Data Science : Sante, Assurance, Finance

Reconnaissance de Langues Écrites

Étudiants :
MOMBO IVOLOU Vhiny-Guilley
TRAORE Babou

Encadré par :
Pr. Christophe AMBROISE

December 19, 2021

Contents

1 Introduction

En vue de classifier des textes bruts dont certain en Anglais et d'autres en Français, nous allons implémenter deux algorithmes, l'un de Bayes et l'autre de Markov.
Afin de construire ces deux algorithmes, notre plan est de considérer que les densités conditionnelles aux classes, appartiennent à une famille de densité définies par peu de paramètres. Notre travail se structure autour des étapes suivantes :

- le choix d'un modèle ;
- l'estimation des paramètres de ce modèle;
- l'implémentions du classifieur du modèle.

L'idée principale de la dernière étape revient à approximer les probabilités à posteriori par :

$$\hat{\pi}(k|x) = \frac{\hat{p}_k . f_k(x|\hat{\theta}_k)}{\sum_{l=1}^{K} \hat{p}_l . f_l(x|\hat{\theta}_l)}$$

Pour finir, nous allons créer un court texte d'au plus 1000 caractères et utiliserons l'algorithme de Viterbi pour identifier les langues d'où proviennent les lettres qui constituent le dit texte et l'algorithme de Baum-Welch pour estimer les paramètres.

2 Exercice 1 : Base de textes

2.1 Base de fichiers de textes bruts

Notre base de textes est constituée de textes extraits issus de diverse sources et sont de différentes natures allant des poèmes aux narrations en passant par discours philosophiques et politiques de personnalités célèbres. La base contient 40 textes dont la moitié en anglais et l'autre moitié en français. Lors de la création du dataframe que nous utilisons, nous avons stocké les textes dans une variable 'textes', ajouté une colonne 'lang' qui correspond à la langue du texte et une autre colonne 'label' qui prend la valeur -1 si le texte est en anglais et 1 sinon. Vous trouverez un résumé de se tableau ci-dessous.

```
liste_texte.head(5)
```

	textes	label	lang
0	Je ne sais si je dois vous entretenir des prem...	1	Francais
1	Toujours et Jamais étaient toujours ensemble\n...	1	Francais
2	Les mécanismes qui régissent le climat d'une p...	1	Francais
3	Un village écoute désolé\nLe chant d'un oiseau...	1	Francais
4	La trompe de l'éléphant,\nc'est pour ramasser ...	1	Francais

Figure1 : Échantillon de la liste de textes

2.2 Tableau de fréquence des symboles

Avant le comptage des symboles qui sont contenus dans le texte, nous procédons à un nettoyage du texte en trois principales étapes : la tokenisation, le traitement de la ponctuation et la normalisation.
Pour ce faire, nous avons crée une fonction "transTokeNorm" (Tokénization,Traitement et Normalisation) qui regroupe toutes ces opérations.
Nous avons remplacé les “NA” par des zéros que nous avons ensuite traiter en prenant (1+ matrice de comptage).
Nous optons pour une séparation des textes en lettres de l’alphabet uniquement, enlevant ainsi tous les espaces, toutes les ponctuations et accents. Toute fois, la fonction implémentée permet à un utilisateur de procéder autrement.

Une fois les textes nettoyés, nous procédons au comptage des symboles. Pour ce faire ,nous avons également implémentée une fonction appelée “freqUnigrams”qui utilise la fonction de nettoyage au préalable.

Une fois le décompte de symboles fait, nous avons consigné les résultats dans le tableau ci-dessous.

```
[10]: X_1j.head(10)
```

