

Détection des spam dans les emails

PRÉSENTÉ PAR MÉLODY, ZOHRA & IMEN

SOMMAIRE





- ANALYSE DES DONNÉES
- VISUALISATIONS
- DÉFINIR L'EMBEDDING
- DÉMONSTARTION DES DIFFÉRENTS MODÈLES
- INTERFACE UTILISATEUR
- CONCLUSION

Contexte

APERÇU DE L'ENCADREMENT DU PROJET

Nous travaillons dans l'équipe data d'une agence data spécialisée en natural language processing (NLP). Notre agence vient de gagner un appel d'offres avec 6 projets différents.

Pour répondre à cet appel d'offre, le chef data officer a créé 6 équipes avec un chef de projet. Notre équipe est composée d'un data analyst, un data scientist et un data engineer.

		Sinn x	cosn x	tann x
1.0000 1.1052 1.2214 1.3499 1.4918	1.0000 .90484 .81873 .74082 .67032	.00000 .10017 .20134 .30452 .41075	1.0000 1.0050 1.0201 1.0453 1.0811	.00000 .09967 .19738 .29131 .37995
1.6487 1.8221 2.0138 2.2255 2.4596	.60653 .54881 .49659 .44933 .4065	.52110 .63665 .75858	1.1276 1.1855 1.2552 1.3374 1.4331	.46212 .53705 .60437 .66404 .71630
2.7183 3.0042 3.3201 3.6693 4.0552	53 660	1.1752 1.3356 1.5095 1.6984 1.9043	1.9 2.15	.76159 .80050 .83365 .86172 .88535
4.4817 4.9530 5.4739 6.0496 6.6859	2313 20190 8268 6530 4957	2.1293 2.3756 2.6456 2.9422 3.2682	2.352 2.577 2.828 3.107 3.41	.90515 .92167 .93541 .94681 .95624
7.3891 8.1662 9.0250 9.9742 11.023	34	3.6269 4.0219 4.4571 4.0370	3.72 3.5569	.96403 .97045 .97574 .98010 .98367
12.182 13.464 14.880 16.445 18,174		7.4063 8.1919 9.0596	6.1323 6.7690 7.4735 8.2527 9.1146	.98661 .98903 .99101 .99263 .99396
20.086 22.198 24.533		10.018 11.076 12.246 13.538 14.965	10.068 11.122 12.287 13.575 14.999	.99505 .99595 .99668 .99728 .99777
		16.543 18.285 20.211 22.339 24.691	16.573 18.313 20.236 22.362 24.711	.99818 .99851 .99878 .99900 .99918

