

# Identification de partenariats entre acteurs économiques

Azim Makboulhoussen

*OpenClassrooms*  
*Projet 8 – Parcours Data Scientist*  
*05 Août 2018*



## Table des matières

<b>I.</b>	<b>Introduction .....</b>	<b>3</b>
<b>II.</b>	<b>Présentation de Snorkel .....</b>	<b>3</b>
1.	Quel est l'objectif de Snorkel ? .....	3
2.	Comment fonctionne Snorkel ? .....	4
3.	Comment utiliser Snorkel ?.....	7
<b>III.</b>	<b>Mise en application de Snorkel .....</b>	<b>8</b>
1.	Présentation du cas testé .....	8
2.	Les différentes étapes de l'implémentation.....	9
3.	Évaluation des résultats.....	11
<b>IV.</b>	<b>Conclusion .....</b>	<b>12</b>

# **I. Introduction**

Les algorithmes de Machine Learning connaissent un succès indéniable. Néanmoins pour être efficaces, ces algorithmes ont besoin d'apprendre sur des quantités massives de données.

Le volume de données et d'information ne cessent d'augmenter de nos jours cependant une grande partie de ces données n'est pas exploitable directement par les machines car non structurées. Pour appliquer l'apprentissage machine, il faut alors réaliser un travail minutieux et demandant souvent de l'expertise du domaine pour préparer ces données dites d'entraînement. Ce travail complexe et long est souvent un frein à l'utilisation de l'apprentissage machine dans de nombreux domaines.

Afin de répondre à cette contrainte, une équipe de recherche de l'université de Stanford a développé la solution Snorkel, un système facilitant la génération de données d'entraînement.

Dans le cadre du projet 8 du parcours Data Scientist, nous avons choisi de faire une veille thématique sur cette solution.

Dans ce document, nous commencerons par une explication théorique du système Snorkel puis nous le mettrons en pratique afin d'extraire dans des articles de presse les relations de partenariat entre entreprises.

## **II. Présentation de Snorkel**

### **1. Quel est l'objectif de Snorkel ?**

De plus en plus de données sont collectées mais près de 80% de celles-ci ne sont pas exploitées aujourd'hui car elles sont stockées dans des formats difficilement lisibles par les machines. Ces données non utilisées et pourtant précieuses sont appelées Dark Data.

C'est particulièrement le cas dans le domaine biomédical et scientifique avec les imageries, les rapports scientifiques et les notes qui sont plus adaptés à la lecture humaine que des ordinateurs.

Pour extraire ces données de grandes valeurs, il faut mettre en place des solutions complexes afin de les formater correctement pour ensuite pouvoir les analyser.

Partant de ce constat, une équipe de chercheurs de Stanford a développé la solution Snorkel. Snorkel a pour objectif de faciliter la mise en œuvre de solutions prédictives ainsi que l'exploitation des Dark Data. Elle s'attaque au principal goulet d'étranglement dans l'apprentissage machine en facilitant la création des jeux de données d'entraînement.

Prenons l'exemple de scientifiques souhaitant utiliser l'apprentissage machine pour prédire s'il existe ou pas un lien entre un produit chimique et une maladie à partir de la littérature scientifique.



Dans un projet classique de Machine Learning, les étapes sont les suivantes :

1. Extraire des couples Produit Chimique / Maladie à partir de chaque phrase de chacun des documents
2. Création d'un jeu d'apprentissage. Pour cela la contribution d'un expert est nécessaire pour indiquer si oui ou non le lien entre le couple existe effectivement dans le texte.
3. Ensuite il faut y ajouter des features pour donner à la machine beaucoup de petites informations qu'elle utilisera pour faire ses prédictions. Par exemple quels sont les mots que nous trouvons entre les 2 entités, ...
4. Enfin nous pouvons utiliser un algorithme de classification pour la phase d'apprentissage et faire par la suite les prédictions sur de nouvelles données.

Il existe aujourd'hui de nombreuses librairies facilitant grandement la mise en œuvre de la dernière étape : l'apprentissage. Ces librairies offrent des algorithmes puissants qui peuvent être appliqués sans trop de difficultés.

L'avènement du Deep Learning a en outre permis d'automatiser l'extraction des features qui demandaient jusque-là une certaine ingénierie.