	a	b	c	d	e	f	g	h	i	j	...	r	s	t	
0	0.058404	0.004287	0.017976	0.017507	0.132994	0.012334	0.006186	0.006660	0.062896	0.014689	...	0.042977	0.082866	0.057503	0.0
1	0.080680	0.010595	0.024548	0.016432	0.090457	0.002364	0.005900	0.003544	0.069703	0.019919	...	0.058604	0.077399	0.073008	0.0
2	0.064925	0.007474	0.027852	0.038100	0.141717	0.010099	0.011845	0.005721	0.046981	0.001764	...	0.059118	0.059949	0.052440	0.0
3	0.067304	0.005051	0.026235	0.025001	0.145798	0.008822	0.011328	0.013828	0.060178	0.008822	...	0.042138	0.061369	0.061369	0.0
4	0.064661	0.019456	0.034753	0.029044	0.132303	0.005877	0.003922	0.013659	0.038540	0.001963	...	0.057268	0.068338	0.044195	0.0
5	0.061607	0.004065	0.027718	0.036789	0.137546	0.010535	0.008921	0.005686	0.057769	0.005686	...	0.054689	0.059690	0.057000	0.0
6	0.066356	0.006218	0.025656	0.019558	0.112920	0.006218	0.010341	0.005184	0.060502	0.003114	...	0.062457	0.056581	0.064409	0.0
7	0.049683	0.010667	0.026457	0.033741	0.158812	0.012257	0.008011	0.004814	0.050703	0.000536	...	0.047129	0.064874	0.069386	0.0
8	0.063995	0.015531	0.017242	0.027446	0.114880	0.008658	0.008658	0.012100	0.059089	0.018950	...	0.070499	0.057448	0.045887	0.0
9	0.061810	0.004677	0.025458	0.024544	0.137333	0.012579	0.006076	0.005144	0.056507	0.004211	...	0.063131	0.071459	0.053845	0.0

10 rows × 27 columns

Figure 2 : Extrait du tableau des occurrences

2.3 Histogramme des log-fréquences

Voici les histogrammes que nous avons obtenu après le calcul des logarithmes de (1 + fréquences):

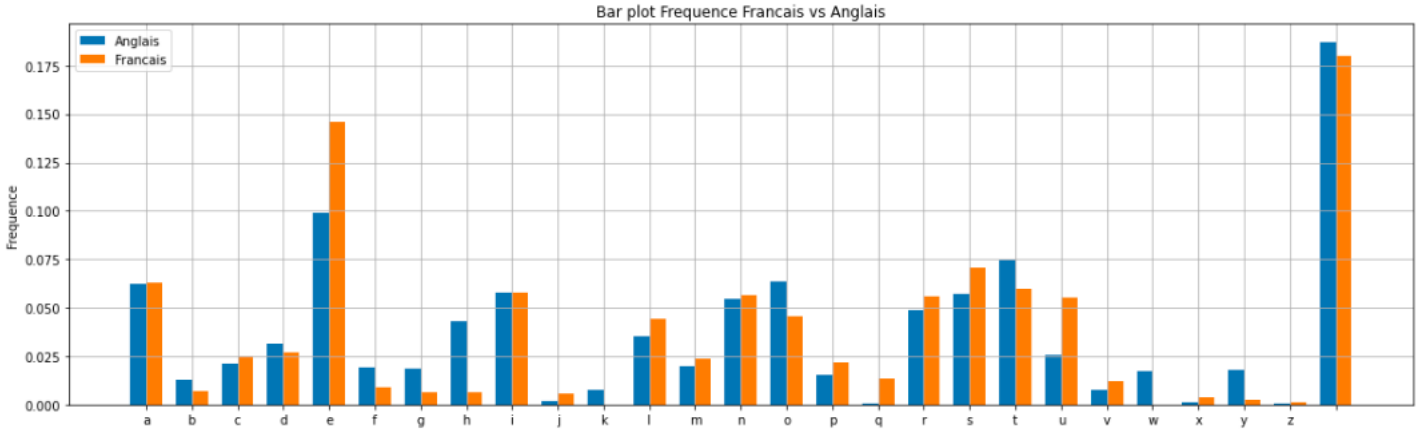


Figure 3 : Distributions des caractères Francais vs Anglais

Premièrement quand on observe la figure, ce qui nous tape à l’oeil est qu’il y a beaucoup de symbole

d'espace et de "e" dans les deux groupes de textes. Par contre c'est pas suffisant pour comparer les histogrammes des deux groupes de textes. Aussi en tant qu' utilisateur régulier de la langue. On a des priori sur la fréquences des caractères. En effet, on remarque que le "e", "l", "m", et le "q" sont très fréquent en français par rapport à l'anglais. De même les lettres "t", "h", "y", "w", "g" ont des fréquences très élevées en anglais par rapport au français. Cependant on voit que la log fréquences des lettres 'k', 'w' est quasi- nulle sur notre échantillon de texte en francais ce qui n'est pas surprenant au vu du nombre du faibles nombres de mots qui contient ces caractères en francais officiel 538 pour 'w' et '1625' pour 'k' sur les 103000 mots de langue francais (source : même site), contre 73106 pour 'w' et '77911' pour k en anglais sur les 270000 courants de la langue.

