

MARKETING INTELLIGENCE CONSULTING



3

APPROCHES POUR UN MODELE DE PREDICTION DE SENTIMENT

Moussa Kibaly - INGENIEUR IA

Nous allons aborder dans cet article 3 approches différentes pour la conception d'un modèle de prédiction du sentiment associé à un tweet. Elles ont été développées dans le studio Azure Machine Learning de Microsoft et se présentent sous les formes suivantes :

- API sur étagère
- Modèle sur mesure simple
- Modèle sur mesure avancé

1. API sur étagère

Ce modèle développé dans un notebook dans Azure ML Studio récupère 1000 tweets du fichier initiale de 1 600 000 tweets. L'échantillon contient autant de tweets positifs ("sentiment" = 4) que négatifs ("sentiment" = 0).

	sentiment	id	date	query	user	tweet
0	0	2214557997	Wed Jun 17 16:54:53 PDT 2009	NO_QUERY	katirech21	just ateeee. so tired
1	0	2203600066	Wed Jun 17 00:06:07 PDT 2009	NO_QUERY	SomersetBob	@Claire_Cordon Oh! That's odd. Mine's always b...
2	0	2245264644	Fri Jun 19 15:53:21 PDT 2009	NO_QUERY	amhicks01	Lost my phone...This means I about to get some...
3	0	1992065737	Mon Jun 01 08:08:28 PDT 2009	NO_QUERY	zenjar	Why do I always feel tired, I sleep more than ...
4	0	2258718810	Sat Jun 20 16:23:03 PDT 2009	NO_QUERY	rosoco	thinkin bout goin to the improv tonite to go s...

APPEL AU SERVICE COGNITIF AZURE

Le modèle envoie un tweet via une requête de type POST au service d'analyse de sentiment de Microsoft Azure, en fournissant des clés d'authentification et le point de terminaison qui sont définis lors de la création d'une ressource liée au service cognitif d'Azure et que l'on a nommé Tweet :

- la clé de service (champ "CLE 1" ou "CLE 2" dans Azure)
- la région des utilisateurs appelant le service Tweet (champ "Emplacement" dans Azure)
- le point de terminaison (champ du même nom)

Tweet | Clés et point de terminaison

Cognitive Services

Rechercher (Cmd+ /)

«

Régénérer Key1

Régénérer Key2

Vue d'ensemble

Journal d'activité

Contrôle d'accès (IAM)

Étiquettes

Diagnostiquer et résoudre les problèmes

Gestion des ressources

Démarrage rapide

Clés et point de terminaison

Niveau tarifaire

Réseau

Identité

Facturation par abonnement

Propriétés

Ces clés sont utilisées pour accéder à votre API de service cognitif. Ne partagez pas vos clés. Stockez-les en lieu sûr, par exemple avec Azure Key Vault. Nous vous recommandons également de régénérer ces clés régulièrement. Une seule clé est nécessaire pour effectuer un appel d'API. Pendant la régénération de la première clé, vous pouvez utiliser la deuxième pour continuer à avoir accès au service.

Afficher les clés

CLÉ 1

.....

CLÉ 2

.....