Mais en contrepartie, pour nourrir ces algorithmes il faut leur fournir une quantité massive de données.

C'est donc l'étape de construction des données d'entraînement qui est la plus coûteuse. Ces données doivent être annotées ce qui demande non seulement du temps mais surtout une certaine expertise du domaine. C'est donc la partie la plus fastidieuse à produire dans les projets d'apprentissage machine de nos jours.

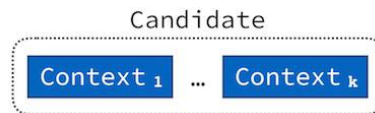
Snorkel est un projet open source qui aide les utilisateurs à créer de large quantité de données d'entraînement simplement de manière programmatique. Elle évite ainsi le travail pénible de devoir labéliser à la main chacune des données d'entraînement et a pour ambition de faciliter l'utilisation des techniques d'apprentissage machine dans de nombreux domaines.

## 2. Comment fonctionne Snorkel ?

L'idée derrière le système Snorkel est qu'au lieu de créer des données labélisées manuellement nous pouvons créer des modèles performants en nous basant sur des données annotées de manière programmatique. En modélisant notre processus de création de jeu de données d'entraînement, nous pouvons utiliser des sources de données de faible qualité pour entraîner des modèles à haute performance.

Snorkel se base sur la Data Programmation qui consiste à écrire des scripts qui vont étiqueter les données.

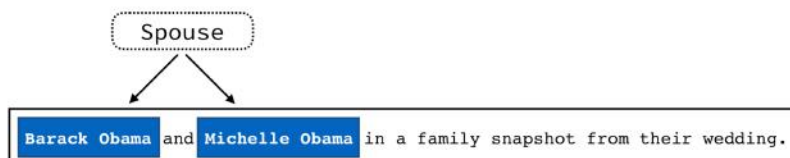
### a) Candidat et Contexte



Snorkel utilise un ensemble d'objets de base servant à modéliser nos données. Dans un projet Snorkel, il faut commencer par définir un **Candidat**. Un **Candidat** représente les mentions potentielles que nous voulons extraire de nos données et donc la prédiction que nous souhaitons réaliser. Un candidat est défini sur la base d'un ou plusieurs objets **Contexte**. Un **Contexte** est simplement une unité textuelle comme un mot ou une phrase.



Par exemple, si on veut construire une base de connaissance de couples mariés en réalisant une extraction des relations de mariage à partir d'article de presse, nos candidats seront toutes les paires de noms de personnes trouvés dans nos documents.



### b) Les fonctions de labélisation

La deuxième étape consiste à écrire les fonctions de labélisation ou **Labeling Functions** en Anglais (LF).. Les LF sont simplement des règles métiers programmées par l'utilisateur et qui vont permettre d'annoter un sous ensemble de nos données.

Dans le cas d'une classification binaire, la fonction va retourner 1 pour un label positif, -1 pour un label négatif et 0 si elle s'abstient de labéliser.




Les fonctions de labélisation s'appliquent sur nos **Candidats**. Les fonctions de labélisation sont appliquées à l'ensemble des candidats et doivent prédire à la fois les labels positifs et négatifs.

Snorkel permet d'écrire deux types de **Labeling Functions** :

- Les **Pattern Based LFs**. Elles sont implémentées à l'aide d'expression régulières et d'autres méthodes d'analyse de texte.
- Les **Distant Supervision LFs**. Elles vont chercher les informations sur d'autres sources comme des bases de connaissances externes (ex : DBPedia, WikiData, ...) pour labéliser les données.

