TP5 - Analyse de tweets

Analyse de sentiment appliquée aux tweets - PART 1
MongoDB, ElasticSearch et Kibana

L'analyse du sentiment exprimé par un utilisateur à travers un texte par informatique est un outil puissant. A travers des tweets, des commentaires ou des reviews postés sur le web, il s'agit de déterminer automatiquement l'image attachée à une marque ou un produit. Que ce soit pour les clients qui désirent choisir un produit apprécié ou pour les marques qui souhaitent comprendre l'impact de leur communication, l'analyse de sentiment peut s'avérer d'une grande aide. Nous nous intéresserons ici à la première étape de l'analyse de sentiment : l'analyse de polarité qui étiquette une review, un tweet ou un commentaire comme exprimant un sentiment négatif, neutre ou positif.

A travers ce premier TP, nous allons ainsi mettre en place une architecture capable de stocker, requêter et d'analyser un grand nombre de tweets étiquetés avec leur polarité. Pour cela, nous utiliserons le couple MongoDB/ElasticSearch : MongoDB pour le storage et ElasticSearch pour les requêtes de recherche d'information et de fouille dans les tweets. Nous installerons également Kibana pour restituer l'analyse des données stockées dans ElasticSearch à travers un Dashboard visuel et accessible aux gens du métier.

A rendre

Un export du Dashboard ("Save > export schema") ainsi que vos requêtes MongoDB, ElasticSearch doivent être envoyées à nicolas.dugue @univ-orleans.fr

Installation MongoDB + ElasticSearch

Ayant travaillé sous Ubuntu 12.04, vous pouvez lancer un conteneur Docker pour disposer de la meme install :

```
docker pull ubuntu:12.04
docker run -i -t ubuntu:12.04
```

Installer MongoDB par les paquets sur Linux : http://docs.mongodb.org/manual/tutorial/install-mongodb-on-ubuntu/

Télécharger et installer ElasticSearch : http://www.elasticsearch.org/overview/elkdownloads/

Après avoir installé MongoDB et ElasticSearch, nous allons maintenant les connecter. Pour cela, on utilise la river MongoDB d'ElasticSearch que l'on installe ainsi si elasticsearch a été décompressé dans /usr/share/ :

```
/usr/share/elasticsearch/bin/plugin -i
com.github.richardwilly98.elasticsearch/elasticsearch-river-mongodb/2.0.1
```

Pour connecter MongoDB et elasticsearch, tout d'abord, dans un terminal, lancer un serveur mongoDB en précisant que le mode réplication est activé :

```
sudo mongod –replSet testset
```

Dans un autre terminal, lancer le client MongoDB avec la commande **mongo**
Créer une db "infra_prod" avec la commande :

```
testset:PRIMARY> use infra_prod
```

Créer une collection "tweets_sentiment" et vérifier qu'elle existe avec :

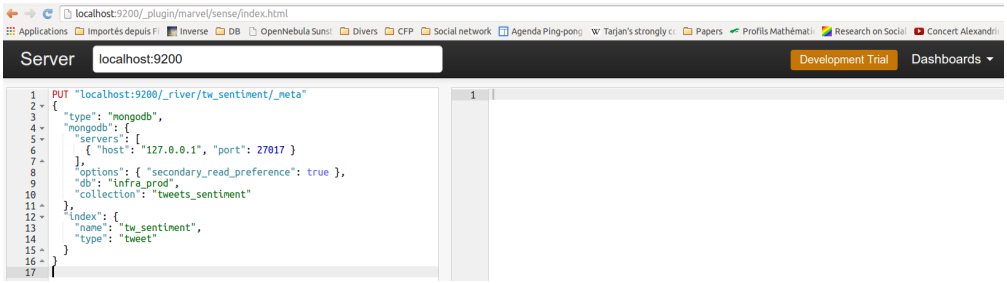
```
testset:PRIMARY> show collections
```

