusement.

- Pour éviter d'avoir trop de coefficients:

  $$\text{pen}(P_n) = \lambda \sum |a_i|$$

  où $P(x) = \sum a_i x^i$

- Ou encore la pénalité utilisée en ...:

  $$\text{pen}(P_n) = \frac{\ln(n)}{2} \deg(P_n)$$

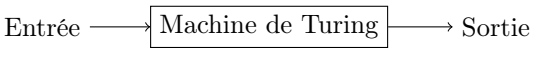
> Remarque
>
> Pour le dernier choix il n'y a pas de λ. La philosophie derrière le $\ln(n)$ et que quand on a beaucoup de données on cherche à les simplifier.

Petite devinette: trouver le meilleur choix pour le terme suivant de la suite: $1, 1, 2, 3$.

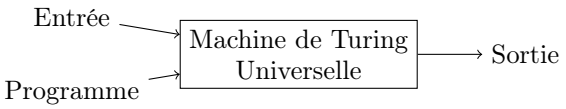
2 Machine de Turing

2.1 Machine de Turing

Une machine de Turing est un programme informatique (par exemple écrit en C++, en python,etc.) qui prend une entrée et renvoie une sortie.



Une machine de Turing universelle est l'équivalent d'un "langage de programmation". C'est à dire une machine de Turing prend en entrée un programme P et une entrée e et renvoie la sortie du programme en question pour l'entrer e.



2.2 Complexité de Kolmogorov.

Soit M une machine de Turing universelle. Étant donné un programme P sans entrée on note $s(p)$ sa sortie et $l(p)$ sa longueur.

Soit une suite de caractère x. On considère l'ensemble de tous les programmes P qui renvoie x. On définit alors la complexité de Kolmogorov comme la taille du plus petit de ces programmes:

$$\mathrm{K}(x) = \min(l(P) : s(P) = x)$$

Existe t il un programme qui calcule K ?

La réponse est **non**.

En effet supposons par l'absurde qu'il existe un tel programme et que sa longueur soit $k \in \mathbb{N}$.

Considérons alors le programme suivant:

```
def K(n):
        [Programme qui calcule K la complexite de Kolmogorov]

while K(n)< k+10000
        n=n+1
print(n)
```

On a donc écrit un programme qui contient moins de $k +$ quelques lignes et qui renvoie le plus petit n tel que $K(n) > k + 1000$. Or par définition un tel programme doit contenir au moins $k + 1000$ caractères. On aboutit donc à une contradiction.

> Remarque
>
> La complexité de Kolmogorov dépend du langage de programmation: et on aurait du écrire $K_M(x)$ avec M la machine de Turing universelle.

> Remarque
>
> Soient deux machines de Turing universelles M_1 et M_2, alors $\exists c$ tel que $K_{M_1}(x) \leq K_{M_2}(x)+c$.
> (Il suffit intuitivement d'écrire un compilateur de M_1 en langage M_2)
> Donc asymptotiquement le choix de la machine de Turing n'est pas important.

3 Induction de Solomonoff

3.1 Les statistiques bayésiennes.

On dispose une loi aléatoire dépendant d'un paramètre θ et renvoie $X_1, X_2, \ldots$.

Exemple : Des Bernoulli i.i.d $\mathbb{P}(X = 1) = \theta$, $\mathbb{P}(X = 0) = 1 - \theta$.

Formule de Bayes. On considère que le paramètre θ est lui même aléatoire selon une loi α (a priori).

Alors par la formule de Bayes:

$$\mathbb{P}(\theta|X_1,...,X_n) = \frac{\mathbb{P}(X_1,...,X_n|\theta)}{\mathbb{P}(X_1,...,X_n)}\mathbb{P}(\theta)$$

La question est alors de savoir ce que vaut α appelé l'a priori.

3.2 Suite de suite ?

Sachant $X_1, X_2, ..., X_n$ que vaut X_{n+1} ?

L'idée générale est la suivante: On peut estimer θ avec la formule de Bayes. Ensuite on tire X_{n+1} suivant la loi de paramètre θ.

3.3 Les a priori universels de Solomonoff

Soit x une suite de symboles. Et on associe à cette suite une probabilité d'arriver.

$$\mathbb{P}_2(x) \sim \sum_{P \text{ programme déterministe}} 2^{-l(P)} 1_{P \text{ renvoie } x}$$

> Remarque
>
> Attention il n'est pas évident que cette suite converge. C'est effectivement le cas si on écrit P seulement avec des 0 et 1.