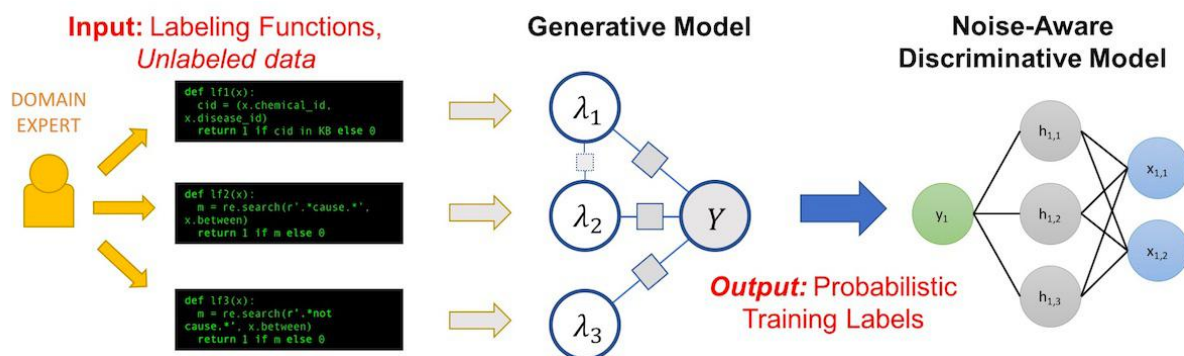
Exemple de LF :



```
def LF_same_last_name(c):
    """
    Label as positive if both
    """
    p1_last_name = last_name(c.person1.get_span())
    p2_last_name = last_name(c.person2.get_span())
    if p1_last_name and p2_last_name and p1_last_name == p2_last_name:
        if c.person1.get_span() != c.person2.get_span():
            return 1
    return 0
```

Comme les LF sont juste des bouts de codes, il est alors facile de les modifier, les réutiliser et les adapter à d'autres problématiques. Elles sont écrites directement dans le notebook Jupyter et peuvent être testées et débuggées simplement.

### c) Le workflow Snorkel



Le flux Snorkel comprend 3 grandes étapes :

1. Extraction des candidats
2. Application des LFs sur les données et apprentissage d'un modèle génératif. Celui-ci sera en charge d'apprendre un modèle statistique pour unifier les LFs.
3. Entraînement d'un modèle discriminatif sur la sortie du modèle génératif pour la prédiction finale.

### L'utilisateur :

1. Charge les données non labélisées
2. Écrit les fonctions de labélisation (LFs)
3. Choisit un modèle discriminatif pour sa prédiction

### Snorkel :

1. Les données sont analysées et les candidats sont extraits
2. Les LFs sont appliquées aux candidats.
3. Un modèle génératif est utilisé pour apprendre la précision et les corrélations des fonctions de labélisation et un poids est affecté à chaque LF.
4. Le modèle génératif va produire une liste de probabilité pour les labels d'entraînement. Cette liste est alors utilisée pour entraîner un modèle discriminatif pour faire les prédictions finales.

Snorkel va donc s'appuyer sur un ensemble de LFs afin d'annoter les données d'apprentissage. Les sorties de ces fonctions contiendront probablement du bruit, des chevauchements ou des conflits rendant certainement les annotations imparfaites.

Mais la puissance de Snorkel vient justement de sa capacité à apprendre sur ces imperfections.

Dans le modèle génératif, Snorkel va comparer les points d'accords et de désaccords des LFs pour apprendre. Un modèle génératif modélise la façon dont les données ont été générés. Il apprend la distribution conjointe de  $x$  et  $y$ .

Le modèle discriminatif qui sera appliqué par la suite, ne va pas se soucier de la façon dont les données ont été générés. Il apprend la probabilité de  $y$  sachant  $x$ . Les modèles de Deep Learning sont utilisés pour cette étape.

Un petit jeu de données annotés manuellement va nous aider à affiner la qualité des fonctions de labélisations. Nous appellerons cet ensemble, *données de développement*. Ce jeu de développement est également utilisé dans la phase d'apprentissage de la modélisation discriminative afin d'évaluer le modèle.

Nous aurons également besoin d'un jeu de test également annotés. Celui-ci ne sera pas du tout utilisé pendant l'apprentissage et va nous servir à évaluer la qualité de prédiction de notre système sur de nouvelles données.

## 3. Comment utiliser Snorkel ?

La compréhension de la philosophie Snorkel demande un certain investissement mais sa mise en œuvre est plutôt simple. Les équipes de Snorkel ont par ailleurs mis à disposition un grand nombre de documentations, d'articles de publication de recherche, de tutoriaux sur le site de Snorkel afin d'aider à sa prise en main :

<https://hazyresearch.github.io/snorkel/>

L'installation est assez simple. Il faut suivre la procédure décrite ici :

<https://github.com/HazyResearch/snorkel#installation>

Une fois Snorkel installé, pour comprendre son fonctionnement, il est conseillé de suivre le tutorial d'introduction qui décrit bien les 3 étapes principales de l'utilisation de la solution Snorkel :

1. Le prétraitement et l'analyse syntaxique des documents avec l'instanciation des objets **Candidates**.
2. L'écriture des fonctions de labélisations et l'application du modèle génératif
3. L'application du modèle discriminatif pour la prédiction finale.