Emplacement ⓘ

centralus

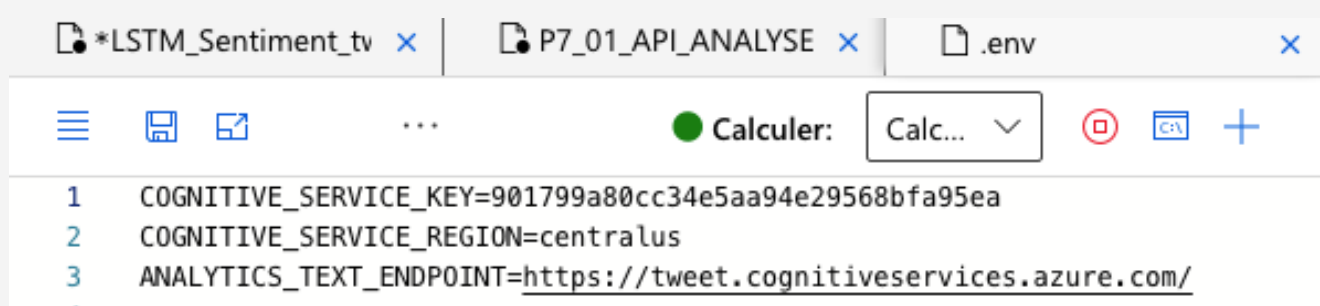
Point de terminaison

https://tweet.cognitiveservices.azure.com/

NOTEBOOK DANS LE STUDIO AZURE ML

Ces clés étant confidentielles, elles doivent être stockées dans des variables d'environnement pour des raisons de sécurité.

Dans le studio Azure ML, nous avons stocké ces variables dans un fichier .env :



```
*LSTM_Sentiment_tv x | P7_01_API_ANALYSE x | .env x
1 COGNITIVE_SERVICE_KEY=901799a80cc34e5aa94e29568bfa95ea
2 COGNITIVE_SERVICE_REGION=centralus
3 ANALYTICS_TEXT_ENDPOINT=https://tweet.cognitiveservices.azure.com/
4
```

Le notebook récupère ces variables d'environnement contenant les clés d'appel au service Tweet en utilisant la fonction **load_dotenv** de la librairie python **dotenv**.

```
1 import os
2 import requests
3 import uuid
4 import json
5 from dotenv import load_dotenv
6 load_dotenv()
7
8 key1_var_name = 'COGNITIVE_SERVICE_KEY'
9 key2_var_name = 'COGNITIVE_SERVICE_REGION'
10 endpoint_var_name = 'ANALYTICS_TEXT_ENDPOINT'
11
12
13 if not key1_var_name in os.environ:
14     raise Exception('Please set/export the environment variable 1: {}'.format(key1_var_name))
15 subscription_key = os.environ[key1_var_name]
16
17 if not key2_var_name in os.environ:
18     raise Exception('Please set/export the environment variable 2: {}'.format(key2_var_name))
19 region_key = os.environ[key2_var_name]
20
21
22 if not endpoint_var_name in os.environ:
23     raise Exception('Please set/export the environment variable 3: {}'.format(endpoint_var_name))
24 endpoint = os.environ[endpoint_var_name]
25
26 path = 'text/analytics/v3.0/sentiment'
27 constructed_url = endpoint + path
28
```

REQUETE POST A L' API ANALYSE DE SENTIMENT

L' url complet d'appel à l'API est constitué de la concaténation du point de terminaison (dans notre cas "https://tweet.cognitiveservices.azure.com/") et de la variable path = "text/analytics/v3.0/sentiment" qui spécifie qu'on fait appel au service d'analyse du sentiment.

```
28
29 headers = {
30     'Ocp-Apim-Subscription-Key': subscription_key,
31     'Ocp-Apim-Subscription-Region': region_key,
32     'Content-type': 'application/json',
33     'X-ClientTraceId': str(uuid.uuid4())
34 }
35 taux_success = []
36
37
38 def detectLang(*param):
39     positive_succ = 0
40     negative_succ = 0
41     i = 0
42     for elt in param:
43         body = {
44             "documents": [
45                 {
46                     "language": "en",
47                     "id": i,
48                     "text": elt
49                 }
50             ]
51         }
52         request = requests.post(constructed_url, headers=headers, json=body)
53         response = request.json()
54         id = response['documents'][0]['id']
55         sentiment = response['documents'][0]['sentiment']
56         sc = response['documents'][0]['confidenceScores']
```

Le tweet en anglais est envoyé dans la variable "body" sous format json.

