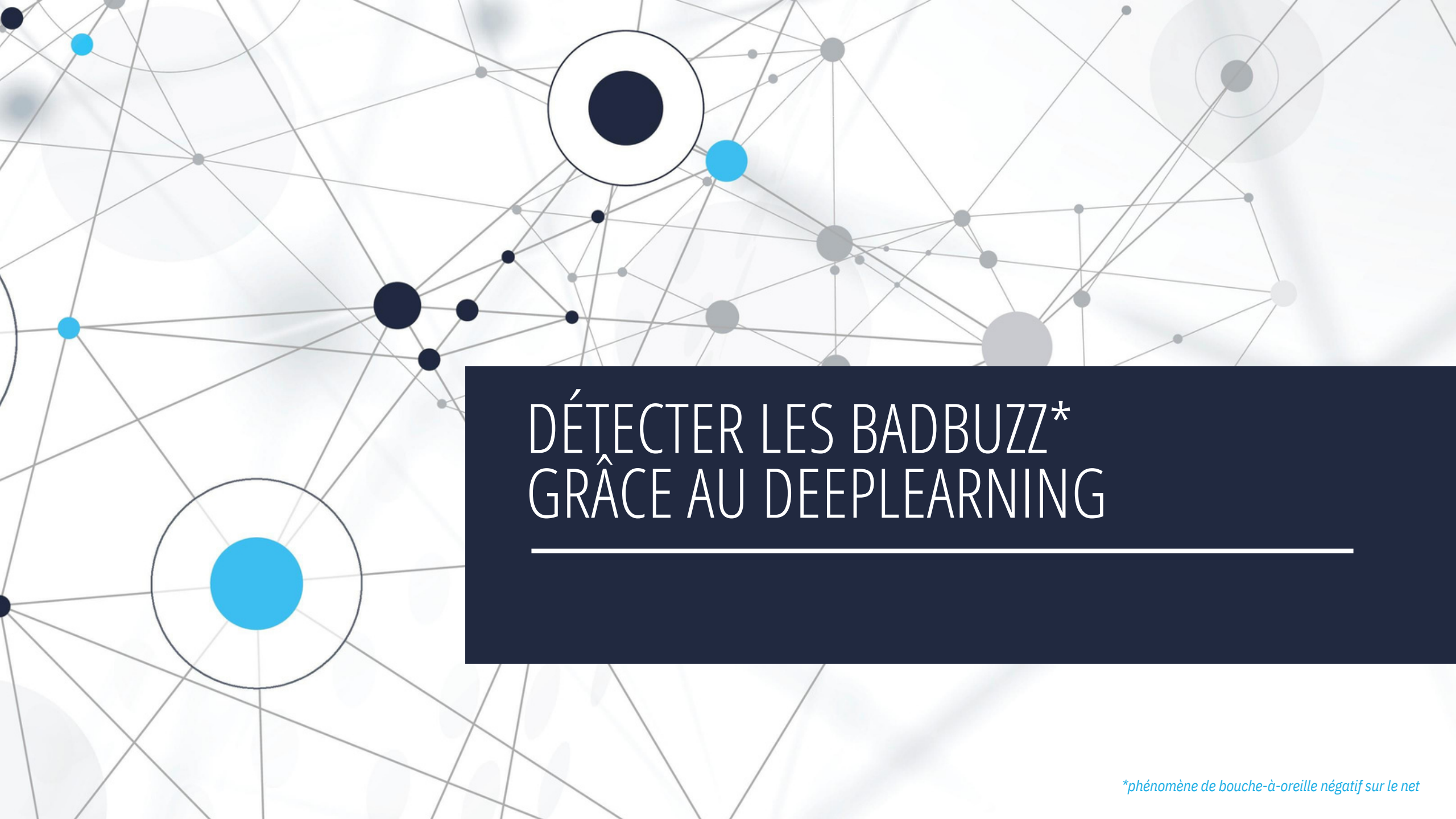


DÉTECTER LES BADBUZZ* GRÂCE AU DEEPLEARNING

**phénomène de bouche-à-oreille négatif sur le net*

AGENDA DU JOUR

PROJET ET DÉMARCHE

PRÉ-TRAITEMENT DES DONNÉES

PRÉSENTATION DES APPROCHES

CHOIX MODÈLE ET DÉPLOIEMENT

PROJET ET DÉMARCHE

CONTEXTE PROJET ET DÉMARCHE

Problématique

Surveiller la réputation sur les réseaux sociaux

Prédire le sentiment associé à un tweet

Livrer un prototype fonctionnel du modèle

Source de travail

Utiliser des données open source d'analyse de sentiment

Trouver des tweets annotés du sentiment exprimé (+ive ou -ive)

