Modélisation mathématique

Le projet Verboïde

Nous allons construire une série de programmes automatiques qui nous permettrons d'analyser des textes français et anglais. Si tout se passe bien on pourrait caractériser les textes ou mieux encore catégoriser ceux-ci ou ultimement trouver les verbes et leurs conjugaisons les plus utilisées.

Sur le site :plouffe.fr/IUT/Modelisation maths/Projet Verboide/ vous trouverez des textes en grande quantité en français et anglais. Ce sont tous (ou presque) des romans, textes un peu techniques.

Le modèle que nous utiliserons sera d'abord fait de scripts Unix qui préparent un fichier texte suivi de filtres permettant de les classer.

Scripts utiles

La première chose à s'assurer est que le texte est en format approprié, la commande `file` permet d'identifier le type de document. Ici nous prendrons le fichier : Un frère de Nicolas Foucquet qui porte le numéro : 23199-8.txt de la liste

Exemple :

`file 23199-8.txt donne : 23199-8.txt: ISO-8859 text, with CRLF line terminators`

Ça signifie que le texte est en format MS-DOS il sera préférable de le transformer en format unix. Une des méthodes est d'utiliser la commande 'col –b'

On procède donc :

```
col -b  < 23199-8.txt > fichier1.txt
```

La commande tr permet de traduire un fichier en utilisant un filtre ici " " et "\012" transforment les espaces en saut de ligne.

```
tr " " "\012" < fichier1.txt > fichier2.txt
```

Nous avons maintenant un fichier contenant (en principe) 1 mot par ligne.

Nous allons maintenant trier le fichier

```
sort fichier2.txt > fichier3.txt
```

La commande uniq permet de compter les mots semblables, avec l'option –c ça donne le décompte de chaque mot.

On procède donc

uniq –c fichier2.txt > fichier3.txt

on retrie de fichier mais basé sur la colonne du décompte du nombre de mots.

sort –nb fichier3.txt > fichier4.txt

Et le résultat est que c'est assez décevant.

- On remarque que beaucoup de caractères indésirables se trouvent dans le décompte comme chaise et chaise, ou d'autres caractères comme - ; \, /, [ ], etc.
- Il faudrait les enlever.

Pour enlever les mauvais caractères on peut utiliser la commande sed. Sed est le 'stream editor' elle permet de modifier une chaîne de caractères (plutôt que seulement 1 avec tr).

Par exemple la commande

sed '1,$s/chaise/fauteuil/g' fichier4.txt > fichier5.txt change tout ce qui est chaise en fauteuil. Donc, pour enlever les , il faut taper

sed '1,$s/,//g' fichier4.txt > fichier5.txt on suppose ici qu'il n'y a pas de lignes contenant 2 mots séparés par une virgule. Sinon il faut filtrer AVANT le fichier pour transformer les , en espace.

Si on procède avec les autres caractères il faudra taper toutes les commandes une à une et ensuite utiliser une série de fichiers, 1,2,3,4,5,… ça fait beaucoup.

On peut faire plus court en spécifiant à sed que les commandes seront mises dans 1 fichier de commande comme suit. On crée le fichier 'commandesSED' qui contient

1,$s/,/ /g;

1,$s/=/ /g;

1,$s/\]//g' ici le caractère] doit être précédé de \.

…

…

Il suffira alors de taper

sed –f commandesSED fichier4.txt > fichier5.txt

Une fois le fichier nettoyé on peut le mettre en forme en utilisant sort et uniq –c.

Ensuite viennent les ‘noise words’ , les noise words sont les mots les plus souvent utilisés mais qui ne sont pas considérés comme du contenu à proprement parler comme de, la, les, mon, ma, etc. Noise words est ‘stopwords’ en français.