Andlyses des données



5796 Lignes



3 Colonnes



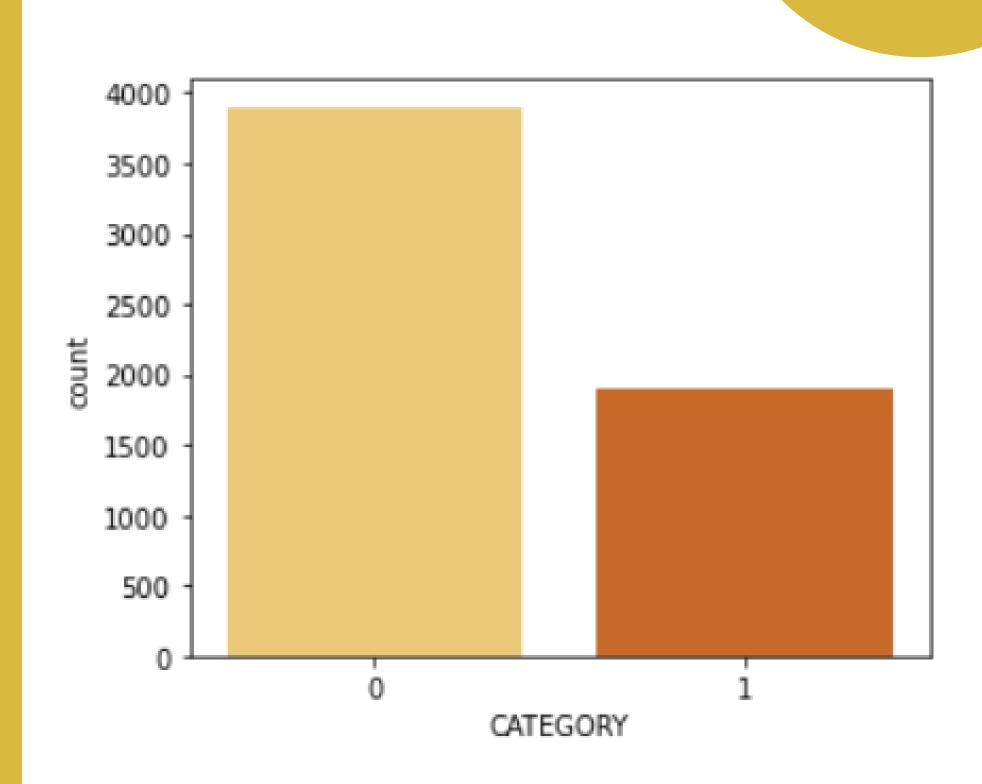
0 Valeurs manquantes



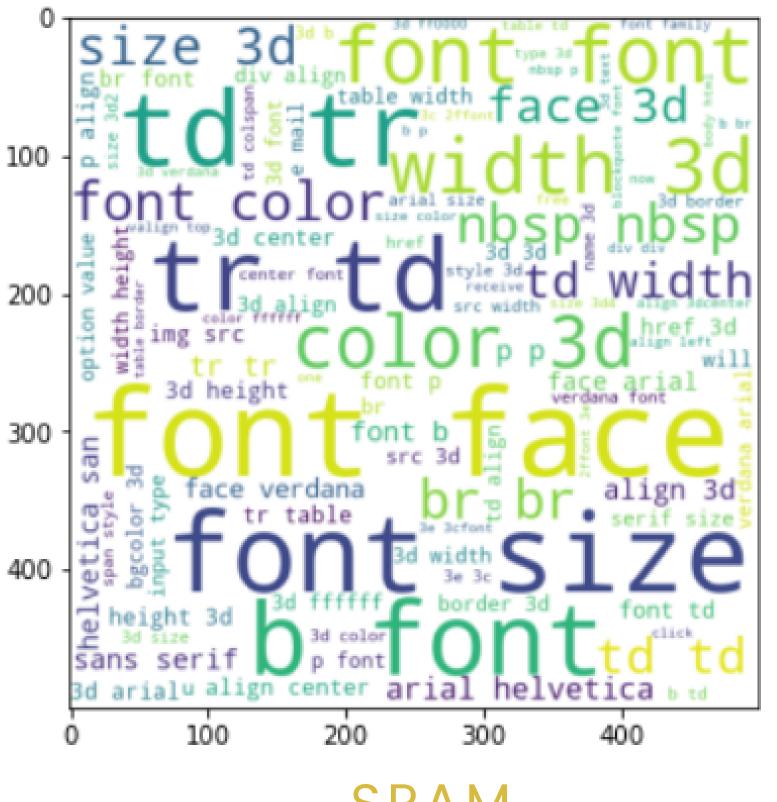
0 Duplicates

67.3% de Ham 32.7% de Spam

RÉPARTITION DES DONNÉES



NUAGES DE MOTS