Jeu de données : Sentiment140

Démarche

Analyser et prétraiter les données

Explorer les différentes approches

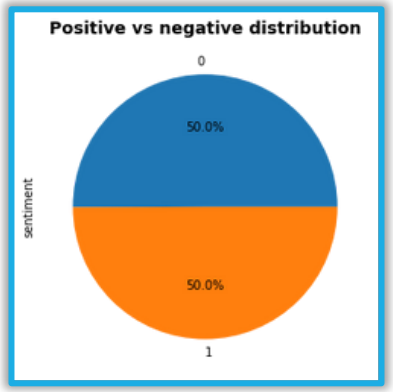
Déployer le meilleur modèle

PRÉ-TRAITEMENT DES DONNÉES

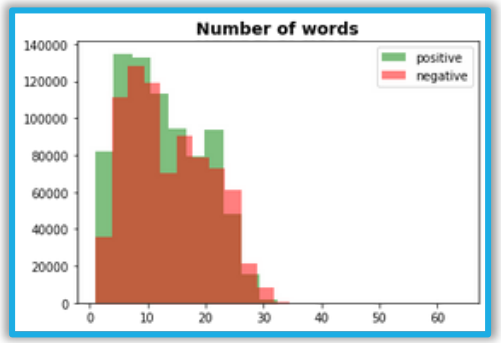
EXPLORATION DES DONNÉES

1,6 millions de Tweets collectés sur 3 mois: avril-juin 2009

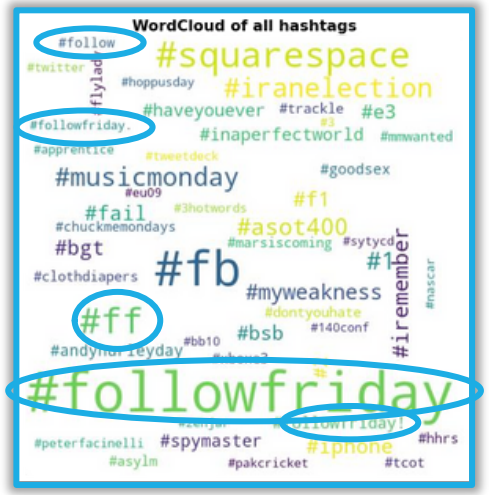
Classes cible équilibrées



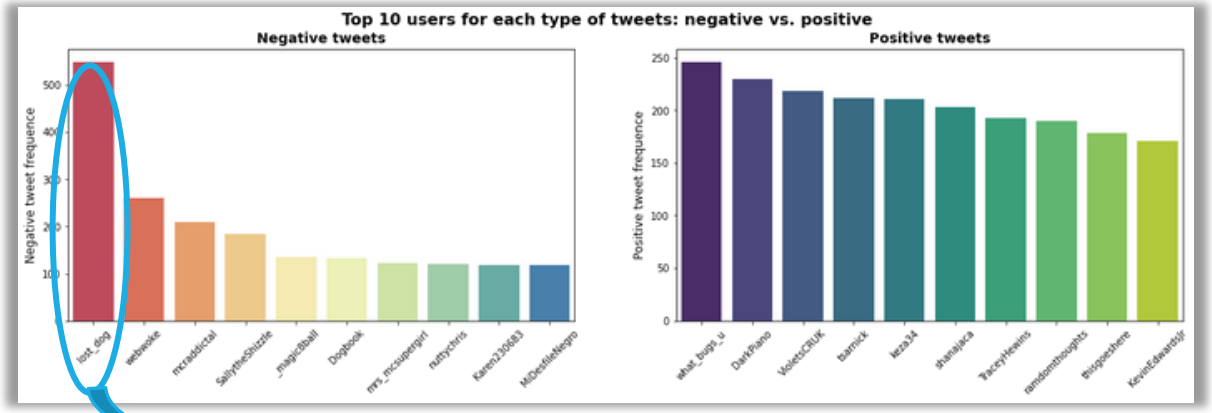
13/14 mots en moyenne



Hashtags peu informatifs (top 50)



Identification d'utilisateur étrange

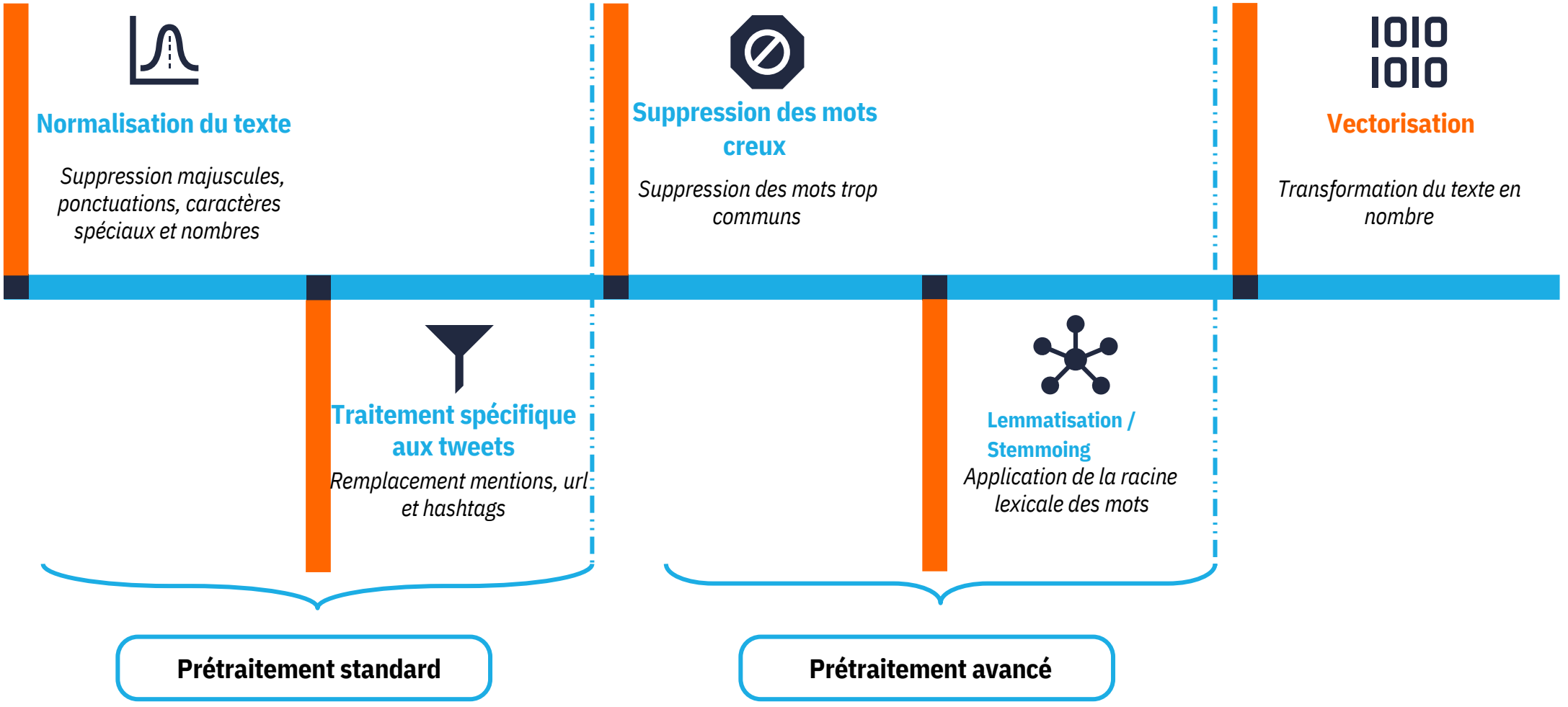


N°1 NEGATIVE MOOD
(549, 3)

	sentiment	user	tweet
43935	0	lost_dog	@NyleW I am lost. Please help me find a good home.
45574	0	lost_dog	@SallyD I am lost. Please help me find a good home.
46919	0	lost_dog	@zuppaholic I am lost. Please help me find a good home.
47949	0	lost_dog	@LOSTPETUSA I am lost. Please help me find a good home.
50572	0	lost_dog	@JeanLevertHood I am lost. Please help me find a good home.

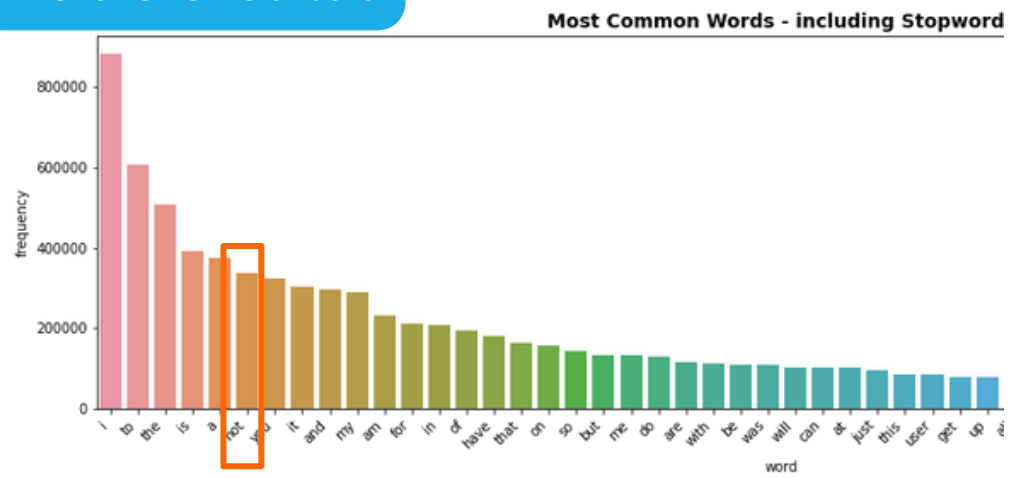
APPLICATION DE 2 TYPES DE PRÉTRAITEMENT

Nettoyage préliminaire : Suppression des doublons sur le sous-ensemble [Utilisateur –Tweet] et des lignes vides