REPONSE DE L' API

La réponse revient sous format json avec le sentiment pouvant avoir les valeurs "positiv", "neutral", "negativ" avec pour chacune d'elle, un score de confiance compris entre 0 et 1.

```

xte : '@Swing_Rat MS doesn't ship software w/viruses, but I get your point.' a le sentiment : 'negative'
un score de : '{ 'positive': 0.04, 'neutral': 0.39, 'negative': 0.57 }'
xte : '@LittleLiverbird I let you win , go to bed !!' a le sentiment : 'positive' avec un
score de : '{ 'positive': 0.8, 'neutral': 0.19, 'negative': 0.01 }'
xte : '@damnedredhead You should be Chocolate in mail to you.' a le sentiment : 'neutral'
score de : '{ 'positive': 0.01, 'neutral': 0.98, 'negative': 0.01 }'
xte : 'I want the X-Men Origins of storm' a le sentiment : 'neutral' avec un score de :
'sitive': 0.03, 'neutral': 0.96, 'negative': 0.01 }'
xte : '@april_q8 well shwaya mo fahmetah lol abeech it-thab6eny' a le sentiment : 'positive'
un score de : '{ 'positive': 0.97, 'neutral': 0.02, 'negative': 0.01 }'
xte : 'On a boat trip with my bestfriend.' a le sentiment : 'positive' avec un score de
'sitive': 0.74, 'neutral': 0.24, 'negative': 0.02 }'

```

2. Modèle sur mesure simple

Ce modèle développé dans le studio Azure ML utilise l'interface graphique drag & drop (rubrique Concepteur dans le menu). L'outil est utile pour le développement rapide d'un modèle en minimisant le codage en python ou en R.

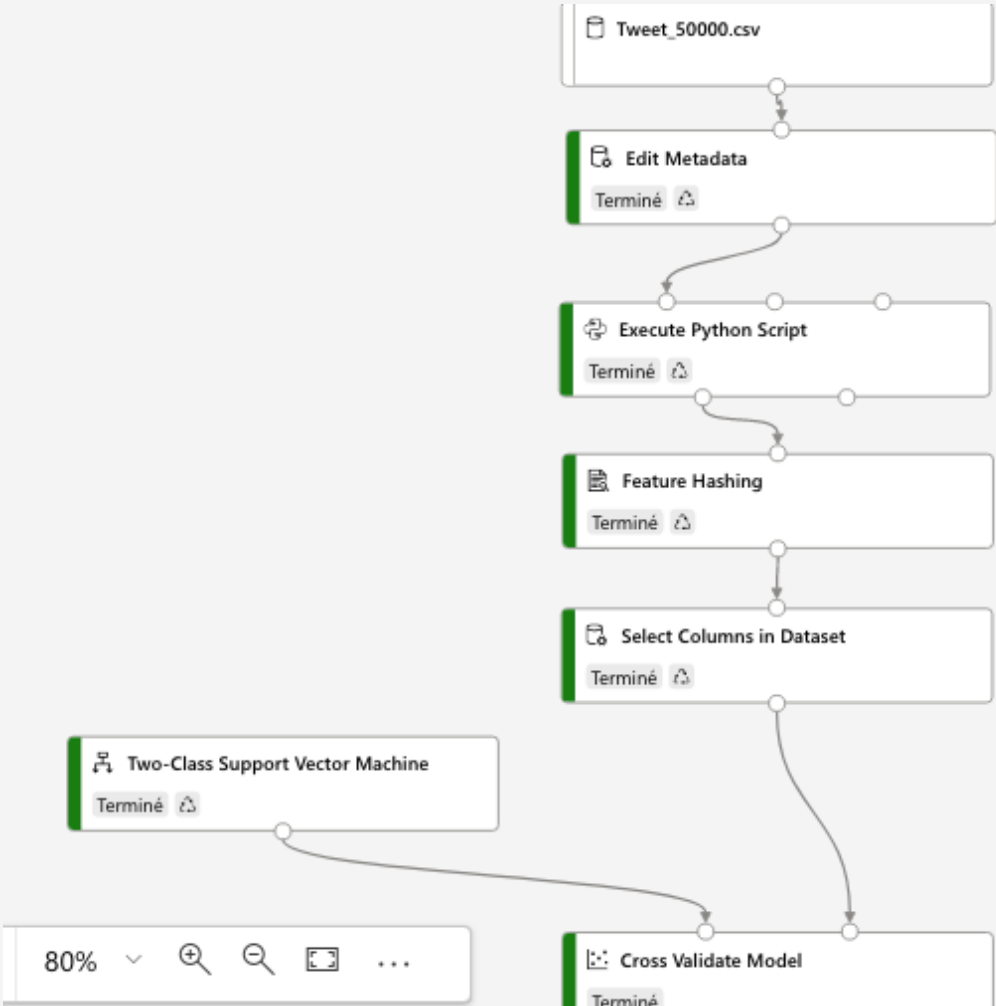
Nous avons retenu le modèle de classification binaire développé par la biais de l'interface graphique du studio Azure ML :