On peut aussi définir d'autres probabilités:

$$\mathbb{P}_3(x) \sim \sum_{P \text{ programme aléatoire}} 2^{-l(P)} \mathbb{P}(P \text{ renvoie } x)$$

> Remarque
>
> Une machine de Turing aléatoire est une machine de Turing à laquelle on a rajouté une bande contenant des 0 et 1 aléatoires.

ou encore:

$$\mathbb{P}_4(x) \sim \sum_{\mu \text{ mesure de probabilité}} 2^{-|\mu|} \mu(x)$$

3.4 Petite application

$\mathbb{P}_4(X_{n+1}|X_1,...,X_n) = \frac{\sum_\mu W(\mu)\mu(X_{n+1}|X_1,...,X_n)}{\sum_\mu W(\mu)}$.

$W(\mu) = 2^{-|\mu|}\mu(X_1,...,X_n)$

La statistique Bayésienne ne renvoie pas un paramètre mais une statistique sur celui-ci.

Il apparaît ainsi un compromis entre la "simplicité" de la loi aléatoire donné par le terme de Kolmogorov et la "validation" des données donné par $\mu(X_1,...,X_n)$.

3.5 Autre petite application

Considérons une suite de 0 et de 1, par exemple 0011000101010....

On découpe cette suite par paire puis on réalise une statistique sur ces paires. Pour compresser ces données on peut utiliser les résultats de Shannon.

Par exemple:

Paire	00	01	10	11
Fréquence	2^{-1}	2^{-3}	2^{-3}	2^{-2}
Encodage	0	111	110	10

Dans le cas on a quelque chose comme $l(x) = \log_2(p(x))$ où $l(x)$ est la longueur x une fois compressé.

On peut introduire : $\log(W_\mu) = \mu - \log(\mu(X_1, ..., X_n))$.

Interprétation : on code la manière de décoder les données. Puis on code les données.

En pratique, c'est compliqué car la complexité de Kolmogorov est non calculable.

La précision optimale $\varepsilon^* \approx \sqrt{\frac{1}{J(\theta^*)}}$ où $J(\theta)$ est l'information de Fisher et θ^* est l'argument optimal pour θ.

4 Un cas pratique

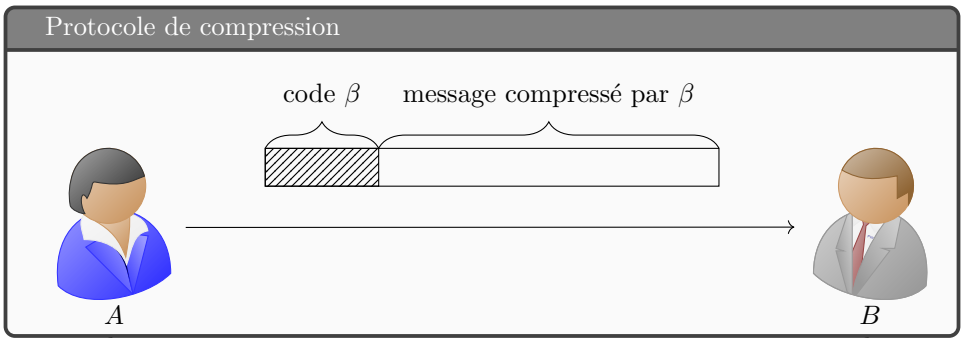
On cherche à minimiser β le postérieur Bayésien.

$$L(\beta) = -\mathbb{E}\left(\sum_{i=1}^{n} \log(\mathbb{P}(X_i)\right) + KL(\beta||\alpha)$$

KL : Kubac Labour

Exemple:

Alice souhaite envoyer un mail à Bob. On suppose qu'a priori A et B se sont mis d'accord sur un code α. En observant son message A s'est rendue compte de la présence de nombreuses redondances. L'idée est alors d'envoyer le mail avec un codage β tenant compte de ces redondances avec comme entête la manière de décompresser le message. Si le mail est très très long alors on choisit une manière de coder très sophistiquée tandis que si le mail est assez court on utilise un codage assez léger; c'est le sens du terme KL qui mesure une certaine distance de α à β.



Remarque

On a KL qui est une bonne mesure de la complexité, par exemple:

$$KL(\beta||\alpha) + \mathrm{K}(\alpha) \geq \mathrm{K}(\beta)$$