COMPARAISON DES PRÉTRAITEMENTS

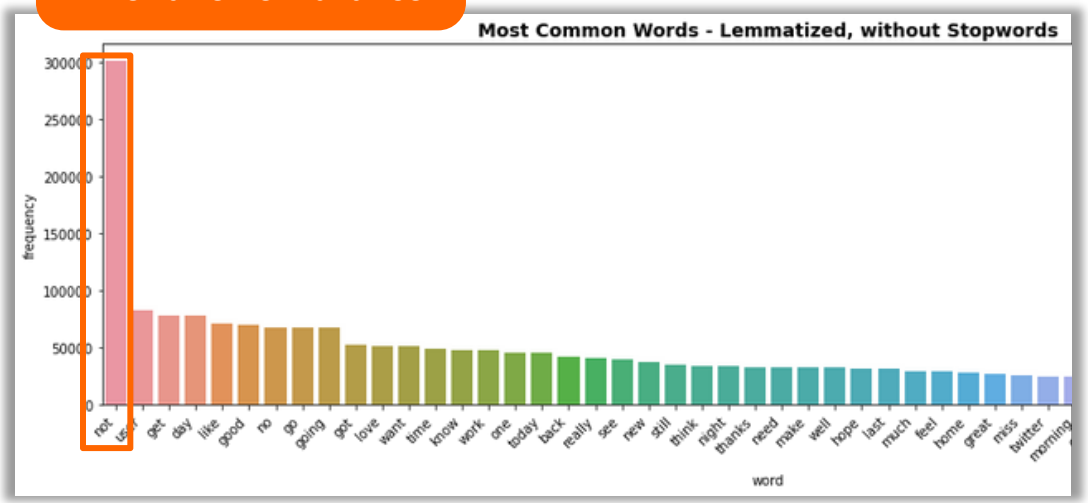
Prétraitement standard



‘user nope they did not have it’

‘user no it is not behaving at all i am mad why am i here because i can not see you all over there’

Prétraitement avancé



user nope not

‘user no not behaving mad not see’

Utilisation des données avec Prétraitement standard : garder la séquence des mots, qui est importante dans l’analyse de sentiment

PRÉSENTATION DES APPROCHES

3 APPROCHES DIFFÉRENTES

Modèle «Prêt-à-l'emploi »

1.600 tweets

**Vectorisation si applicable*

**Entraînement si applicable*

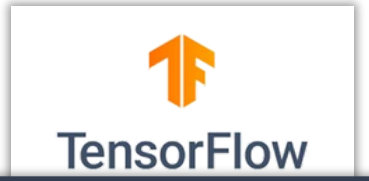
Bibliothèque de modules pour construire des modèles

1.600 tweets

Hachage des caractéristiques

Extraction des caractéristiques n-grams

Régression logistique



Modèles «propriétaire» de réseaux de neurones profonds

1.600 & 400.000 tweets

Word Embeddings (plongement de mots)

Réseaux de neurones récurrents

Prédiction

Evaluation des 3 approches + Déploiement du meilleur modèle

MODÈLE CLÉ-EN-MAIN : API SENTIMENT ANALYSIS DE MICROSOFT AZURE

Modèle «Prêt-à-l'emploi »

Prérequis:

Ouverture d'un compte Azure (sur le Portail)

Souscription au service **Text Analytics** de la collection Azure Cognitive Services

Récupération du point de terminaison et de la clé de souscription pour utiliser le service

Utilisation du Kit de Développement Logiciel (SDK) Azure Machine Learning pour Python pour appeler le service

Pour valider notre hypothèse sur le prétraitement des données, nous avons testé l'API (Interface de programmation d'application avec les données :

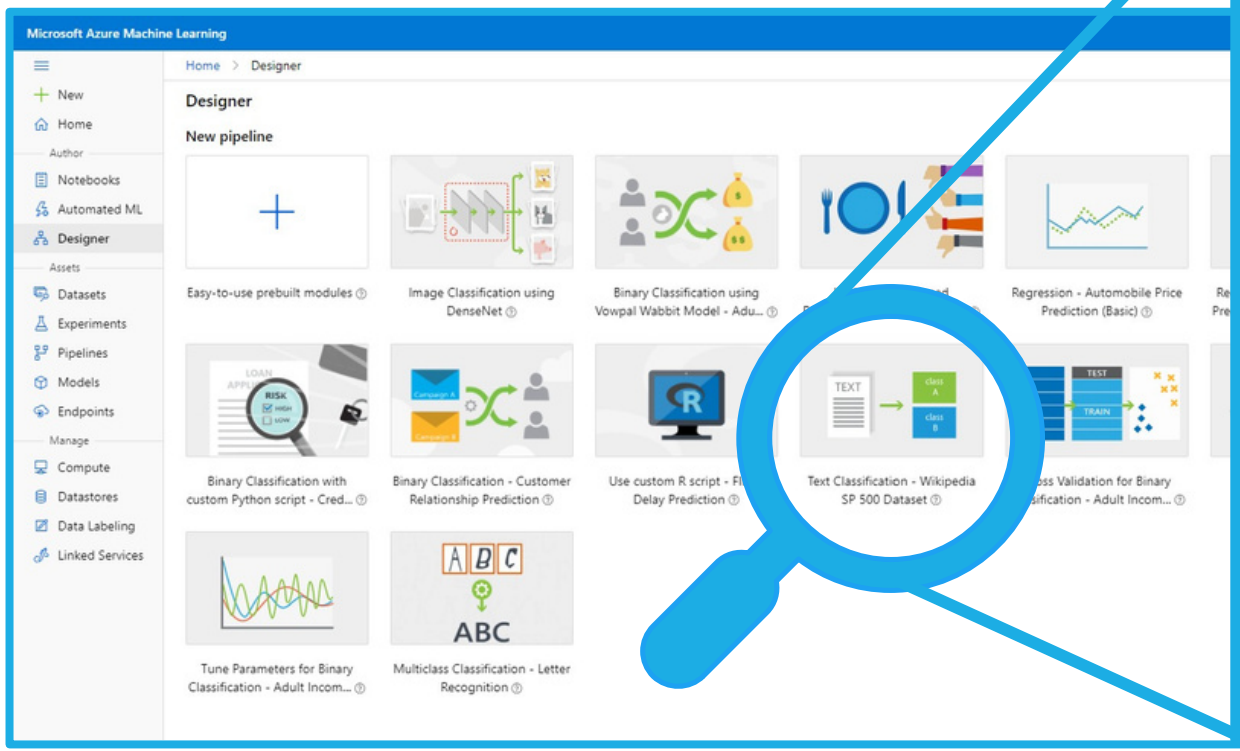
- **Brutes**, sans aucun traitement;
- **Prétraitées**, standard;
- **Prétraitées**, avancé.