D'après ces points nous concluons que les deux groupes de lettres sont répartis différemment.

3 Classifieur de Bayes Naif

Supposons que :

$$f_k(x|\theta_k) = \mathcal{N}_p(x; \mu_k, \sigma_k^2 I_p)$$

$\{\mu_k\}$: le vecteur des moyennes;
$\{\sigma_k^2\}$: le vecteur des variances;
x : vecteur des log fréquences;
$k \in /Anglais, Francais/$.

3.1 Estimation des Moyennes et des Variances des classes

L'estimation des paramètres est faite par maximum de vraisemblance et donc le développement conduit à calculer juste les moyennes arithmétiques et les variances pour les symboles et par classe. Pour cela, la fonction "estim.parametres" a été implémentée et les résultats présentés en dessous:

	Mean.English	Mean.French	Variance.English	Variance.Français
a	0.057197	0.063681	0.000101	0.000059
b	0.012509	0.009364	0.000016	0.000027
c	0.020476	0.024358	0.000036	0.000022
d	0.030342	0.026506	0.000042	0.000034
e	0.092614	0.126923	0.000060	0.000317
f	0.019699	0.009792	0.000016	0.000011

figure 4 : Extrait du tableau des Paramètres

3.2 Évaluation des performances par validation croisée

Nous évaluons la performance du modèle par un "kfold cross validation". On utiliser k = 5 folds pour faire l'étude. On remarque que tous nos indicateurs (accuracy, precision, recall et f1) sont au égaux à 1. Une interprétation de cela est que nos textes sont suffisant long pour l'algorithme de Naives voit suffisamment de caratères dont la fréquence est représentative de la langue. Pour rendre cela plus challengeant on a édité des textes courts qui pourrait tromper l'algorithme. Un exemple est 'I really appreciate.' contre 'I really appreciate you.', l'algorithme prédit le premier comme étant du français et le deuxième étant de l'anglais. On estime que le présence du 'y' dans la deuxième phrase change la donne.

4 Classifieur de Markov

Supposons que :

$$f_k(x|\theta_k) = \mathcal{MC}(x; \pi_k, A_k)$$

A_k : Matrice de transition d'un symbole à un autre;
π_k : vecteur de probabilité des états initiaux;
x : vecteur des log fréquences;
$k \in \{1, 2\}$.

4.1 Estimation des paramètres des classes

L'estimation de ces paramètres est faite en comptant le nombre de chaque symbole respectif et en divisant chacun de ces nombres par la somme des occurrences dans un unigram pour les probabilités des états initiaux et dans un bigram pour les matrices de transition. Pour cela, la fonction "estim.parametres" a été implémentée et les résultats présentées en dessous:

index	Francais	Anglais
a	0.062709	0.062137
b	0.007044	0.012689
c	0.025001	0.020922
d	0.026935	0.031189
e	0.146103	0.098702
f	0.009082	0.019372
g	0.006457	0.018210
h	0.006147	0.043055
i	0.057702	0.057633
j	0.005905	0.001501

Figure 5 : Extrait des probabilités des états initiaux

```
[39]: pd.DataFrame(A['Anglais']).head(10) # matrice de transition anglais
```

[39]:

	a	b	c	d	e	f	g	h	i	j	...	r	s	t
a	0.000764	0.021390	0.049656	0.022154	0.082506	0.016043	0.012987	0.082506	0.012223	0.004584	...	0.066463	0.027502	0.033613
b	0.076389	0.017361	0.003472	0.003472	0.020833	0.003472	0.003472	0.013889	0.017361	0.003472	...	0.020833	0.003472	0.010417
c	0.128821	0.002183	0.006550	0.004367	0.102620	0.002183	0.002183	0.002183	0.150655	0.002183	...	0.019651	0.058952	0.019651
d	0.070149	0.001493	0.001493	0.022388	0.200000	0.001493	0.001493	0.001493	0.049254	0.001493	...	0.055224	0.001493	0.001493
e	0.001938	0.031492	0.026163	0.038760	0.030039	0.019864	0.024225	0.203488	0.019864	0.005814	...	0.120640	0.043605	0.064922
f	0.018779	0.002347	0.002347	0.002347	0.039906	0.042254	0.004695	0.002347	0.096244	0.002347	...	0.014085	0.004695	0.007042
g	0.057214	0.002488	0.002488	0.017413	0.024876	0.002488	0.017413	0.002488	0.087065	0.002488	...	0.019900	0.002488	0.004975
h	0.003279	0.001093	0.075410	0.002186	0.005464	0.001093	0.064481	0.001093	0.001093	0.001093	...	0.002186	0.054645	0.534426
i	0.027961	0.015625	0.013980	0.034539	0.018914	0.032895	0.010691	0.077303	0.000822	0.000822	...	0.068257	0.052632	0.105263
j	0.017544	0.052632	0.017544	0.017544	0.017544	0.017544	0.017544	0.017544	0.017544	0.017544	...	0.017544	0.017544	0.017544

10 rows × 27 columns

Figure 6 : Extrait Matrice de Transition (Anglais)

```
pd.DataFrame(A['Francais']).head(10) # matrice de transition francais
```

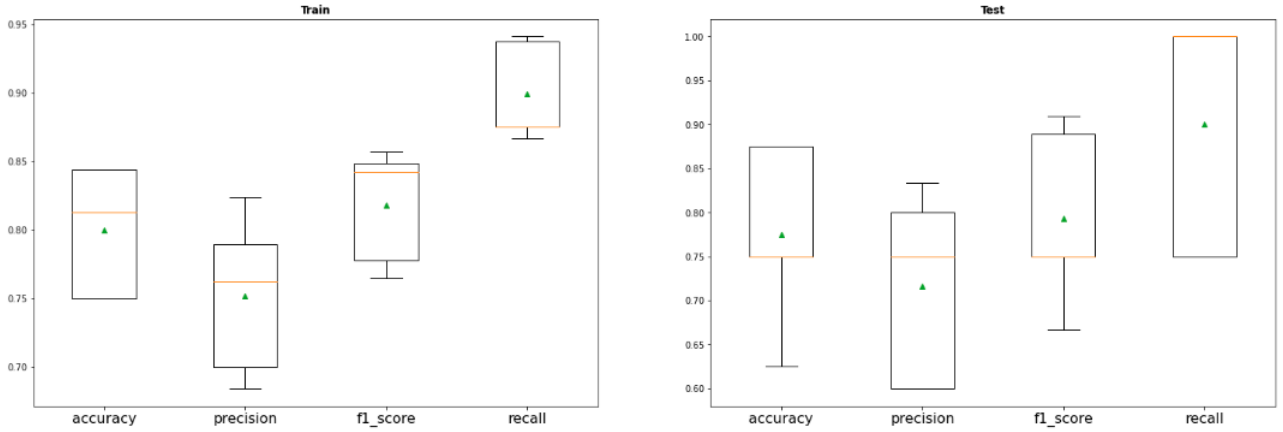
	a	b	c	d	e	f	g	h	i	j	...	r	s	t
a	0.001086	0.011944	0.028773	0.030945	0.022801	0.040174	0.016830	0.024430	0.007600	0.015744	...	0.108578	0.044517	0.070575
b	0.143478	0.004348	0.004348	0.004348	0.026087	0.004348	0.004348	0.004348	0.069565	0.004348	...	0.060870	0.004348	0.004348
c	0.077333	0.001333	0.014667	0.001333	0.125333	0.001333	0.001333	0.001333	0.033333	0.001333	...	0.040000	0.014667	0.005333
d	0.017370	0.001241	0.001241	0.001241	0.022333	0.001241	0.001241	0.001241	0.029777	0.001241	...	0.037221	0.001241	0.001241
e	0.000470	0.008692	0.049096	0.106883	0.010571	0.007517	0.016913	0.015269	0.047921	0.019027	...	0.114635	0.069298	0.076110
f	0.041522	0.003460	0.003460	0.003460	0.096886	0.076125	0.003460	0.003460	0.034602	0.003460	...	0.041522	0.006920	0.003460
g	0.173709	0.004695	0.004695	0.004695	0.122066	0.004695	0.004695	0.004695	0.089202	0.004695	...	0.070423	0.004695	0.004695
h	0.004902	0.004902	0.568627	0.019608	0.009804	0.004902	0.004902	0.004902	0.004902	0.004902	...	0.004902	0.004902	0.009804
i	0.210961	0.010018	0.021803	0.041249	0.020625	0.017678	0.013553	0.012964	0.000589	0.002946	...	0.071892	0.065999	0.073070
j	0.005076	0.015228	0.005076	0.005076	0.030457	0.005076	0.005076	0.005076	0.005076	0.005076	...	0.005076	0.005076	0.005076