### **III. Mise en application de Snorkel**

#### **1. Présentation du prototype mis en place**

Pour tester et évaluer la solution Snorkel, nous avons choisi d'extraire les relations de partenariat entre acteurs économiques à partir d'article de presse.

Pour cela, nous sommes partis d'extraits d'articles de presse fournis par la société GeoTrend. GeoTrend développe une plateforme de Business Discovery et elle souhaitait évaluer le système Snorkel pour sa solution.

Les données fournies étaient composées de phrases anglaises extraites d'articles et annotées avec les relations entre acteurs économiques sous le format suivant :

- actor1; actor2; type; sentence
- actor2; actor1; type; sentence

*Exemple :*

ALTB;Boeing;CAPITALISTIC;"Boeing, Northrop Grumman, and Lockheed are developing the ALTB for the Missile Defense Agency which calls the airborne laser program "a pathfinder" for the nation\'s directed energy program and for missile defense technology, Boeing is the prime contractor on the project, providing the aircraft and the battle management systems, Lockheed developed the beam control/fire control system which sits at the front of the aircraft, and Northrop is in charge of the megawatt-class chemical, oxygen, and iodine laser (COIL)."
---

Notre corpus est composé de 645 phrases mentionnant plus de 450 entreprises.

Les types de relation entre 2 entreprises annotés dans nos données sont :

- Partenariat,
- Concurrence,
- Clients,
- Alias,
- Procès,
- Filiale
- et Non Défini

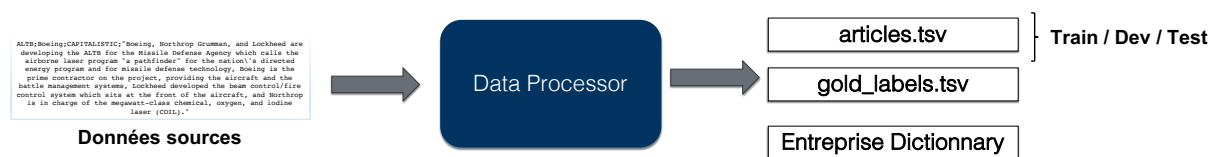


Notre objectif était de monter en compétence de manière progressive. Nous avons donc choisi dans ce projet de nous limiter à extraire qu'un seul type de relation, celui de partenariat. Nous avons donc implémenter une classification binaire indiquant un partenariat ou pas. Il aurait fallu mettre en place une classification multi-classes pour prendre en compte toutes les types de relations.

## 2. Les différentes étapes de l'implémentation

### a) Préparation des données

Dans un premier temps nous avons mis en forme nos données pour pouvoir les utiliser dans les notebooks Snorkel.



Pour cela nous avons :

1. Créé un nouveau fichier de données contenant uniquement les phrases et rattaché à un identifiant unique de document.
2. Générer nos jeux d'entraînement, de développement et de test. Pour les jeux de développement et test nous avons dû faire un traitement particulier pour y ajouter les annotations de relations
3. Générer un dictionnaire des noms d'entreprise

### b) Définition du schéma et extraction des candidats

Nous avons modéliser un candidat comme une relation binaire entre 2 noms d'entreprises.



Puis nous avons utilisé les fonctionnalités de Snorkel pour extraire l'ensemble de nos candidats de notre corpus. Pour cela nous nous sommes basés le dictionnaire des noms d'entreprises que nous avons générés.

### c) Fonctions de labellisation et modèle génératif

Nous nous sommes basés sur l'outil, **SentenceNGramViewer** pour nous aider à écrire les fonctions de labélisation. Cet outil affiche chacun des candidats détectés par Snorkel dans leur contexte. Nous en écrit plus d'une douzaine de LFs en nous appuyant sur des expressions régulières mais aussi sur des fonctions fournis par Snorkel facilitant les « *matchings* » (**helpers functions**).