	Model	Predict_time	AUC_Score	Accuracy
0	Original tweets	186.1	74.349%	74.335%
1	Cleaned tweets	192.2	75.989%	75.949%
2	Lemmatized tweets	184.0	75.380%	75.319%

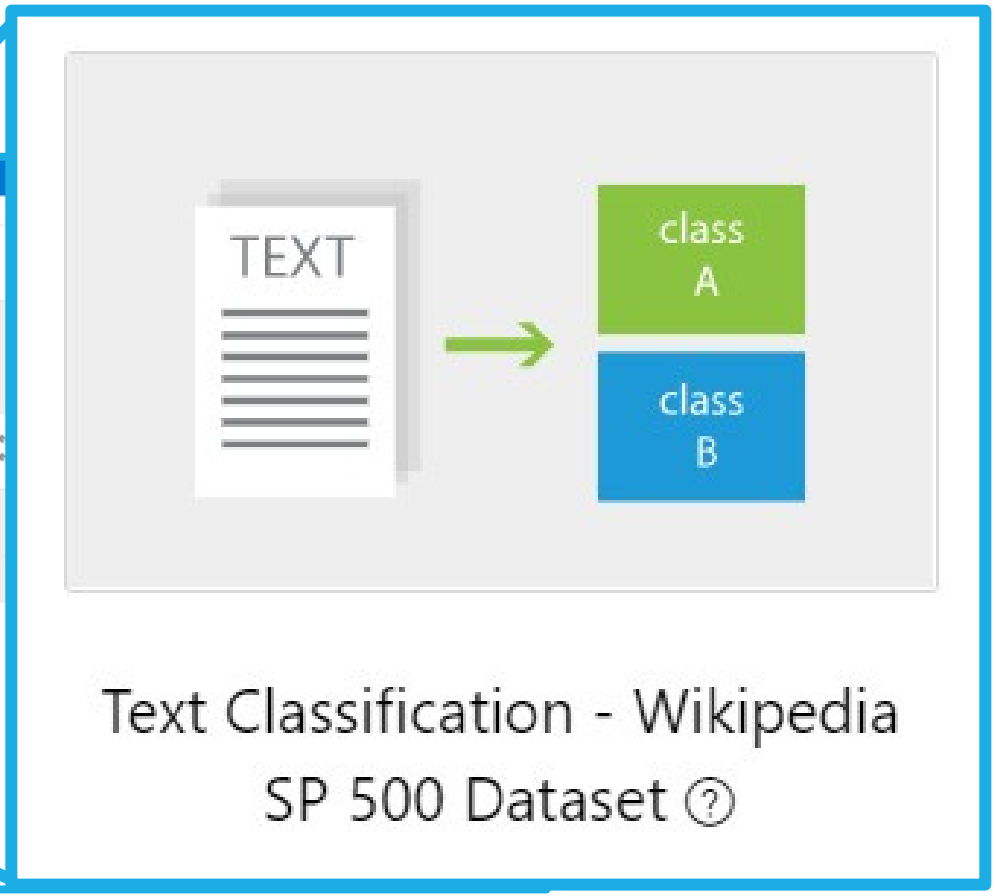
MODÈLE BOÎTE-À-OUTILS : AZURE MACHINE LEARNING STUDIO (AMLS)

AZURE Machine Learning

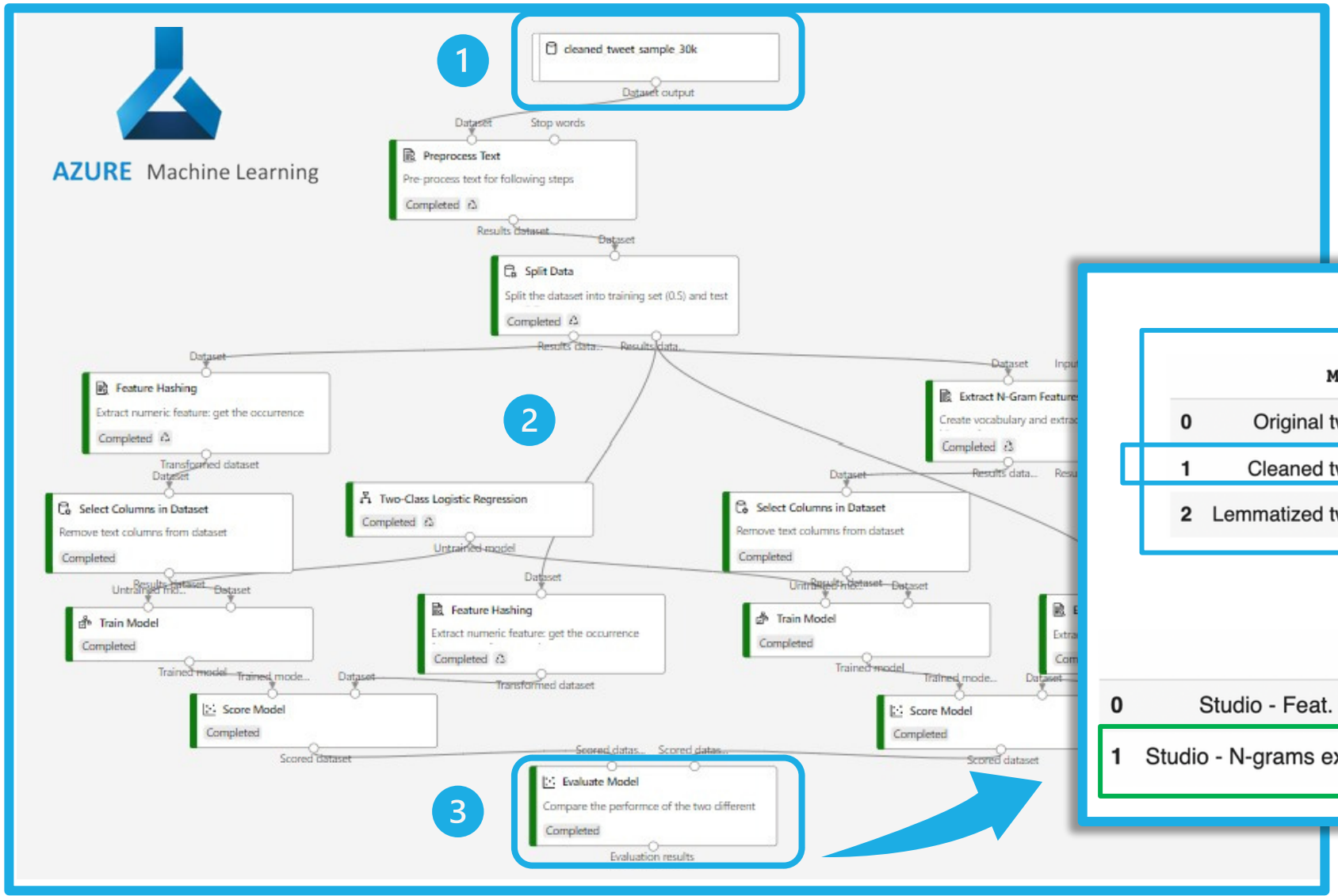
Prérequis:
Ouverture d'un compte Azure (sur le Portail)
Souscription d'un espace de travail **Azure Machine Learning**



Clonage d'un pipeline existant



AMLS: CONCEPTEUR ET PIPELINE DE MODULES



1. Remplacement du jeu de données
2. Correction des paramètres des modules
3. Exécution et évaluation des modèles

Approche 1

	Model	Predict_time	AUC_Score	Accuracy
0	Original tweets	186.1	74.349%	74.335%
1	Cleaned tweets	192.2	75.989%	75.949%
2	Lemmatized tweets	184.0	75.380%	75.319%

Approche 2

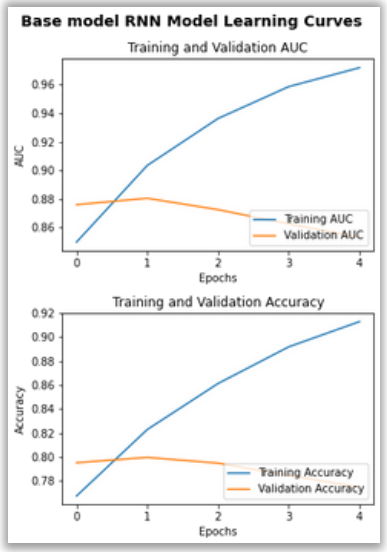
	Model	Predict_time	AUC_Score	Accuracy
0	Studio - Feat. hashing	81.5	70.200%	63.400%
1	Studio - N-grams extraction	103.4	78.400%	70.600%

