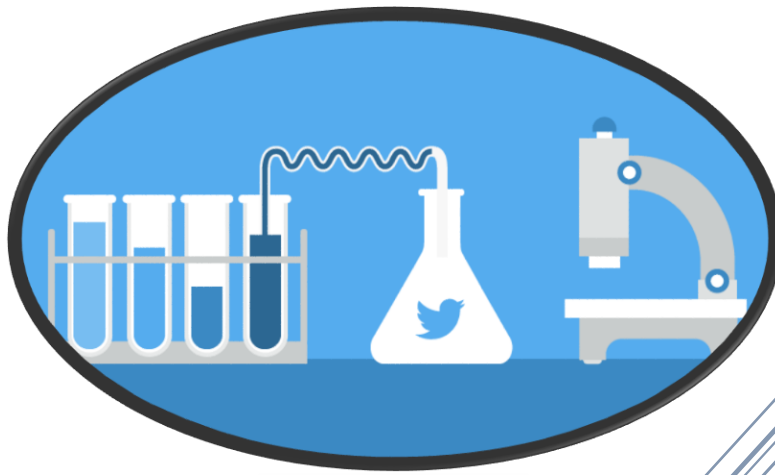


Streaming Text Analytics using Python

Alexis Ribat



Efrei Paris
Big Data Analytics

Summary

I. Création du compte développeur de Twitter 3

II. ApacheSpark 4

III. Partie A – Trends in Twitter..... 6

IV. Partie B – Real-time sentiment Analysis on
Twitter Topics..... 6

V. Conclusion 7

Dans ce rapport, nous voir comment se connecter à l'API de Twitter, récupérer des Tweets afin de les analyser.

I. Création du compte développeur de Twitter

Twitter a mis au point un moyen simple et rapide d'analyser leurs données via une API. Ce système est très utile pour les entreprises qui souhaitent avoir accès en temps réel à des données qui peuvent leur servir dans leurs choix stratégiques de communication.

Après avoir envoyé un mail au service de vérification de Twitter afin d'expliquer les tenants et les aboutissants de mon projet, j'ai eu mes accès de développeur pour me connecter.

J'ai pu créer une application avec les « API Key & secret » et « l'access Token & secret » qui sont mes identifiants propres à mon application.

Avec ces identifiants, j'ai pu m'amuser avec la librairie « Tweepy » (une librairie de Python pour l'accès au Twitter API) et j'ai pu télécharger mes tweets avec la fonction :

```
import tweepy

auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
auth.set_access_token(access_token, access_token_secret)

api = tweepy.API(auth)

public_tweets = api.home_timeline()
for tweet in public_tweets:
    print(tweet.text)
```

J'ai pu connecter mon application avec la fonction :

```
auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
```

On peut également « follow » des gens, voir notre liste d'amis ou même envoyer des Tweets via Tweepy.

L'étape suivante a été de configurer mon PC pour utiliser Spark Apache.

II. ApacheSpark

L'objectif d'Apache Spark Streaming est d'identifier les tweets et de les analyser afin de les retourner dans un Dashboard en temps-réel.

Spark Streaming apporte une API intégrée qui gère le traitement de flux (en utilisant le langage Python dans mon cas).



Spark Streaming fournit un haut niveau d'abstraction appelé DStream qui représente un flux de données. Concrètement, Spark Streaming fonctionne comme suit :



En Python, nous importons SparkContext. C'est le point d'entrée principal pour les fonctionnalités de streaming :

```
from pyspark import SparkContext
from pyspark.streaming import StreamingContext

# Create a local StreamingContext with two working thread and batch
interval of 1 second
sc = SparkContext("local[2]", "NetworkWordCount")
ssc = StreamingContext(sc, 1)
```

Ce contexte nous permet de créer un DStream qui représente les données de streaming à partir d'une source TCP (Protocol de transport fiable, ici « localhost » et son port (9999 par exemple).

```
lines = ssc.socketTextStream("localhost", 9999)
```

« lines » représente le flux de données reçu du data server. Chaque enregistrement de ce DStream est une ligne de texte. Ensuite, nous voulons diviser les lignes par espace en mots.

```
words = lines.flatMap(lambda line: line.split(" "))
```

“flatMap” crée un nouveau DStream en générant plusieurs nouveaux enregistrements à partir de chaque enregistrement dans le DStream source. Chaque ligne sera divisée en plusieurs mots et le flux de mots est représenté dans le wordsDStream, puis on compte les mots.

```
pairs = words.map(lambda word: (word, 1))
wordCounts = pairs.reduceByKey(lambda x, y: x + y)
wordCounts.pprint()
```

Le wordsDStream est mappé à un DStream de (word, 1) paires, puis réduit pour obtenir la fréquence des mots dans chaque lot de données. Enfin, wordCounts.pprint() imprime quelque-uns des comptes générés chaque seconde.