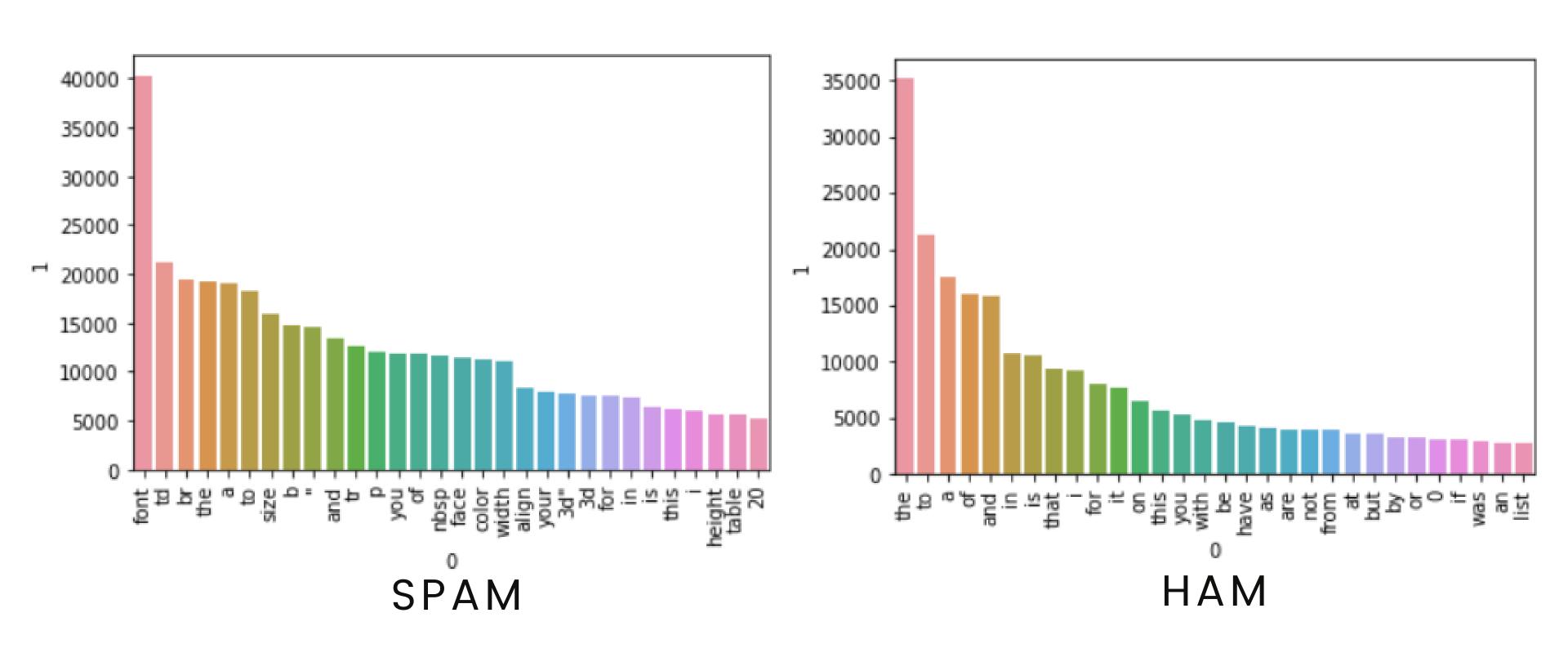


SPAM

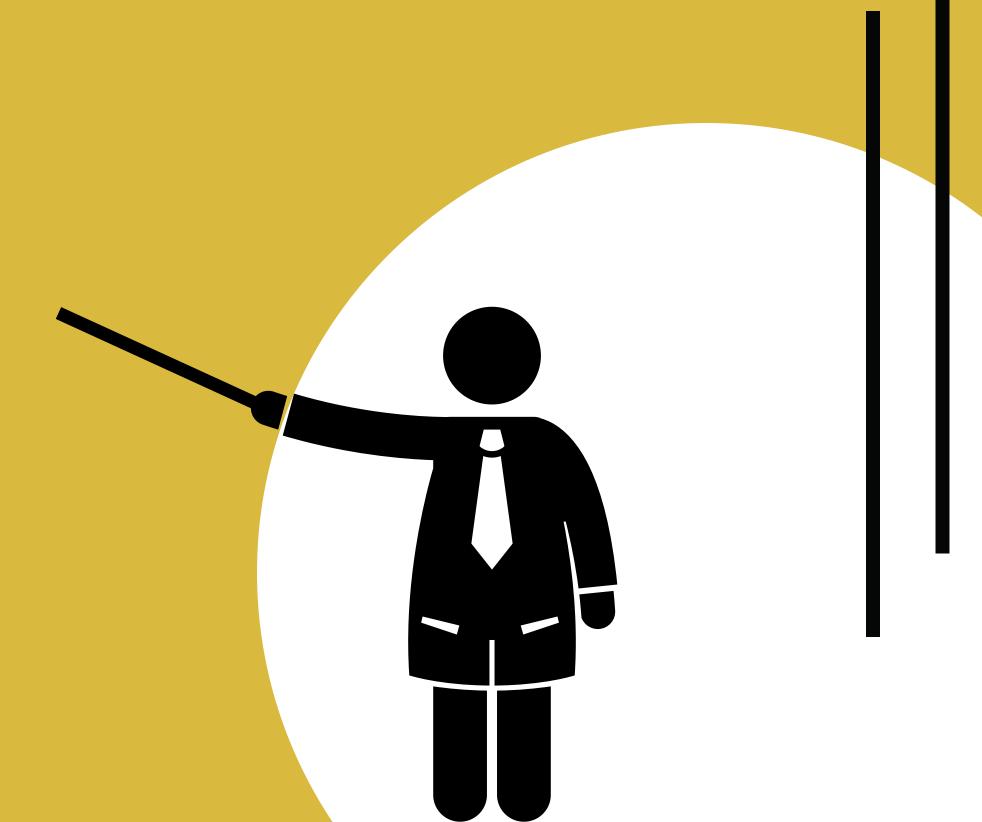
HAM

BARPLOT

Répartitions des 20 premiers mots sur chaques catégories



Explication du fonctionnement de l'Embedding



Illustration









Shrek

Incredibles The Triplets of Belleville

