# Probabilité et modèle de langues

Guillaume Wisniewski
Université Paris Sud & LIMSI
guillaume.wisniewski@limsi.fr

8 janvier 2016

## 1 Estimation de probabilités

À partir du fichier `english.txt` disponible sur le site du cours [1] estimez la probabilité d'apparition de la lettre `a` dans un texte anglais. On veillera à supprimer du document tous les caractères qui ne sont pas des lettres de l'alphabet anglais « standard ».

On souhaite maintenant évaluer l'impact de la taille du corpus utilisé pour estimer la probabilité. Il suffit, pour cela, de limiter le corpus au $n$ premiers caractères. Représenter graphiquement la valeur de la probabilité estimée pour les valeurs suivante de $n$ : 100, 200, 300, 400, 500, 750, $10^3$, $1,5 \cdot 10^3$, $2 \cdot 10^3$, $5 \cdot 10^3$, $8 \cdot 10^3$, $10^4$, $5 * 10^4$, $10^5$, $10^6$, $10^7$, $10^8$. Que peut-on en conclure ?

Pour la représentation graphique, il est possible d'utiliser la bibliothèque `matplotlib` de la manière suivante :

```
# liste des abscisses
x = ...
# liste des ordonnées correspondantes
y = ...
import matplotlib.pyplot as plt
# échelle logarithmique pour les abscisses.
plt.semilogx(x, y, "x-")
plt.savefig("plot.png")
```

## 2 Génération de textes à l'aide d'un modèle de langue

L'objectif de cet exercice est de construire un programme capable d'écrire automatiquement des critiques de bouteille de vin comme celle que l'on peut trouver sur le site Wine Spectator en utilisant un *modèle de langue*.

1. https://perso.limsi.fr/wisniews/enseignement/15-16_proba_m1/

Un modèle de langue d'ordre $n$ permet de déterminer $p(w_i|w_{i-1}, w_{i-2}, ..., w_{i-n+1})$, la probabilité de trouver un mot $w_i$ après avoir observé les $n-1$ mots précédents et permet, entre autre, de déterminer la probabilité d'une phrase. Cette information est au cœur des systèmes de reconnaissance de la parole ou de traduction automatique. Elle peut également être utilisée pour générer des phrases similaires à celles produites par un humain.

## 2.1 Estimation du modèle de langue

Les probabilités du modèle de langue seront estimées à partir des critiques que l'on peut trouver sur le site Wine Spectator. Le programme python suivant permet de récupérer toutes les critiques du site :

```
#!/usr/bin/env python -tt

import urllib.request, urllib.error, urllib.parse
from lxml.html import fromstring
import sys
import time

import argparse

parser = argparse.ArgumentParser()
parser.add_argument("--output", required=True,
                    type=argparse.FileType("wt", encoding="utf-8"))

args = parser.parse_args()

urlprefix = "http://www.winespectator.com/dailypicks/category/catid/{}/page/{}"

for catid, max_page in [(1, 830), (2, 816), (3, 806)]:
    for page in range(1, max_page + 1):
        out = "-> On page {} of {}....      {:.2%}%"
        print(out.format(page, max_page, page / max_page))

        try:
            response = urllib.request.urlopen(urlprefix.format(catid, page))
            html = response.read()
            dom = fromstring(html)
            sels = dom.xpath('//div[(@class="paragraph")]')
        except Exception as e:
            print(e)
            continue
```

```
        for review in sels:
            if review.text:
                args.output.write("BEGIN NOW " + review.text.strip() + " END\n")
        time.sleep(2)
```

Ces critiques peuvent également être téléchargées sur le site du cours.

1. Soit $w_i$ le mot à la i$^{e}$ position d'une phrase. Comment est estimée la probabilité d'observer le mot $w_i$ connaissant les deux mots précédents $w_{i-1}$ et $w_{i-2}$ ?
2. Écrivez une fonction qui à partir d'une liste de mots renvoie la liste des triplets successifs : appelée avec la liste `["I" , "love", "chocolate", "ice-cream", "."]`, cette fonction renverra la liste `[("I", "love", "chocolate"), ("love", "chocolate", "ice-cream"), ("chocolate", "ice-cream", ".")]`.
3. Écrivez une fonction qui à partir d'une liste de triplets `(a, b, c)` estime les probabilités $p(c|a, b)$.
4. Le programme récupérant les critiques de vin, ajoute les deux mots `BEGIN NOW` au début de chaque critique. Pourquoi ? Pourquoi ajoute-t-on deux mots (et non 1 ou 5) ?

## 2.2 Génération

Une fois le modèle de langue estimé, il est possible de générer de nouvelles phrases de la manière suivante : le $i^{e}$ mot de la phrase est choisi aléatoirement selon la probabilité $p(w_i|h)$ où $h$ est un historique composé des deux mots $w_{i-1}, w_{i-2}$ choisis aux étapes $i-1$ et $i-2$ ; l'historique est ensuite mis à jour $h \leftarrow w_{i-1}, w_i$ et la procédure répétée jusqu'à ce que l'on génère le mot `END`.

Si une distribution $X$ générant des événements $a_i$ avec une probabilité $p_i$ est représentée par un dictionnaire dont les clés sont les $a_i$ et les valeurs associées les probabilités $p_i$, il est possible de générer un élément $a_i$ avec la probabilité $p_i$ en utilisant la fonction suivante :

```
import numpy as np

def sample_from_discrete_distrib(distrib):
    words, probas = list(zip(*distrib.items()))
    return np.random.choice(words, p=probas)
```

1. Comment doit-être initialisé l'historique ?
2. Implémentez cet algorithme. Que pensez-vous des phrases obtenues ?
3. Comment pourrait-on estimer la qualité des phrases produites ?