MODÈLE AVANCÉ: RÉSEAUX DE NEURONES PROFONDS

TensorFlow

Optimisation du modèle de base (*baseline*)

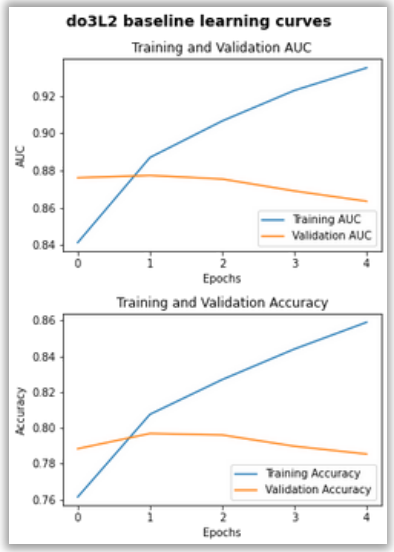
```python
# Add sequential model
base_model = Sequential()
base_model.add(Embedding(input_dim=VOCAB_SIZE,
                         output_dim=EMBEDDING_DIM,
                         input_length=MAX_LEN))
base_model.add(SimpleRNN(128, dropout=0.3))
base_model.add(Dense(64, activation='relu'))
base_model.add(Dense(1, activation='sigmoid'))
base_model.summary()
```



	Model	AUC_Score	Accuracy	Loss
0	SimpleRNN	85.245085	77.525002	56.828576

```python
# Add sequential model
do3l2_model = Sequential()
do3l2_model.add(Embedding(input_
                          output_
                          input_length=MAX_LEN))
do3l2_model.add(SpatialDropout1D(0.3))
do3l2_model.add(SimpleRNN(128, dropout=0.3))
do3l2_model.add(Dropout(0.3))
do3l2_model.add(Dense(64, activation='relu',
                      kernel_regularizer=regularizers.l2(0.001)))
do3l2_model.add(Dropout(0.3))
do3l2_model.add(Dense(1, activation='sigmoid'))
do3l2_model.summary()
```

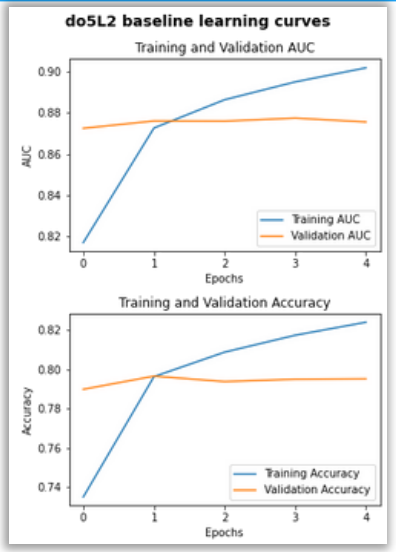
Ajout de Dropout (0,3) et Regularizers(L2)



	Model	AUC_Score	Accuracy	Loss
0	do3L2 SimpleRNN	86.35236	78.49375	49.211758

```python
# Add sequential model
do5l2_model = Sequential()
do5l2_model.add(Embedding(input
                          output
                          input_length=MAX_LEN))
do5l2_model.add(SpatialDropout1D(0.5))
do5l2_model.add(SimpleRNN(128, dropout=0.5))
do5l2_model.add(Dropout(0.5))
do5l2_model.add(Dense(64, activation='relu',
                      kernel_regularizer=regularizers.l2(0.001)))
do5l2_model.add(Dropout(0.5))
do5l2_model.add(Dense(1, activation='sigmoid'))
do5l2_model.summary()
```

Optimisation du Dropout à 0,5



	Model	AUC_Score	Accuracy	Loss
0	do5L2 SimpleRNN	87.459093	79.468751	44.949293

MODÈLE AVANCÉ: DEEPLEARNING & PLONGEMENT DE MOTS

Modèle avancé : RNN & Embeddings

TensorFlow

Paramètres communs

Configure all modeling variables

VOCAB_SIZE:	50000
MAX_LEN:	36
EMBEDDING_DIM:	256
DROP_OUT:	0.5
OPTIM_LR:	0.001
REGUL_LR:	0.001
NUM_EPOCHS:	5
BATCH_SIZE:	250

Modèle de base qu'on veut encore améliorer

Model: "base_model"

Layer (type)	Output Shape	Param #
embedding (Embedding)	(None, 36, 256)	12800000
spatial_dropout1d (SpatialDr	(None, 36, 256)	0
simple_rnn (SimpleRNN)	(None, 128)	49280
dropout (Dropout)	(None, 128)	0
dense (Dense)	(None, 64)	8256
dropout_1 (Dropout)	(None, 64)	0
dense_1 (Dense)	(None, 1)	65

Total params: 12,857,601
Trainable params: 12,857,601
Non-trainable params: 0

Meilleure mémoire du passé proche

lstm (LSTM)	(None, 128)	197120

Lecture gauche/droite et droite/gauche

bidirectional (Bidirectional	(None, 256)	394240

Utilisation du modèle LSTM avec les Embeddings pré-entraînés

Word2Vec

embedding_3 (Embedding)	(None, 36, 300)	28044000

GloVe

embedding_4 (Embedding)	(None, 36, 200)	18696000

USE

keras_layer (KerasLayer)	(None, 512)	256797824

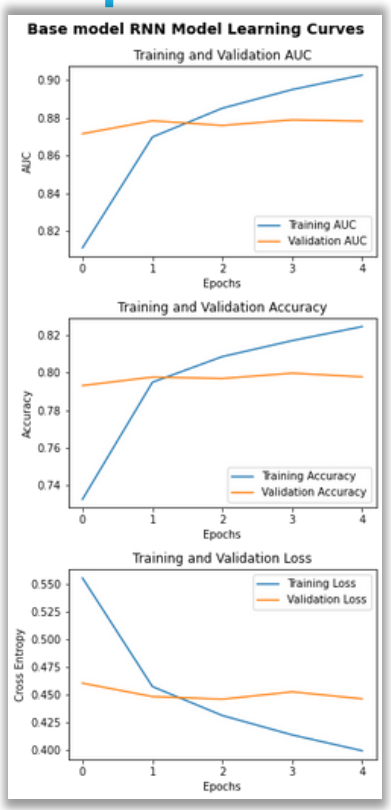
Transforme une phrase entière en vecteurs, n'utilise pas de LSTM

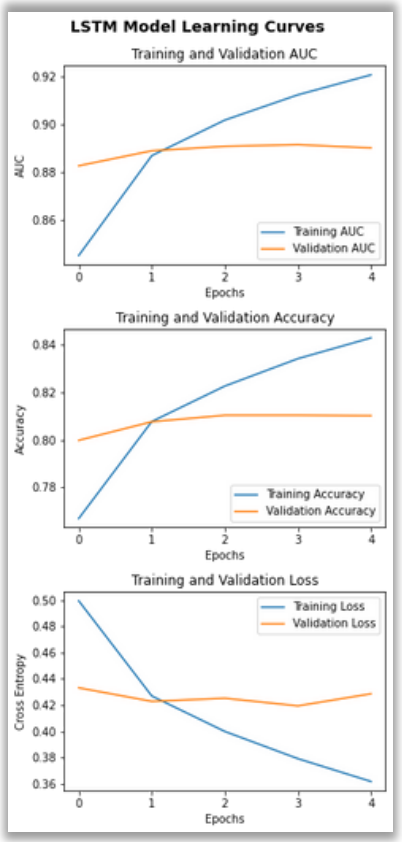
Différents réseaux de neurones : RNN → LSTM → LSTM bidirectionnel