Maintenant, il s'agit de créer un index ElasticSearch. Dans cet index seront répliqués et indexés les documents stockés dans la db infra_prod de MongoDB. Pour interagir avec Elasticsearch, on pourra utiliser le plugin Sense pour le navigateur Chrome qui permet d'effectuer des requêtes REST pour ElasticSearch. On peut également installer Marvel, un plugin ElasticSearch (http://www.elasticsearch.org/guide/en/marvel/current/index.html#_simple_install) qui propose une implémentation de Sense. Néanmoins, Marvel est disponible seulement 7 jours en version d'essai. Nous utiliserons ici le client curl par souci de généricité mais les requêtes peuvent etre tapées directement dans Sense également :

```
curl -XPUT "localhost:9200/_river/tw_sentiment/_meta" -d '
{
 "type": "mongodb",
 "mongodb": {
  "servers": [
   { "host": "127.0.0.1", "port": 27017 }
  ],
  "options": { "secondary_read_preference": true },
  "db": "infra_prod", //Nom de la db mongo
  "collection": "tweets_sentiment" //Nom de la collection mongo
 },
 "index": {
  "name": "tw_sentiment", //Nom de l'index
  "type": "tweet" //Type des documents
 }
}'
```

Pour lire le JSON retourné par ElasticSearch, vous pouvez utiliser http://json.parser.online.fr/.

Avec Marvel/Sense :



Importer et requêter les données

Lancer ElasticSearch avec la commande

```
/usr/share/elasticsearch/bin/elasticsearch
```

En lançant plusieurs instances d'ElasticSearch sur la même machine, celles-ci se connectent automatiquement. Cela permet un scaling vertical simple et efficace. Pour vérifier que les noeuds sont bien ajoutés au cluster ES :

```
curl -GET 'http://localhost:9200/_cluster/health'
```

Vous devez normalement voir plusieurs nodes.

Nous allons maintenant importer les données, des tweets étiquetés avec des polarités positives ou négatives. Ces tweets forment un sous ensemble de ceux du jeu de données de http://help.sentiment140.com/

Au format CSV ces tweets suivent le format : sentiment, tweet_id, date, query, user, tweet. Le premier fichier ci-dessous contient 30 000 tweets étiquetés comme expriment un sentiment négatif, le second en présente 20 000 étiquetés de façon positive.

30000neg.csv 20000pos.csv

Téléchargez ces fichiers puis insérez les dans MongoDB via mongoimport :

```
mongoimport -d infra_prod -c tweets_sentiment --fieldFile FILE_DESCRIPTOR --type
csv --file FILETOIMPORT
```

FILE_DESCRIPTOR = fichier décrivant le format csv du fichier à importer avec un champ par ligne : télécharger fieldFile. FILETOIMPORT = 30000neg.csv || 20000pos.csv

Vérifier que les fichiers ont bien été importés dans MongoDB (requête count sur la collection). De même avec ElasticSearch :

```
curl 'localhost:9200/_cat/indices?v'
```

doc.count doit etre égal à environ 50 000 documents pour l'index tw_sentiment.

Requêtes :

Avec MongoDB, obtenir les dates minimum et maximum du jeu de données.

Avec MongoDB et ElasticSearch, obtenir les tweets postés par "rhidown".

Formulez les requêtes MongoDB et ElasticSearch capables de retourner les 10 utilisateurs ayant postés le plus postés de tweets positifs ou négatifs -> http://docs.mongodb.org/manual/tutorial/aggregation-zip-code-data-set/ http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/_executing_aggregations.html