Exemple de LF :

```
acquisitions = ['acquir(e|ed|es|ing){1}', 'acquisition', 'purchase(s|d)?', 'buy(ing)?', 'bought', 'parent']

def LG_acquisitions_between(c) :
    return -1 if re.search(r'{A}.{0,100}' + ltp(acquisitions) + '{0,100}{B}', get_tagged_text(c), re.I) else 0
```

```
def LF_clients(c):
    clients = { 'by', 'from', 'include' }
    return -1 if len(clients.intersection(get_left_tokens(c[1], window=3))) > 0 else 0

labeled = coverage(session, LF_clients, split=0)
tp, fp, tn, fn = error_analysis(session, LF_clients, split=1, gold=L_gold_dev)
```

- Chaque fonction de labellisation est appliqué aux candidats de nos données d'entraînement afin d'obtenir une matrice de labels.
- Cette matrice est alors utilisée pour entraîner un modèle génératif. On obtient en sortie la probabilités de partenariat de chacun des candidats de notre jeu d'entraînement. Une valeur proche de 1 indiquant une forte probabilité de partenariat et 0 le contraire. Cette liste est appelée **marginals**.

### Métrique d'évaluation

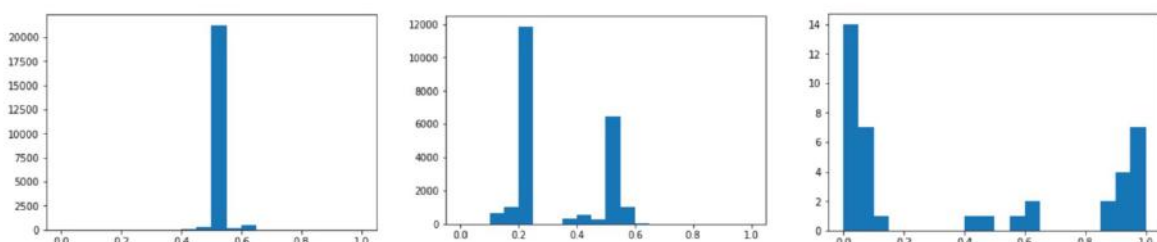
Pour déterminer la qualité et optimiser nos fonction de labélisation, nous nous sommes basés sur les métriques suivantes :

- **Accuracy** : représente le pourcentage de candidats que la fonction labellise correctement.
- **Coverage** : représente le pourcentage de candidats labélisés par une ou plusieurs LF.
- **Conflit** : représente le pourcentage de candidats dont les LFs donnent des résultats contradictoires.

Ces métriques ont été obtenus sur le jeu de développement contenant les annotations. Une bonne fonction est une fonction ayant une forte couverture (coverage) et une bonne précision (accuracy). Les conflits sont plutôt bienvenus car ils permettent à l'algorithme d'apprendre.

Nous avons également utilisé les métriques classiques de **précision**, **rappel** et **F1-score** dans notre évaluation.

Nous nous sommes appuyés sur la distribution des marginals pour analyser la qualité de notre modèle génératif. Idéalement on doit obtenir une distribution bimodale avec une franche séparation entre les 0 et les 1 comme la figure complètement à droite (les 2 autres figures montrent une mauvaise qualité du modèle).



#### d) Application d'un modèle discriminatif

Nous avons ensuite utilisé le résultat du modèle génératif pour entraînement notre modèle final. Cette dernière modélisation permettant de mieux généraliser la prédiction et donc d'améliorer la qualité de prédiction sur des données nouvelles.

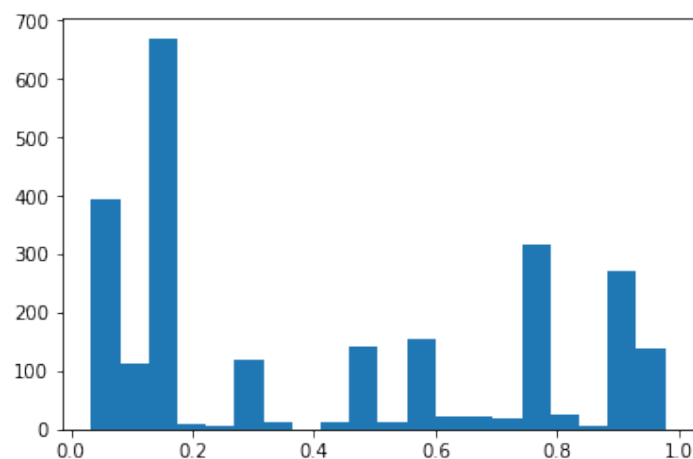
Nous avons utilisé l'algorithme **LSTM** (Long Short-Term Memory) qui fait partie des algorithmes de type réseau de neurones récurrents. C'est un algorithme qui est particulièrement efficace pour les tâches de classification de texte.