- le modèle Two-Class Boosted Decision Tree

Ils possèdent les briques suivantes :

- Lecture du jeu de données Tweet_50000.csv contenant les colonnes "sentiment" de valeur 0 ou 1, et "tweet" avec le texte des tweets
- Edit Metadata afin de modifier la colonne "sentiment" en type catégoriel
- Execute Python Script contenant un code pour le nettoyage des tweets (suppression des stop words, conversion en minuscules, lemmatisation)
- Feature Hashing pour convertir les mots en entiers
- Select Columns in Dataset afin de sélectionner les colonnes nécessaires au modèle
- Two-Class Boosted Decision Tree pour une classification binaire basée sur l'algorithme de l'arbre de décision boostée
- Cross Validate Model afin d'effectuer la validation croisée sur le jeu de données d'entraînement et de test

Two-Class Boosted Decision Tree



Résultat du modèle

Cross Validate Model visualisation des résultats

Evaluation results by fold							Scored results	
Lignes 12							Colonnes 11	
Fold Number	Number of examples in fold	AUC	Accuracy	Precision	Recall	F1		
0	5000	0.617154	0.5822	0.591977	0.574518	0.583117		
1	5000	0.610365	0.5784	0.586774	0.576151	0.581414		
2	5000	0.606979	0.5766	0.582211	0.56627	0.57413		
3	5000	0.606916	0.5706	0.576039	0.565769	0.570857		
4	5000	0.608956	0.5732	0.576109	0.57336	0.574731		
5	5000	0.607959	0.5774	0.572516	0.577374	0.574935		

3. Modèle sur mesure avancé

Ce modèle développé dans le studio Azure ML utilise un notebook codé en python et se base sur les réseaux récurrents de neurones avec des cellules LSTM. Ce modèle est déployé dans Azure Container Instances afin de l'utiliser comme service web pour la prédiction des sentiments de tweets.

```
1 def getModel():
2     embedding_layer = layers.Embedding(input_dim = vocab_length,
3                                         output_dim = Embedding_dimensions,
4                                         weights=[embedding_matrix],
5                                         input_length=X.shape[1],
6                                         trainable=False)
7
8     model = keras.Sequential([
9         embedding_layer,
10        layers.Bidirectional(layers.LSTM(100, dropout=0.3, return_sequences=True)),
11        layers.Bidirectional(layers.LSTM(100, dropout=0.3, return_sequences=True)),
12        layers.Conv1D(100, 5, activation='relu'),
13        layers.GlobalMaxPool1D(),
14        layers.Dense(16, activation='relu'),
15        layers.Dense(1, activation='sigmoid'),
16    ],
17    name="Sentiment_Model")
18    return model
```