Kibana : un Dashboard pertinent

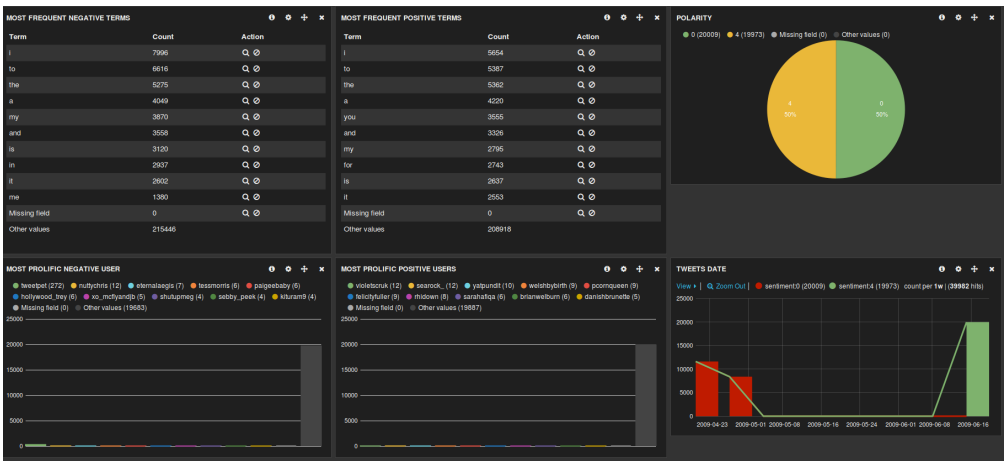
Installation de Kibana : http://www.elasticsearch.org/overview/kibana/installation/

L'objectif sera de restituer visuellement à travers des panels graphiques un résumé des données et les statistiques qui nous intéressent.

Sauvegardez votre Dashboard régulièrement avec la disquette en haut à droite, il doit être exporté à la fin du TP.

On souhaite ainsi être capable de :

- Requêter les tweets de polarité négative/positive : http://www.elasticsearch.org/guide/en/kibana/current/working-with-queries-and-filters.html ;
- Afficher sous forme de tableau les termes les plus fréquents dans les tweets positifs/négatifs du jeu de données : http://www.elasticsearch.org/guide/en/kibana/current/panels.html - Utiliser un terms panel
- Afficher sous forme d'histogramme les 10 utilisateurs ayant postés le plus postés de tweets positifs/négatifs (terms panel)
- Afficher un histogramme résumant les dates auxquels les tweets ont été postés (histogram panel)



Gestion de la Date

Pour consulter le type des objets stockés dans ElasticSearch :

```
GET /tw_sentiment/_mapping
```

Par défaut, le field date qui correspond au champ mongoDB est de type string. Il est donc impossible de l'utiliser pour tracer un histogramme en fonction du temps sur Kibana.

Pour obtenir des données de type date exploitable, mettre à jour la collection MongoDB en ajoutant un nouveau champ date_new de type date à tous les documents de la database. Pour cela, vous pouvez vous aider du code ci-dessous qui montre comment convertir une string en date et des **Cursor** MongoDB pour boucler sur tous les documents et les update :

```
testset:PRIMARY> var test = db.tweets_sentiment.findOne()
testset:PRIMARY> test.date
Mon Apr 06 22:20:03 PDT 2009
testset:PRIMARY> new Date(test.date)
ISODate("2009-04-07T05:20:03Z")
```

On obtient un nouveau mapping qui permet de dessiner l'histogramme des dates dans Kibana :

```
{
   "tw_sentiment": {
      "mappings": {
         "tweet": {
            "properties": {
               "date": {
                  "type": "string"
               },
               "date_new": {
                  "type": "date",
                  "format": "dateOptionalTime"
               },
               "query": {
                  "type": "string"
               },
               "sentiment": {
                  "type": "long"
               },
               "tweet": {
                  "type": "string"
               },
               "tweet_id": {
                  "type": "long"
               },
               "user": {
                  "type": "string"
               }
            }
         }
      }
   }
}
```

Un dataset équilibré

Avec MongoDB, faire en sorte, en supprimant des documents, d'avoir un jeu de données équilibré avec 50% de tweets étiquetés positifs, 50% de tweets étiquetés négatifs.