Afin de démarrer le traitement, nous appelons ssc.start() et ssc.awaitTermination().

III. Partie A – Trends in Twitter

Voir:

Code : « Apache Spark Streaming » & « Twitter_app »

Malheureusement, lorsque j'ai implémenté le code de Hanees 'Medhat Shousha' du site <https://www.toptal.com/apache/apache-spark-streaming-twitter>, j'ai eu une des erreurs « Value Error and TypeError ». En lisant les commentaires sous le post, j'ai vu que je n'étais pas le seul. J'ai essayé différentes techniques, sans succès. Je n'ai donc pas pu afficher les données sous forme de tableau.

IV. Partie B – Real-time sentiment Analysis on Twitter Topics

Voir:

Code : « Twitter Sentiment Analysis »

Dans cette partie, j'ai utilisé « TextBlob », une bibliothèque Python pour le traitement de données textuelles. Grâce à son API simple, elle permet l'analyse des sentiments, la classification etc...

Ainsi, j'ai pu extraire des tweets contenant le mot « Trump » et classer ceux qui étaient « positifs », « négatifs » ou « neutres » et afficher certains de ces Tweets. Les étapes principales sont :

- Autoriser le client API Twitter via nos codes secrets ;
- Envoyer une requête GET à l'API de Twitter pour récupérer les tweets d'une requête particulière ;

- Analyser les tweets et les classer à l'aide TextBlob.

V. Conclusion

Pour conclure, j'ai bien aimé travailler avec l'API de Twitter car elle est assez simple à comprendre et on arrive facilement à avoir des résultats acceptables. Cependant, j'aurai aimé pouvoir afficher les résultats sous forme de tableaux.


Également, **la mise en place du cluster Apache Spark m'a pris beaucoup de temps**. En effet, j'ai dû modifier beaucoup de choses dans la configuration de mon ordinateur, et lorsque j'ai finalement réussi à installer tout correctement, j'ai eu des erreurs liées à ma version de Python qui n'était pas compatible avec les fonctions que je voulais utiliser et celle d'Apache Spark. J'ai donc dû downgrade ma version de Python à la version 3.7.5, ce qui se fait assez bien via Anaconda en temps normal, mais n'y arrivant pas, j'ai dû tout désinstaller et réinstaller à la bonne version.

Finalement, j'ai réussi à avoir un environnement de travail qui fonctionnait sur ma machine, mais cela m'a pris beaucoup de temps. Cette phase a été fastidieuse et assez frustrante, mais j'ai beaucoup appris sur l'environnement Hadoop Spark, les variables d'environnements et les gestions des chemins d'accès via l'invit de commande !

Bien que ce sujet m'intéressât beaucoup au premier abord, je n'avais pas estimé la quantité de travail qui était à fournir simplement pour mettre en place de quoi commencer le projet. Pour info, j'ai commencé ce projet un mois et demi avant la date limite (par « commencer » je veux dire m'y mettre vraiment et lancer les démarches auprès de Twitter pour les accès du compte Twitter Développeur puis commencer à coder).

Une autre solution plutôt que passer autant de temps à configurer ma machine aurait été d'utiliser une VM avec tout d'installer déjà dessus. Malheureusement, bien que je sache configurer les VM, j'utilisais celles fournies par les professeurs et je ne connais pas de sites sécurisés pour en télécharger. Également, je n'avais pas conscience du temps que cela me prendrait de tout configurer et avec les autres projets des autres matières en parallèle et les DE à réviser, j'ai eu du mal à finir à bien ce projet, encore plus en le faisant tout seul.

Malgré tout, j'ai pu en apprendre plus sur l'environnement d'Hadoop et à lancer des requêtes via Python. Chaque environnement d'Hadoop apporte une solution qui lui est propre et il est intéressant de connaître toutes les alternatives afin de répondre au mieux aux besoins d'un projet. J'ai également appris à utiliser l'API de Twitter et à faire le lien avec Apache Streaming pour afficher des tweets en temps réel.



The End
(for now)