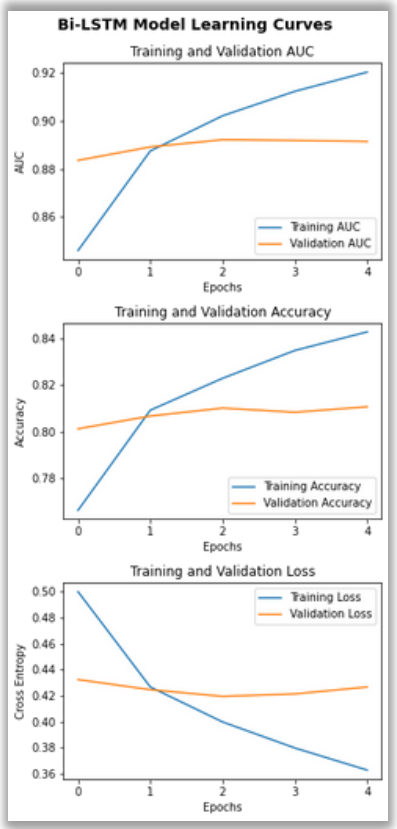
Différents embeddings: fromscratch → pré-entrainés (Word2Vec, GloVe, USE...)

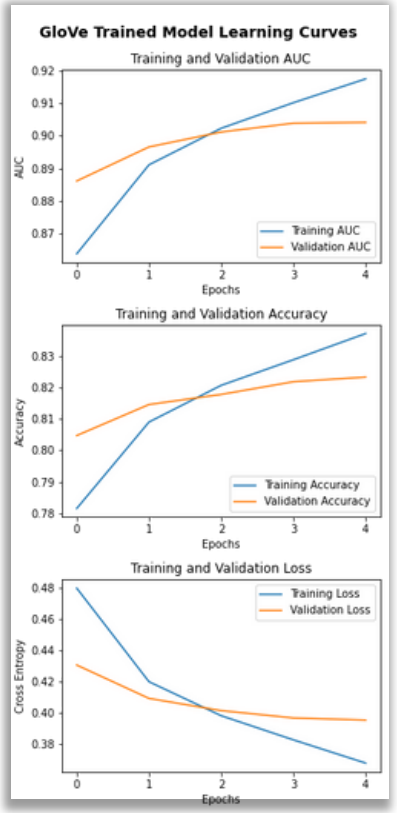
PERFORMANCE DES MODÈLES AVANCÉS

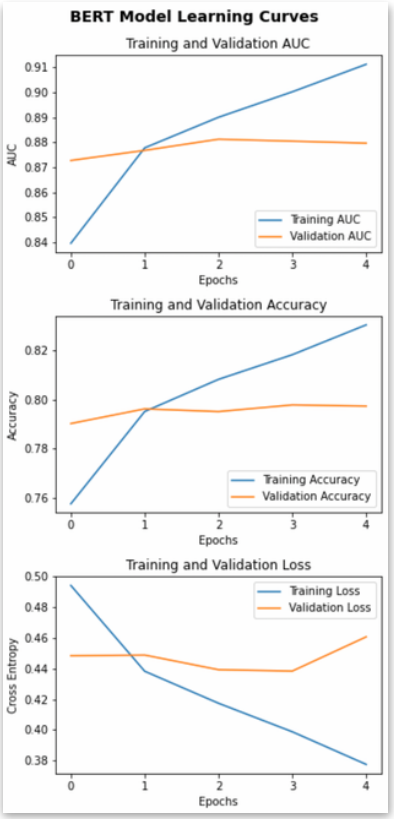
TensorFlow



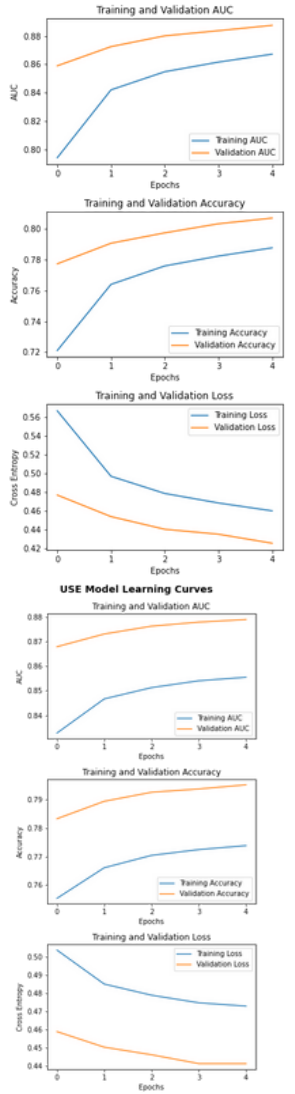


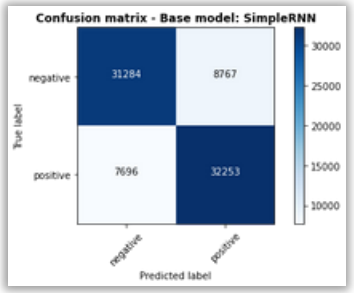


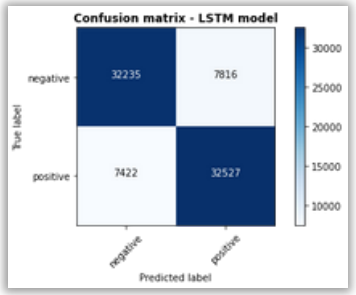


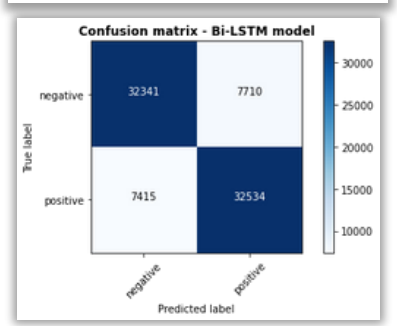


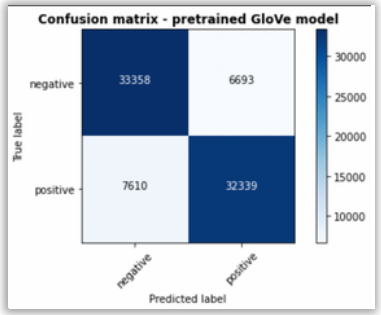
Word2Vec et USE en sous-apprentissage (underfit)

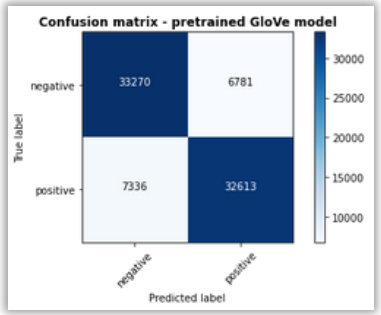












CHOIX MODÈLE ET DÉPLOIEMENT

SYNTHÈSE DES PERFORMANCES MODÈLES AVANCÉS 1600 tweets

	Model	False positives	AUC_Score	Accuracy	Loss	Fit_time	Eval_time	Predict_time
6	Glove (full DF)	24499	91.580%	83.556%	37.130%	174.9	10.3	19.9
4	GloVe +LSTM	54	79.405%	72.500%	62.565%	4.7	0.1	0.5
3	Word2Vec +LSTM	111	73.094%	57.812%	71.840%	5.5	0.1	0.9
1	LSTM	87	68.042%	59.375%	69.497%	7.1	0.1	0.5
5	BERT	38	66.425%	61.719%	65.152%	38.5	0.6	3.9
2	Bi-LSTM	122	65.731%	57.187%	79.331%	7.8	0.1	0.9
0	Simple RNN	99	57.612%	53.438%	75.649%	7.5	0.1	0.2