Snorkel fourni un lien permettant d'appliquer pour l'apprentissage discriminatif tous les modèles TensorFlow. Il intègre également nativement le modèle **LSTM** et un modèle classique de **Régression Logistique**.

Nous avons ensuite utilisé notre jeu de test pour vérifier la qualité de notre prédiction sur de données jamais vu par le modèle.

### 3. Les résultats

Pour le modèle génératif, nous avons obtenu la distribution des marginals suivante :



La distribution n'est pas complètement équilibrée car nous avons une couverture négative plus importante (gauche). Il faudrait écrire plus de LFs positives pour mieux répartir les probabilités.

Scores sur le jeu de développement :

	Précision	Rappel	F1-Score
Modèle génératif	37,9 %	58,1 %	45,9 %
LSTM	39,7%	62,8%	48,6%

#### Scores sur le jeu de test :

	Précision	Rappel	F1-Score
<b>Modèle génératif</b>	44,8 %	61,5 %	51,9 %
<b>LSTM</b>	38,6 %	83,5%	52,8 %

Nous constatons que les scores sont améliorés après l'application de la dernière phase (discriminatif). Le modèle discriminatifs augmente à chaque fois le F1-score. Il prend près de 3 points pour les données de développement et 1 point pour les données de tests.

Ces résultats devraient pouvoir être améliorés notamment en utilisant beaucoup plus de données car l'efficacité des algorithmes d'apprentissage profond (LSTM) dépend de la masse de données utilisée pour l'apprentissage. Nous avons une volumétrie de données relativement faible. Un travail sur les LFs pour en augmenter le nombre et la précision devrait également impacter la qualité des prédictions. Enfin, il serait judicieux de s'appuyer une base de connaissance externe et écrire de nouvelles LFs basée sur cette nouvelle source.

Dans notre cas, nous n'avions malheureusement pas de baseline pour comparer les scores données par Snorkel par rapport à un autre modèle. Cependant Snorkel a clairement faciliter la génération de données d'entraînement.

Au minimum, deux jours de travail à temps plein auraient été nécessaire pour annoter l'ensemble des articles alors que nous avons qu'un petit jeu de données. L'écriture des LF n'a pas pris plus d'une journée. Nous avons donc clairement gagner en temps et c'est bien l'objectif de la solution Snorkel.

## **IV. Conclusion**

Les algorithmes de Machine Learning et particulièrement de Deep Learning s'appuient sur des quantités importantes de données d'entraînement qui nécessitent souvent d'être annotés manuellement. Il est donc pas aisé d'obtenir de telles données.

Snorkel permet de remédier à cette problématique. Snorkel va nous libérer de cette tâche d'annotation manuelle en générant automatiquement un large volume de données en se basant sur des fonctions de labellisation programmés par l'utilisateur. Snorkel est particulièrement adapté dans le développement d'application d'extraction d'information dans des domaines dans lesquels il est difficile d'obtenir les données structurés et annotés.

Snorkel demande un certain effort pour maîtriser correctement ses concepts et comprendre précisément chacune des étapes dans le pipeline de construction du

programme. Cependant il est ensuite assez simple à mettre en œuvre et il dispose de nombreux tutoriaux explicatifs.

Nous avons pu l'appliquer sur un cas concret d'extraction de relation. Les scores obtenus ne sont pas très élevés. Mais ils doivent être améliorables notamment en travaillant les fonctions de labellisation, en augmentant la quantité de données et en optimisant l'algorithme discriminatif. Dans notre cas, nous avons surtout gagné un temps considérable par rapport à une solution avec annotation manuelle.

Snorkel est toujours en cours de recherche et il intègre régulièrement des améliorations. C'est donc un système que nous recommandons très vivement pour tous les projets dans lesquels les données ne seraient pas structurées et dont l'annotation serait compliquée.