Dans la plupart des cas, avoir un jeu de données équilibré simplifie l'apprentissage du modèle destiné à analyser de futurs tweets.

Facultatif :

On constate que les mots les plus fréquents ne sont pas pertinents pour classifier les tweets, ils ne sont pas caractéristiques d'une polarité positive ou négative. Ce sont des mots anglais fréquents, peu importe la polarité de la phrase.

La plupart de ces mots sont appelés stopwords, il s'agit de les supprimer dans MongoDB pour améliorer la pertinence des informations restituées par le Dashboard Kibana.

Pour supprimer ces mots dans MongoDB, utilisez la liste de mots : englishStopWords

Par ailleurs, remplacer dans les fields "tweet" en utilisant des regex :

- les url par le mot clé URL;
- les mentions "@user_name" par le mot-clé USERNAME;
- Les répétitions de n lettres par des répétitions de 2 lettres seulement : les "huuuuuuge" par "huuge", les "goooo" par "goo".

Toutes ces opérations sur les données accéléreront la recherche et amélioreront la pertinence des futurs classifieurs.

Un premier classifieur naïf

En utilisant deux listes de mots anglais positifs et négatifs (http://www.enchantedlearning.com/wordlist/positivewords.shtml http://www.enchantedlearning.com/wordlist/negativewords.shtml), on souhaite créer un classifieur basique.

Ce classifieur va être implémenté en utilisant l'API ElasticSearch et notamment grace aux puissants Analyzers.

Les analyzers permettent de préciser une méthode pour indexer les documents et analyser des requêtes de recherche.

Exécutez par exemple le code suivant pour constater les effets d'un Analyzer capable de détecter des mots comme négatifs :

```
POST /classifieur_negs_stopwords/_close
PUT /classifieur_negs_stopwords
{
    "settings" : {
        "analysis" : {
            "char_filter" : {
                "negs" : {
                    "type" : "pattern_replace",
                    "pattern" : "\\b((?i:never|no)\\b",
                    "replacement" : "NEG"
                }
            },

            "analyzer" : {
                "negcon_tagger" : {
                    "type" : "custom",
                    "tokenizer" : "whitespace",
                    "filter" : ["lowercase", "kstem"],
                    "char_filter" : ["negs"]
                }
            }
        }
    }
}
POST /classifieur_negs_stopwords/_open
GET /classifieur_negs_stopwords/_analyze?analyzer=negcon_tagger&text=The boy was
not happy. He is never happy. Blablabla.
```

Exécutez ce second exemple pour constater qu'ElasticSearch est capable de retirer lui meme les stopwords de ses requêtes :

```
PUT /english_docs
{
    "settings": {
        "analysis": {
            "analyzer": {
                "es_std": {
                    "type":      "standard",
                    "stopwords": "_english_"
                }
            }
        }
    }
}
GET /english_docs/_analyze?analyzer=es_std&text=The boy was not happy. His friend
bangged his mother. Too bad...
```

Il s'agit donc maintenant de créer un Analyzer capable de supprimer les stopwords des requêtes et de remplacer les mots à connotations négatives par NEG, celles à connotations positifs par POS.

Vous modifierez ensuite le mapping de votre index en ajoutant un champ tweet_new dont l'analyzer sera celui que vous avez créé.

Dans MongoDB, vous copierez pour toutes les entrées le contenu du field tweet dans un nouveau field tweet_new. Celui-ci sera ainsi indexé en utilisant votre nouvel Analyzer.

Ainsi, une fois ces taches effectuées, en recherchant des documents de la façon qui suit, il vous sera possible d'obtenir les documents à polarité négatives :

```
GET /tw_sentiment/_search?q=tweet_new:NEG
```

Doc : http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/analysis-custom-analyzer.html

http://www.elasticsearch.org/guide/en/elasticsearch/guide/current/configuring-analyzers.html

http://www.elasticsearch.org/guide/en/elasticsearch/guide/current/mapping-intro.html

http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/indices-analyze.html