Harry Potter

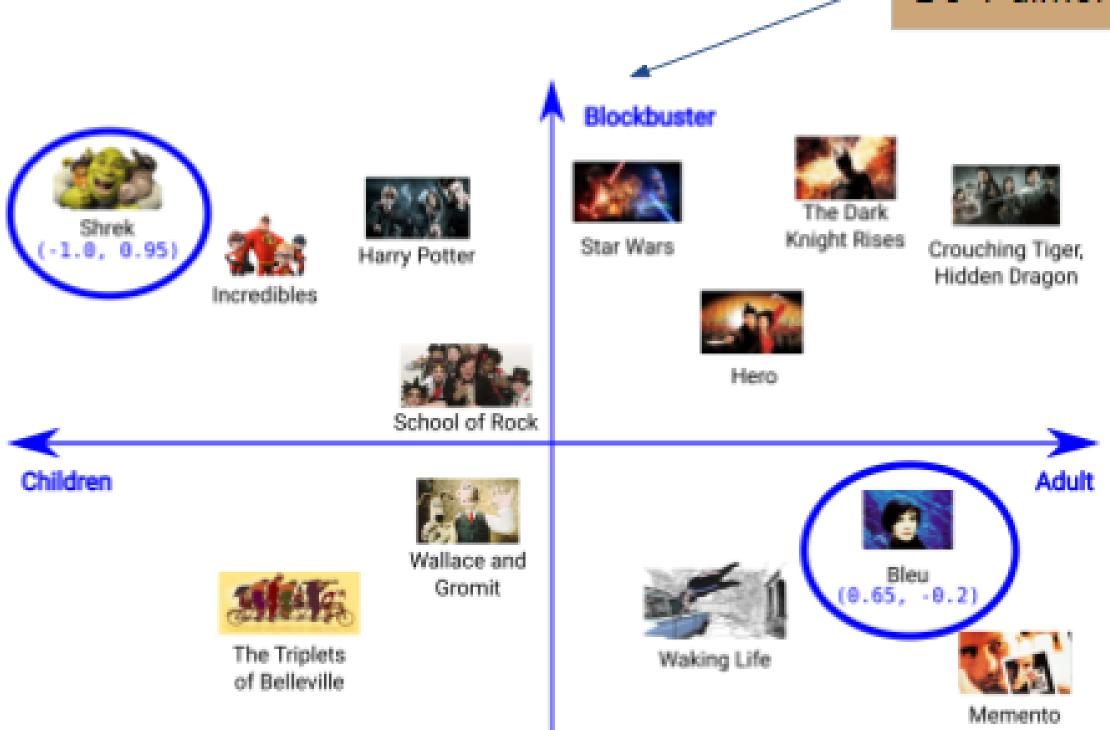
Star Wars

Bleu

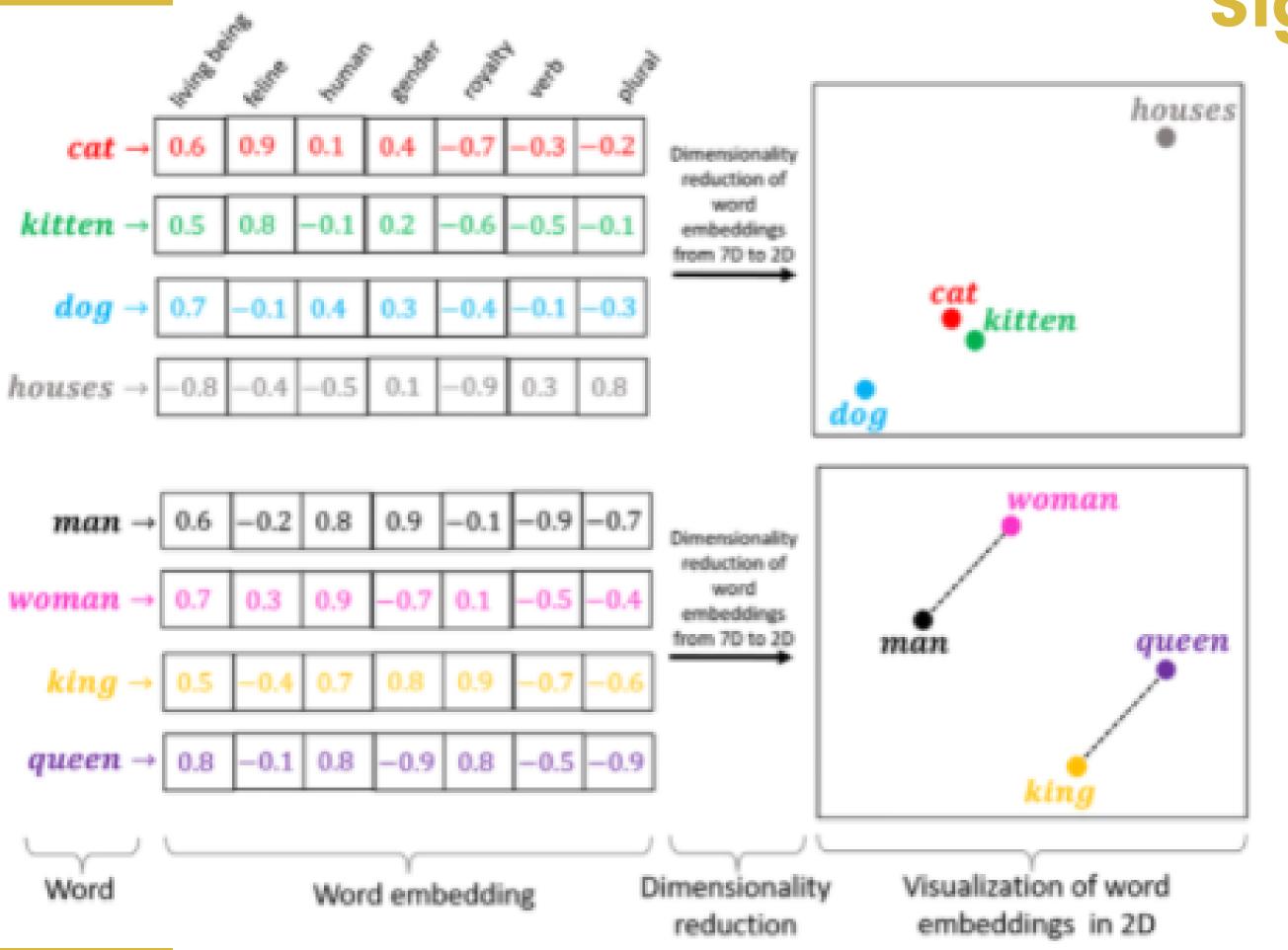
The Dark Knight Rises

Memento



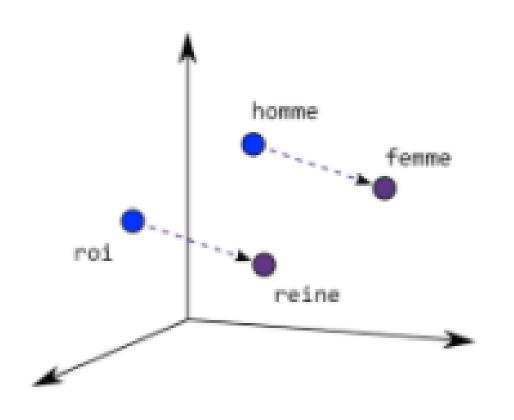


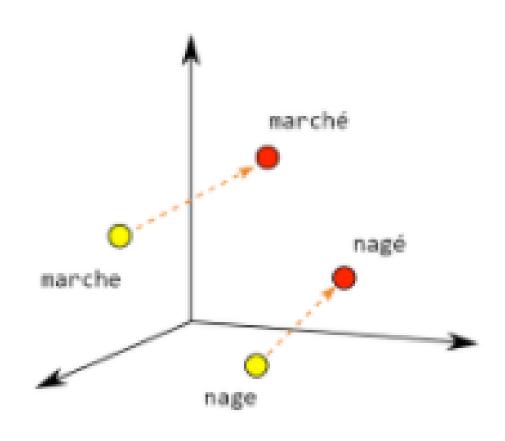
Arthouse

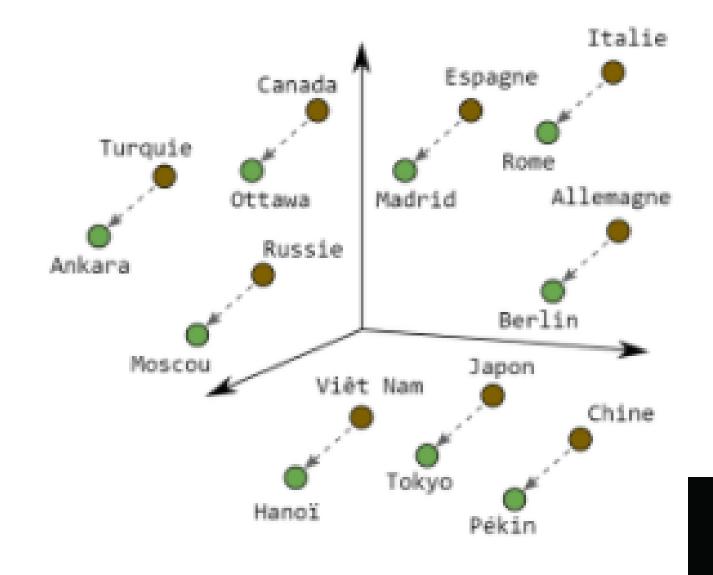


Signification des dimensions

Comparaisons selon les dimensions







Homme - Femme

Temps du verbe

Pays - Capitale



DÉFINIR CE QU'IL SE PASSE

Exemple mots :	BOnjour	Monsieur	je	vous						
	Hello	dear	monsieur	je	vous	contact	pour			
Tokenizer :										
Monsieur		1			5 1	2	3			
Hello		4								
Bonjour		5		2	4 8	1	2	3	6	7
dear		8								
je		2								
vous		3								
contact		6								
pour		7		Changement d	le dimension :					
				Monsieur	0.3	0.1	0.4	0.3	0.9	
				Hello	-0.3	0.7	0.8	0.1	0.03	