Prétraitement standard

	Model	False positives	AUC_Score	Accuracy	Loss	Fit_time	Eval_time	Predict_time
6	Glove (full DF)	29843	89.812%	81.645%	40.549%	145.1	4.7	15.8
4	GloVe +LSTM	46	77.884%	72.812%	65.494%	3.5	0.1	0.4
2	Bi-LSTM	17	76.235%	66.250%	69.596%	5.6	0.1	0.7
3	Word2Vec +LSTM	105	73.460%	61.875%	71.015%	4.7	0.1	0.7
1	LSTM	108	71.757%	59.062%	70.933%	3.9	0.1	0.4
5	BERT	107	64.335%	57.031%	68.499%	31.8	0.6	2.9
0	Simple RNN	161	57.082%	48.438%	77.619%	4.1	0.1	0.2

Prétraitement avancé - Lemmitization

	Model	False positives	AUC_Score	Accuracy	Loss	Fit_time	Eval_time	Predict_time
6	Glove (full DF)	29303	89.037%	80.795%	42.303%	150.0	4.7	15.6
3	Word2Vec +LSTM	100	75.733%	63.125%	68.884%	3.8	0.1	0.7
1	LSTM	83	75.111%	64.062%	69.772%	5.7	0.1	0.4
4	GloVe +LSTM	53	74.997%	68.437%	68.806%	3.4	0.1	0.3
2	Bi-LSTM	111	71.718%	58.438%	71.898%	5.8	0.1	0.7
5	BERT	0	62.302%	51.172%	78.047%	34.0	0.6	2.9
0	Simple RNN	119	57.229%	52.188%	76.632%	6.8	0.1	0.2

Prétraitement avancé - Stemming

SYNTHÈSE DES PERFORMANCES DES APPROCHES (Final)

Vectorisation	Modèle	Machine	Dimension de vecteurs	Echantillon	Paramètres entraînables	Temps d'exécution	AUC Score	Accuracy
N.A.	API Sentiment	CPU	N.A.	1.600	N.A.	3,2 min	75,989 %	75,949 %
Hachage de caractéristiques	Régression logistique	vCPU	N.A.	1.600	N.A.	1,35 min	70,200 %	63,400 %
Extraction de caract. n-grams	Régression logistique	vCPU	N.A.	1.600	N.A.	1,72 min	78,400 %	70,600 %
Embeddings propriétaire (Keras)	RNN Simple	GPU	256	400.000	12.857.601	8,10 min	87,653 %	79,421 %
Embeddings propriétaire (Keras)	LSTM	GPU	256	400.000	13.005.441	7,41 min	88,955 %	80, 953%
Embeddings propriétaire (Keras)	LSTM bidirectionnel	GPU	256	400.000	13.210.753	7,23 min	89,091 %	81,094 %
Embeddings pré-entraînés: Word2Vec	LSTM	GPU	300	400.000	227.969	1,20 min	88,685 %	80,586 %
Embeddings pré-entraînés: GloVe	LSTM	GPU	200	400.000	176.769	1,73 min	90,279 %	82,121 %
Embeddings pré-entraînés: USE	N.A.	GPU	512	400.000	131,585	2,53 min	87,912 %	79,594%
Bert - Transformers	Distill BERT	GPU	768	400.000	108.890.881	159 min	87,742 %	79,484 %

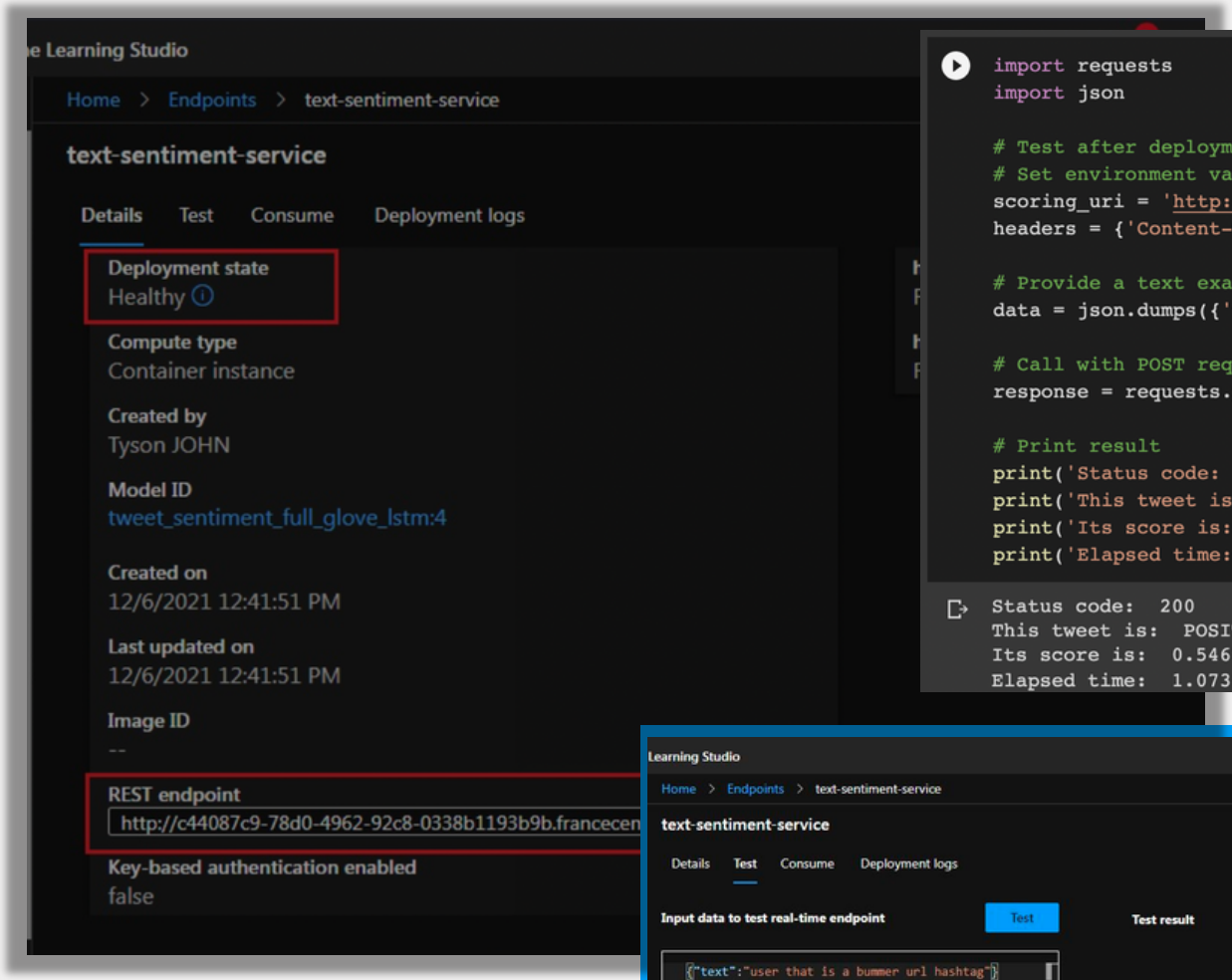
TWEETS LABELISÉS «NÉGATIF» PRÉDITS «POSITIF»