10 rows × 27 columns

Figure 7 : Extrait Matrice de Transition (Français)

4.2 Évaluation des performances par validation croisée

Nous évaluons la performance du modèle par un "kfold cross validation".Nous avons pris k = 5 pour illustrer les résultats.



Figures 7 : Performances du classifieur

La performance globale du modèles est autour de 80 % . Contrairement au résultat précédent avec Naives Bayes, on fait plus d'erreur. On pourrait expliquer cela avec des pincettes, en disant que l'enchaînement des caractères entre les 2 langues n'est pas un facteur plus discriminant des 2 langues. On a essayé de voir sur quelles textes il performait moins bien on a les textes suivants :

```
erreur de prediction sur les textes 8 qui est texte Francais
erreur de prediction sur les textes 11 qui est texte Francais
erreur de prediction sur les textes 20 qui est texte Anglais
erreur de prediction sur les textes 31 qui est texte Anglais
erreur de prediction sur les textes 33 qui est texte Anglais
erreur de prediction sur les textes 38 qui est texte Anglais
erreur de prediction sur les textes 39 qui est texte Anglais
```

Figures 8 : erreur de predictions

Et sur les séquences 'I really appreciate.' et 'I really appreciate you.' , le classifieur Markovien fait la même erreur que Naives Bayes, c'est-à-dire, il prédit Francais pour le premier et Anglais pour le deuxième.

5 Exercice 4 : Décodage de langue par Viterbi

Pour cet algorithme, nous avons besoins d'une matrice d'émission des observations, d'une matrice de transition d'une langue à l'autre et des proportions de chaque classe dans l'ensemble des données. La matrice d'émission sera obtenue en calculant les probabilités stationnaires à partir des matrices de transition entre lettres obtenue à l'exercice précédent. La matrice de transition d'une langue à l'autre sera fixée. A ce niveau, nous fixons à 95% la probabilité de passer de l'anglais à l'anglais et à 5% la chance de passer de l'anglais au français. Également, les mêmes probabilités sont fixées pour la transition du français vers le français et du français vers l'anglais respectivement. Pour finir, au texte aléatoire crée, nous avons crée un vecteur de labels correspondant, question de les comparer avec les labels prédits par Viterbi.

5.0.1 Méthode

Pour créer ce court texte, on a créer une courte fonction 'generate_texte' qui prend en paramètre la liste de texte, et un nombre limite de caractère nchar. Cette fonction sélection un texte aléatoirement. Puis dans ce texte, elle sélection une phrase prises aléatoirement aussi et la concatène à une chaîne de caractère initialement vide. Ce processus est répété tant que la longueur de la chaîne de caractère ainsi créer est reste inférieure à nchar.