Avant le changement de dimension les vecteurs ont la même shape

Exemple mots:	BOnjour	Monsieur	je	vous						
	Hello	dear	monsieur	je	vous	contact	pour			
Tokenizer :										
Monsieur	1			í	5 1	2	3		0	0 0
Hello	4									
Bonjour	5			4	4 8	1	2		3	6 7
dear	8									
je	2									
vous	3									
contact	6									
pour	7			Changement de dimension :						
				Monsieur	0.3	0.1	0.4	0.3	0.9	
				Hello	-0.3	0.7	0.8	0.1	0.03	

Première itération modèle avec Embedding

COUCHES MODÈLE

```
inputs = tf.keras.Input(shape=(14804,))

embedding = tf.keras.layers.Embedding(
    input_dim=30000,
    output_dim=64
)(inputs)

flatten = tf.keras.layers.Flatten()(embedding)
  outputs = tf.keras.layers.Dense(1, activation='sigmoid')(flatten)
  model = tf.keras.Model(inputs=inputs, outputs=outputs)

#compile
model.compile(
    optimizer='adam',
    loss='binary_crossentropy',
    metrics=[
        'accuracy',
        tf.keras.metrics.AUC(name='auc')
    ]
)
```

MODÈLE SUMMARY

```
Model: "model"
                             Output Shape
 Layer (type)
                                                        Param #
 input 1 (InputLayer)
                              [(None, 14804)]
 embedding (Embedding)
                              (None, 14804, 64)
                                                        1920000
 flatten (Flatten)
                              (None, 947456)
 dense (Dense)
                             (None, 1)
                                                        947457
Total params: 2,867,457
Trainable params: 2,867,457
Non-trainable params: 0
None
```

Résultats :

Test Loss: 0.0222

Test Accuracy: 99.37%

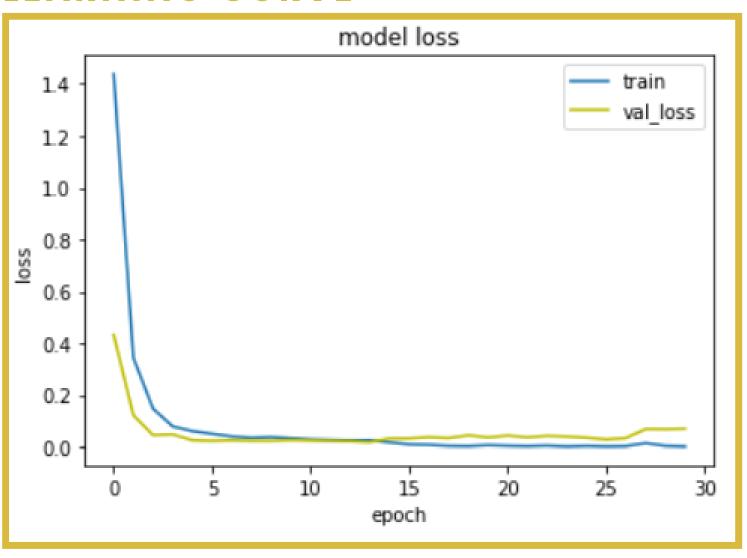
Test AUC: 0.9989

Seconde itération modèle avec Embedding

COUCHES MODÈLE

```
model = Sequential()
#Convolution
model.add(Embedding(input dim = 30000,
                    output dim=64,
                    input shape=input_shape))
#Flatten
model.add(Flatten())
#Couches denses
model.add(Dense(16, activation='relu'))
model.add(Dropout(0.2))
model.add(Dense(1, activation='sigmoid'))
#Modele compile
model.compile(optimizer='adam',
              loss='binary crossentropy',
              metrics=['accuracy', 'AUC'])
#Modele fit
history = model.fit(X train, y train,
                    validation split=0.2,
                    batch size=32,
                    epochs=30)
```