Voici une liste en anglais de ces mots :

about,after,all,also,an,and,another,any,are,as,at,be,because,been,before
being,between,both,but,by,came,can,come,could,did,do,each,for,from,get
got,has,had,he,have,her,here,him,himself,his,how,if,in,into,is,it,like
make,many,me,might,more,most,much,must,my,never,now,of,on,only,or,other
our,out,over,said,same,see,should,since,some,still,such,take,than,that
the,their,them,then,there,these,they,this,those,through,to,too,under,up
very,was,way,we,well,were,what,where,which,while,who,with,would,you,your,a
b,c,d,e,f,g,h,i,j,k,l,m,n,o,p,q,r,s,t,u,v,w,x,y,z,$,1,2,3,4,5,6,7,8,9,0,_

Exercice, vous allez chercher sur internet une bonne liste de mots de ce type :

Un bon point de départ est : http://eduscol.education.fr/cid47916/liste-des-mots-classee-par-frequence-decroissante.html

Ensuite vient la scriptologie, la science des scripts qui consiste à encapsuler toutes ces commandes dans 1 seule qui fera le travail (le plus automatiquement possible).

Pour rendre un fichier exécutable en Unix on tape :

chmod +x fichierdecommande

ensuite en tapant ./fichierdecommande ça exécute la série de commandes automatiquement.

Ces fichiers de commandes peuvent prendre en argument des paramètres qui sont nommés à l’interne du programme $1 $2 $3…

Par exemple pour les scripts vus plus haut on met dans 1 fichier

```
col -b  $1 > $1.a;
tr " " "\012" < $1.a > $1.b ;
sort $1.b > $1.c;
uniq –c $1.c $1.d;
sort –bn $1.d > $1.e;
```

Ensuite vient le moment de filtrer les fichiers pour obtenir des listes propres.

Pour filtrer, l'une des techniques consiste à utiliser un fichier permettant de capturer (soit positivement, soit négativement) le filtrage par ce fichier.

Par exemple on aurait

```
fgrep –f fichierfiltre fichier4.txt > fichier5.txt
```

et fichierfiltre contiendrait par exemple les conjugaisons du verbe aimer

```
aimerai
aimeras
aimera
aimerons
aimerez
aimeront
aimerais
aimerais
aimerait
aimerions
aimeriez
aimeraient
```

Si on veut filtrer négativement on peut écrire

```
fgrep –vf fichiernegatif fichier4.txt > fichier5.txt
```

Mais comment filtrer les mots stopwords tout en préservant les mots utiles ? Les stopwords sont habituellement courts, alors que faire ?

On peut décider de ne considérer les mots utiles que ceux qui sont de longueur 5 au moins et enlever ceux qui sont plus courts. Mais également on ne veut pas filtrer négativement les

acronymes utiles ou significatifs qui pourraient se trouver dans un texte comme SNCF, IBM CQFD, MDR, LOL, etc.

Donc ?, il faut filtrer de façon négative et positive selon le cas.

Référence:
Liste de mots 'noise' en anglais
https://www.drupal.org/node/1202
en français :
http://www.encyclopedie-incomplete.com/?Les-600-Mots-Francais-Les-Plus
Lisibilité d'un texte :
https://en.wikipedia.org/wiki/Flesch%E2%80%93Kincaid_readability_tests
https://en.wikipedia.org/wiki/Gunning_fog_index
Stopwords en français ici :
http://www.ranks.nl/stopwords/french

Modélisation mathématique

Le projet Verboïde (suite)

D'autres tables sont nécessaires.
Tout d'abord des verbes. Sur le site, une liste de verbes conjugués, on en compte plus de 10000 ici :
http://plouffe.fr/IUT/Modelisation%20maths/Projet%20Verboide/verbes.zip

Maintenant les listes de mots dans toutes les langues.
On en trouve un bon paquet ici
Mots en québécois : http://www.dictionnaire-quebecois.com/definitions-c.html
La liste la plus complète se trouve ici :
https://fr.wiktionary.org/wiki/Wiktionnaire:Listes_de_fr%C3%A9quence
ici la liste des 2000 mots les plus courants en français.

Pour ce qui est des accents, la fonction `iconv` de unix permet de passer d'un protocole à l'autre comme ISO-8859 à UTF-8, il est important d'avoir le même alphabet si vous traitez des textes automatiquement.

Pour notre modèle, il reste des tables à ramasser les nombreuses tables ici :
https://onedrive.live.com/?id=3732E80B128D016F%213584&cid=3732E80B128D016F
qui vous serviront dans la 2ème moitié du cours.