



M2 MPRI - Programmation Probabiliste - WebPPL

Christine Tasson - 10 décembre 2021

Webppl est un langage de programmation probabiliste basé sur Javascript. Pour programmer en ligne utiliser le site <http://webppl.org/> avec la documentation <https://webppl.readthedocs.io/en/master/>.

1 Dés

Utiliser WebPPL pour estimer quelle est la distribution de la somme de deux dés équilibrés, sachant que leur différence est égale à 1.

2 Pièces

1. Que fait le programme suivant ?

```
1 var model = function () {  
2   var x = sample(Uniform({a:0, b:1}))  
3   return (x<=0.1)  
4 }
```

2. On considère une pièce de monnaie biaisée de paramètre p .

Que fait l'algorithme suivant ?

- Lancer successivement la pièce deux fois, les lancers étant indépendants.
- Ensuite,
 - retourner 1 si la pièce a renvoyé pile puis face,
 - retourner 0 si la pièce a renvoyé face puis pile,
 - dans les autres cas, recommencer.

3 Militaires et Canabis

Une générale demande à un militaire s'il fume. Il lance une pièce

- si la pièce tombe sur *face*, alors il répond la *vérité*
- si la pièce tombe sur *pile*, alors il relance la pièce :
 - si la pièce tombe sur *face*, alors il répond *oui*
 - si la pièce tombe sur *pile*, alors il répond *non*

La générale qui ne sait pas combien de fois la pièce a été lancée. Avec 160 oui parmi 200, la proportion de fumeurs est de $p = 60\%$.

- Approcher la probabilité de fumer sachant que la réponse est oui.
- On a observé que le nombre de oui sachant le paramètre p suit une loi binomiale. Approcher la densité de probabilité du paramètre p représentant le nombre de fumeurs.

4 Régression linéaire

On se donne une liste de points $(0, 0)$ $(1, 2)$, $(4, 2)$, $(3, 6)$.

On cherche à trouver une droite $f : x \mapsto a * x + b$ qui passe proche de tous les points.

On suppose que les points observés sont de la forme $(x, f(x) + \epsilon)$ où ϵ est une erreur de mesure donnée sous la forme d'une gaussienne centrée en 0 et de variance 1.

Implémenter en WebPPL l'algorithme suivant :

- Échantillonner a d'une gaussienne centrée en 2 et de variance 1
- Échantillonner b d'une gaussienne centrée en 0 et de variance 1
- Pour chacun des points (x, y) , observer en y la loi normale centrée sur $f(x)$
- Inférer les densités des distributions de a et de b

Guide WebPPL

Instructions probabilistes

- Échantillonne une distribution de probabilité.

```
1 Sample(distribution)
```

- Infère la distribution de probabilité d'une fonction fn sans paramètre.

```
1 Infer({model:fn})
```

- Sélectionne les échantillons qui satisfont la propriété P de type bool passée en argument.

```
1 condition(predicate)
```

- Pondère les échantillons de la distribution D en fonction des données observées v (la loi de densité de D en v).

```
1 observe(distribution, value)
```

- Calcule l'espérance de la distribution passée en argument/

```
1 expectation(distribution)
```

Affichage

- Calcule la somme des cases d'un tabl

```
1 display(value)
```

- Trace l'histogramme ou la fonction de densité de la distribution passée en argument.

```
1 viz(distribution)
```

Distributions

- Bernoulli de paramètre p : pièce biaisée qui renvoie `true` avec probabilité p .

```
1 Bernoulli({p:...})
```

- Binomiale : nombre de "face" dans le tirage de n pièces biaisées de paramètre p

```
1 Binomial({p:... , n:...})
```

- Uniforme sur $[a, b]$: tire uniformément un réel entre a et b .

```
1 Uniform({a:... , b:...})
```

- Gaussienne de moyenne μ et de variation σ .

```
1 Gaussian({mu:... , sigma:...})
```

Tableaux

- Calcule la somme des cases d'un tableau t passé en argument.

```
1 sum(arr)
```

- Crée un tableau de taille n dont les cases sont remplies avec les résultats de la fonction sans paramètre fn passés en argument

```
1 repeat(n, fn)
```

- Applique la fonction fn à chacune des cases du tableau arr .

```
1 map(fn, arr)
```