LEARNING CURVE



Résultats :

Test Loss: 0.0693

Test Accuracy: 99.10%

Test AUC: 0.9946

Modèle de Machine Learning avec LightGBMClassifier

STANDARDISER

TFIDFVECTORIZER
(NGRAM_RANGE=1,3)

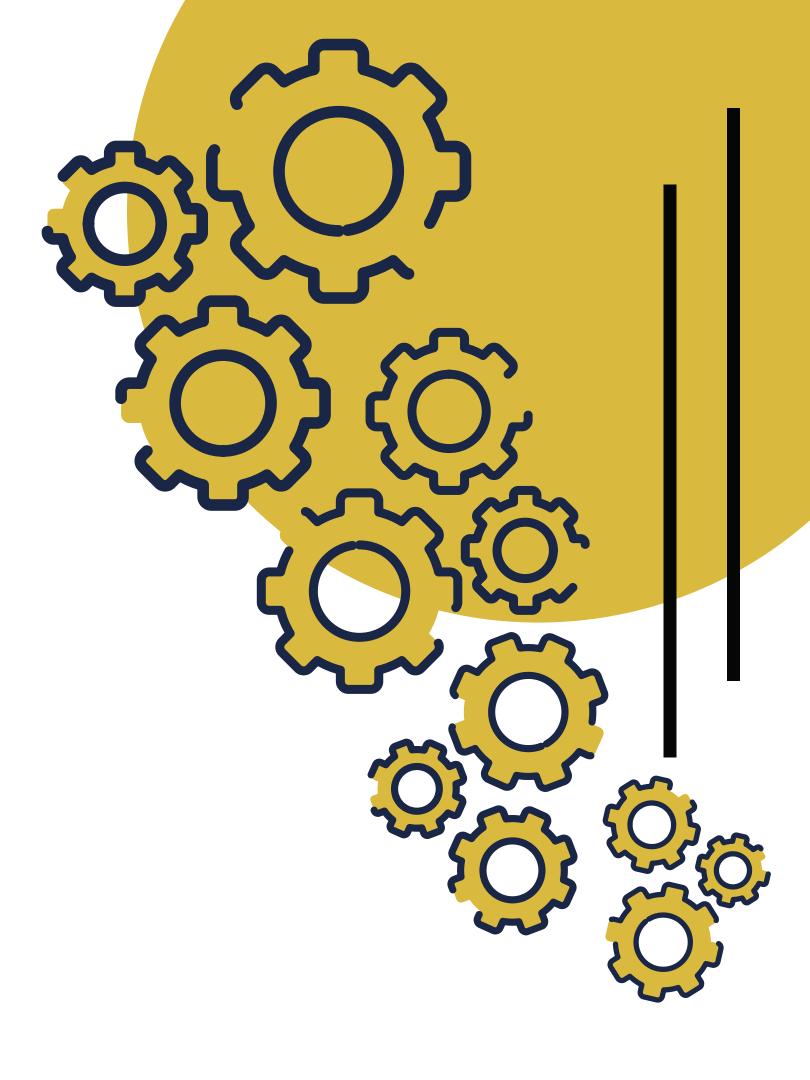
NLTK

WORDNETLEMM ATIZER()

LIGHTGBM CLASSIFIER

Démonstration de l'interface utilisateur





CONCLUSION S

Axes d'améliorations

PREMIER AXE

Réussir à améliorer l'application

DEUXIÈME AXE

Déploiement sur Streamlit en réglant le problème des "versions"

TROISIÈME AXE

Meilleur nettoyage des mots pour améliorer notre corpus