5.0.2 Résultat

L'application de notre fonction 'generate_texte, nous donne le résultat ci-dessous.

```
random.seed(115)
text_gen, df_text_gen, nombre_char = generate_texte() ## generation du texte avec des passsages en anglais et en francais
print('il y a : ',Counter(df_text_gen['lang'])) # nbre de texte en francais vs nbre de texte en anglais
print('Il y a {} caractères.'.format(nombre_char))
```

```
il y a :  Counter({'Francais': 6, 'Anglais': 6})
Il y a 920 caractères.
```

```
print(text_gen) # texte générer
```

```
Y a
t-on réussi.
Au bout du petit matin, cette ville plate étalée.this was shot generationally
or by degree, a quotient of time, drawn
up by the window with you excluded,
you emended from classification.qui le manifestent.Aftershock and jaw-ache,
a stranger wandering a house in tightening circles.
They call her a young country, but they lie:
She is the last of lands, the emptiest,
A woman beyond her change of life, a breast
Still tender but within the womb is dry.L'énergie reçue
par m² dépend de l'angle des rayons solaires par rapport au sol.
you could cut your foot on that accurate division.Mais le père fut sage
De leur montrer avant sa mort
Que le travail est un trésor.There's not a word from feathers or thinly shaped bones
brought about by crashing through muddy grounds.Foudroyés aujourd'hui
par la force mécanique, nous pourrons vaincre dans l'avenir par une force mécanique
supérieure.The Only White Landscape is an assembly point.
```

Figure 9 : Texte généré

Notre nouveau texte est constitué de 920 caractères avec 6 phrases en français et 6 phrases en Anglais. Notre algorithme va essayer de prédire lesquelles sont en 'Francais' et en 'Anglais'.

5.1 Algorithme de Viterbi

Cet algorithme permet de reconstituer les états cachés en dessous des observations en procédant en deux étapes qui sont :

- Forward :
  $V_{k,1} = \pi_k \times \phi_k(x_1)$
  pour $t \geq 2$
  $V_{l,t} = \max_k\{V_{t-1,k} \times A_{k,l} \times \phi_l(x_t)\}$
  $S_{t-1}(l) = \underset{k}{argmax}\{V_{t-1,k} \times A_{k,l} \times \phi_l(x_t)\}$
- Backrward :
  $\hat{Z}_T = argmax\{V_{t,k}\}$
  pour tout t<T
  $\hat{Z}_t = S_t(\hat{Z}_{t+1})$

$K = 2$
$T = 1000$
$(\pi_k)_{k\in[1,K]}$: contient les proportions de chaque de langue;
$(\hat{Z}_t)_{t\in[1,T]}$: les états cachés des T caractères;
$(x_t)_{t\in[1,T]}$: le vecteur de caractères;
$(A)_{K\times K}$: matrice de transition;
ϕ : matrice d'émission.

5.2 Etats cachés prédits

En appliquant l'algorithme sur notre texte, on obtient une performance de 86.52 %. Pour visualiser les résultats on réutilise notre codage -1 pour l'anglais et 1 pour le francais, le tout est affiché sur un graphique sur lequel on a en axe des abscisses l'indice du caractère dans la phrase.