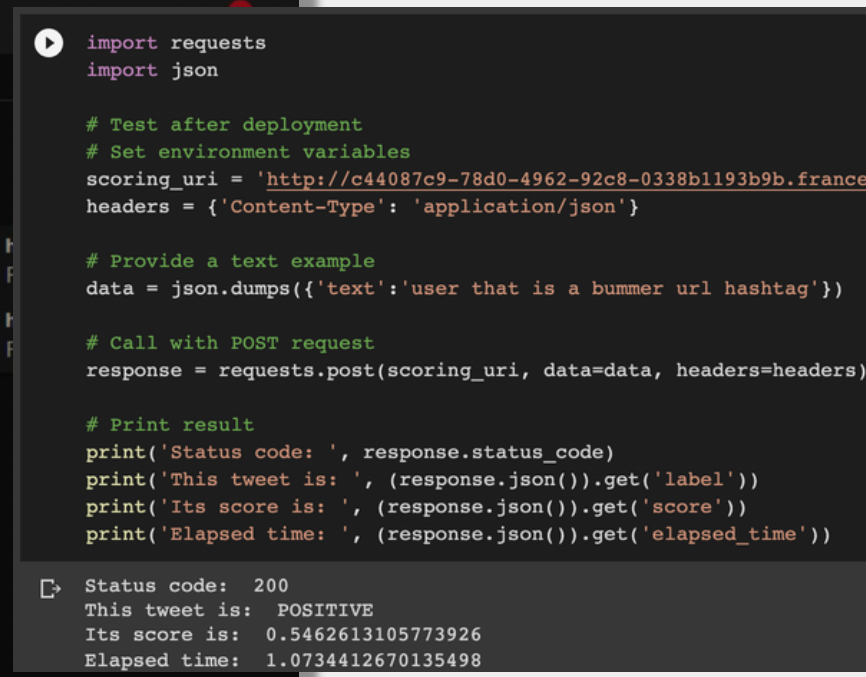
		INTERPRETATION
Housemate has told me to shut up with the hysterical laughing.	housemate has told me to shut up with the hysterical laughing	Erreur réelle du modèle
Holiday Was FANTASTIC	holiday was fantastic	Erreur de label à l'origine? Ce tweet semble très positif
Sugarsnap peas, carrots and califlower. what a lunch!	sugarsnap peas carrots and califlower what a lunch	Détection de sarcasme compliquée
kind of very pissed off that my parentals are going to NYC in Septmember without me. but now I just figured out what they can buy me!	kind of very pissed off that my parentals are going to nyc in septmember without me but now i just figured out what they can buy me	Mix d'émotions: joie, peine...
Rona Howarda? S tim to jde z kopce. Kdyz vezmu Beautiful Mind ... Sifra... Andele. Od deviti k peti	user rona howarda s tim to jde z kopce kdyz vezmu beautiful mind sifra andele od deviti k peti	Tweets qui sont dans une langue autre que l'Anglais (ici, le tchèque)

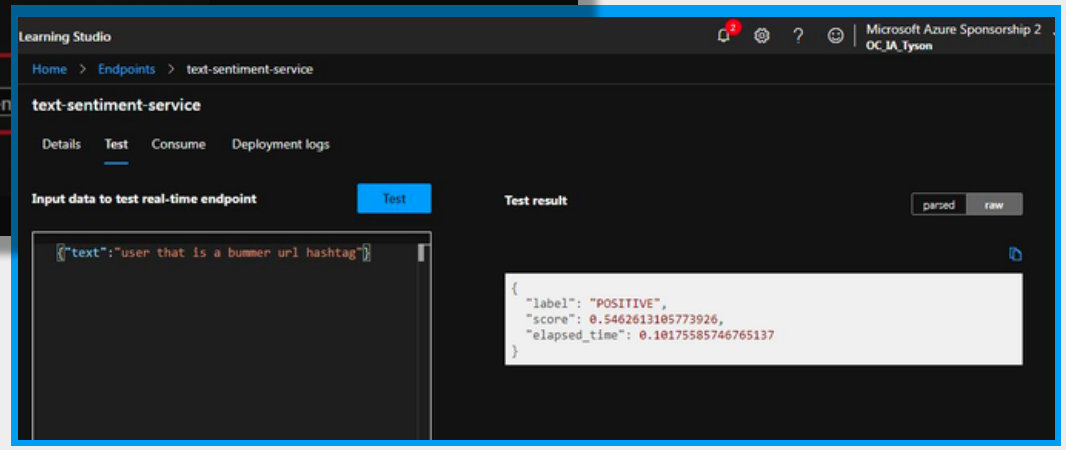
DÉPLOIEMENT SUR AZURE MACHINE LEARNING

Flux de travail

- Installation de Azure SDK pour Python
- Inscription du modèle sur Azure
- Préparation d'un script pour initier le déploiement et prédire avec le modèle
- Définition des configurations: environnement, dépendances, inférence
- Exécution du déploiement
- Utilisation du service web déployé







SYNTHÈSE

PROPOSITION DE CRITÈRES DE SÉLECTION

	Approche 1 «Clé en main»	**Approche 2 «Boîte à outils»**	**Approche 3 «propriétaire»**
Connaissance en IA	Peu de connaissance	Connaissances de base	Expertise
Prise en main	Simple	Relativement simple	Complexe
Adaptabilité	Aucune	Possible	Complète
Local / Cloud	Non / Oui	Non / Oui	Oui / Oui
Performance	Moyen	Bon	Très bon
Investissement matériel	Inclus	Inclus	(à définir)*
Investissement temps	Faible	Modéré	Conséquent
Coûts Cloud	Usage en volumétrie	Usage horaire	(à définir)*

**Le coût d'un GPU est moindre par rapport à un vrai coût matériel*

CONCLUSION

Un **choix d'approche** dépendant principalement des besoins exprimés par les équipes en interne ET des ressources disponibles (humain, financier, temps).

Une **performance du modèle** dépendant principalement :
-La qualité initiale des données ;
-Du prétraitement effectué.

QUESTIONS / RÉPONSES

RÉFÉRENCES

- Analyse de texte (text mining) avec Azure Cognitive Services -TextAnalytics
- Solution Cloud de Machine Learning avec Azure Machine Learning et son concepteur (Drag and Drop)
- Deep Learning : Tensorflow -Réseaux de neurones récurrents, LSTM -TowardsData Science, wordembeddings
- Pre-trained word embeddings: word2vec, GloVe(Global Vectors for Word Representation, USE (Universal Sentence Encoder) -v4
- Transformers : The Illustrated Transformer
- Déploiement : How and where to deploy Machine Learning models to Azure
- Autre: neptune.ai -structure and manage nlp projects

Ce document a été produit dans le cadre de la soutenance du projet n°7 du parcours Ingénieur IA d'OpenClassrooms:
«Détectez les Bad Buzz grâce au DeepLearning »

Mentor : Mohamed Laaraiedh
Evaluateur : Aminata Diaby

OPENCLASSROOMS
En partenariat avec
CentraleSupélec
Microsoft