✓

Architecture du modèle

- Embedding Layer : Layer responsable de la conversion des tokens en représentation vectorielle générée par Word2Vec
- Bidirectional: Traitement bidirectionnel du texte. Cela signifie que le contexte des tweets est traité de gauche à droite et de droite à gauche
- LSTM: Long Short Term Memory, C'est un variant du RNN contenant une cellule à mémoire interne pour apprendre le contexte à long terme des mots plutôt que les mots voisins effectué par le RNN classique.
- Conv1D: couche convolutionnel 1D
- GlobalMaxPool1D: Réduction de la dimension en entrée en prenant le maximum pour chaque Dimension
- Dense : couche dense

Apprentissage sur le jeu d'entraînement

```
1 history = training_model.fit(X_train, Y_train,  
2                               batch_size=1024,  
3                               epochs=12,  
4                               validation_split=0.1,  
5                               callbacks=callbacks,  
6                               verbose=1,  
7 )
```



Sauvegarde et enregistrement dans Azure ML

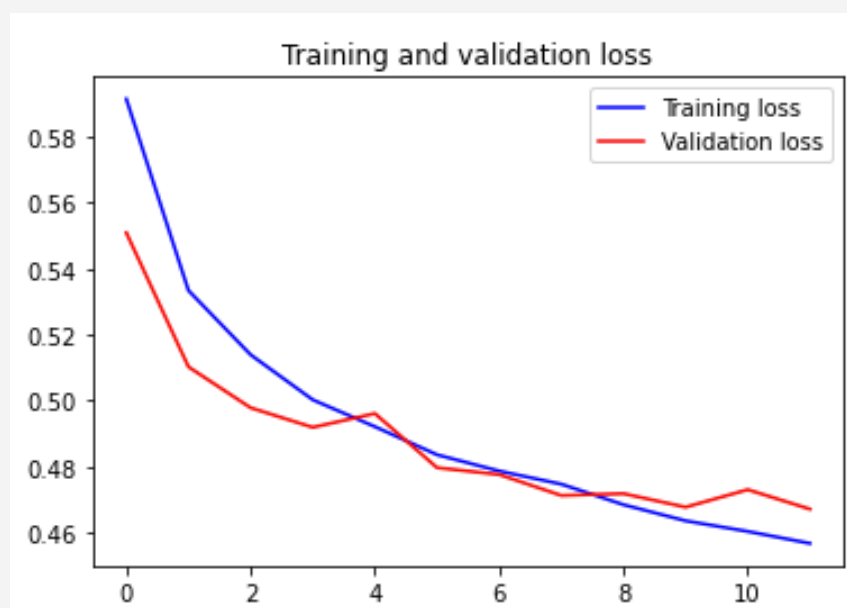
```
1 from azureml.core import Workspace
2 from azureml.core.model import Model
3 import os
4
5
6 ws = Workspace.from_config()
7 print(ws.name, ws.resource_group, ws.location, ws.subscription_id, sep='\n')
8
9 model = Model.register(workspace=ws,
10                        model_path="./outputs/Sentiment_Model.pkl",
11                        model_name="Sentiment_Analysis",
12                        description="Tweet Sentiment Analysis")
13
14 print('Name:', model.name)
15 print('Version:', model.version)
16
```