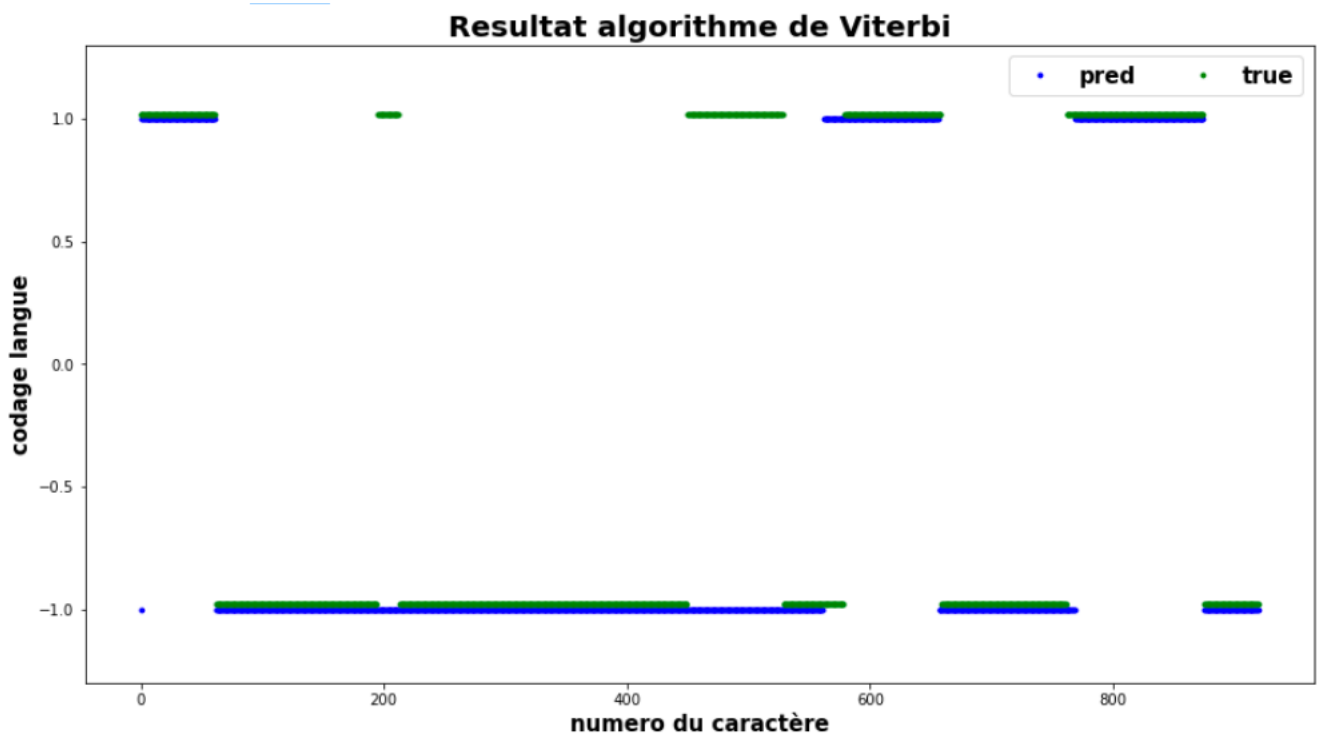


Figure 9 :Prédiction de l'algorithme de Viterbi

Après avoir lancé l'algorithme plus fois sur différents textes générés, nous remarquons l'algorithme prédit plus ou moins des longues séquences et fait des erreurs de prédiction en début de séquence. Cependant lorsque des courtes séquences d'une langue s'insère entre 2 longues séquences d'une autre l'algorithme tant

à ne pas les détecter.

5.3 Algorithme de Baum Welch

- Rappels théoriques: Nous supposons dans cette partie que la logarithme de vraisemblance de nos observations est donnée par

$$log(P(X_{1:T}, Z_{1:T}; \theta)) = \sum_k 1_{(Z_1=k)} \times log(\pi_k) + \sum_t \sum_{j,k} 1_{(Z_{t-1}=j, Z_t=k)} \times log(A_{j,k}) + \sum_t \sum_k 1_{(Z_t=k)} log(\phi_t(k))$$

et son esperance sous conditionnelle qu'on note

$$Q(\theta, \theta^q)$$

est égale à :

$$\sum P(Z_1 = k|X_{1:T}) \times log(\pi_k) + \sum \sum P(Z_{t-1} = j, Z_t = k|X_{1:T}) \times log(A_{j,k}) + \sum \sum P(Z_t = k|X_{1:T}) \times log(\phi_t(k))$$

- E step : On calcule

Forward :

$$\alpha_t \propto \phi_t \odot (A^T \alpha_{t-1})$$

Backward:

$$\beta_{t-1} \propto A(\phi_t \odot \beta_t)$$

En posant :

$$\gamma_t(j) = P(Z_t = k|X_{1:T}) \propto \alpha_t(j)\beta_t(j)$$

et

$$\zeta_{t,t+1}(i, j) = P(Z_{t-1} = j, Z_t = k|X_{1:T}) \propto \alpha_t(j)\phi_t(j)\beta_{t+1}(j)Ai, j$$

- M step : On réestime comme suite

$$\hat{\pi_i} = \gamma_1(i)$$

$$\hat{A}_{i,j} = \frac{\sum_t \zeta_{t,t+1}(i, j)}{\sum_t \gamma_t(j)}$$

$$\hat{B}_t(x_k) = \frac{\sum_{t, X_t = x_k} \gamma_t(j)}{\sum_t \gamma_t(j)}$$

5.3.1 Algorithme de Baum-Welch à deux états cachés

Dans cette partie nous avons choisi d'utiliser une petite partie (soit 25%) du texte obtenu dans la partie qui précède (algorithme viterbi) pour l'estimation des paramètres (matrice de transition, matrice d'émissions et la probabilités des états initiaux) par algorithme de Baum-welch. Après la convergence l'algorithme nous déterminons les états cachés de chaque passage. Nous avons fait ce choix pour éviter un problème des zéros qui apparaissent après quelques itérations de la partie Forward et vu que les partie sont inter-liées alors elles seront toutes affectées par ces valeurs de alpha. En effet, vu que nous faisons des multiplications des nombres avec virgules et compris entre 0 et 1, les alpha devient très vite nulles (une suite compris entre 0 et 1 décroissante convergente vers zéro). Une solution pour résoudre ce soucis consisterai à appliquer la fonction logarithme et apporter quelques modifications dans l'algorithme.