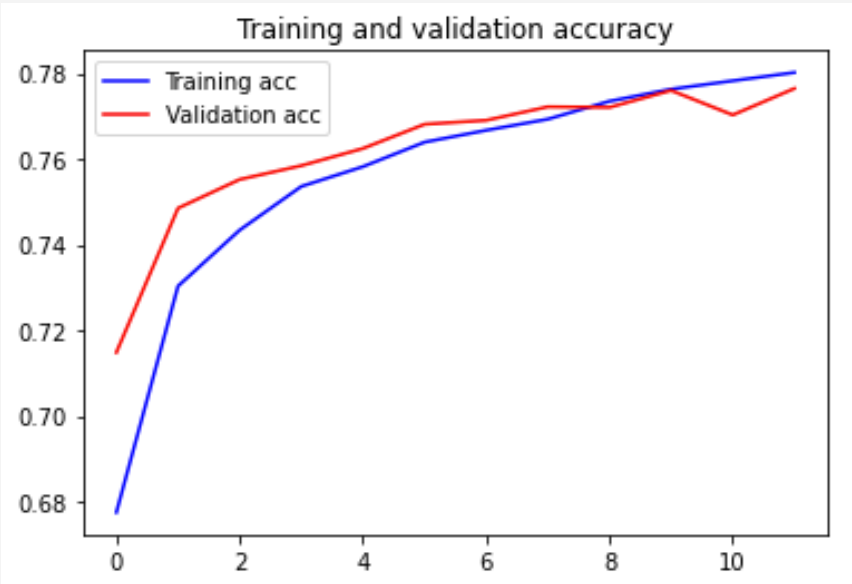
[29] ✓

...
workplace
modèles
centralus
eb572890-26d8-4841-b702-9e75c9697573
Registering model Sentiment_Analysis
Name: Sentiment_Analysis
Version: 8

Le modèle Sentiment_Analysis a été enregistré dans Azure ML

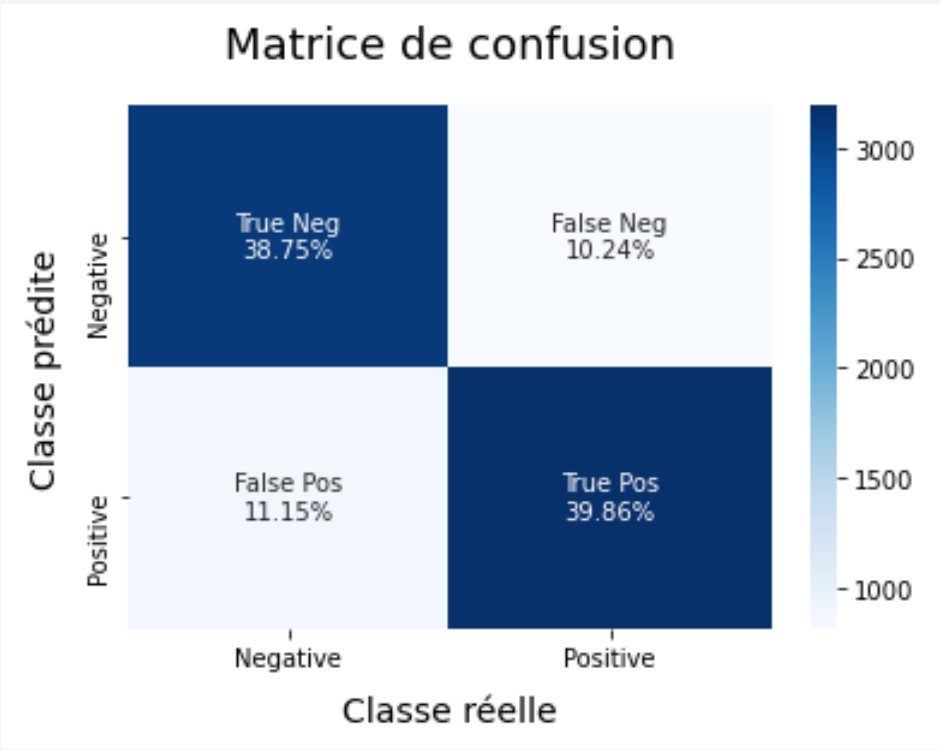
Evaluation des performances





Au dernier epoc, on obtient un accuracy de 77.6% et une fonction de perte à 46.7%

- Matrice de confusion :



- Rapport de la classification :

	precision	recall	f1-score	support
0	0.79	0.78	0.78	3992
1	0.78	0.80	0.79	4008
accuracy			0.79	8000
macro avg	0.79	0.79	0.79	8000
weighted avg	0.79	0.79	0.79	8000