5.3.2 Résultat

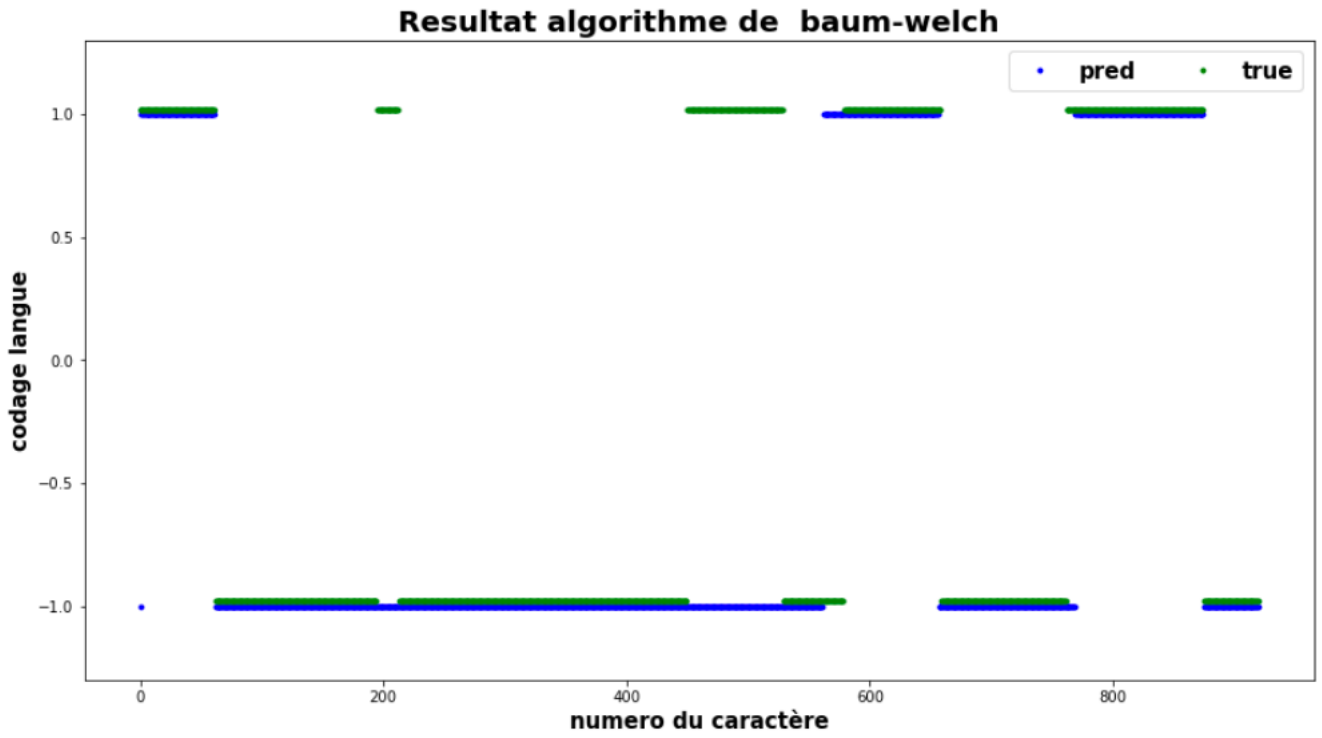


Figure 10 : Resultat Baum-Welch

Sur le graphe ci-dessus nous pouvons constater que l'algorithme de Baum- Welch a ré-estimé les paramètres de la chaîne et ces permettent d'obtenir une performance de 75.6 % qui est acceptable. Nous restons prudent dans nos interprétation car le texte utilisé n'est pas représentatif.